Proceedings of the 7th Workshop on Detection and Classification of Acoustic Scenes and Events (DCASE 2022)

Mathieu Lagrange, Annamaria Mesaros, Thomas Pellegrini, Gael Richard, Romain Serizel, Dan Stowell

Proceedings of the 7th Workshop on Detection and Classification of Acoustic Scenes and Events (DCASE 2022)

Mathieu Lagrange, Annamaria Mesaros, Thomas Pellegrini, Gaël Richard, Romain Serizel and Dan Stowell (eds.)

November 3-4 2022

Citation:

Mathieu Lagrange, Annamaria Mesaros, Thomas Pellegrini, Gaël Richard, Romain Serizel and Dan Stowell (eds.), Proceedings of the 7th Workshop on Detection and Classification of Acoustic Scenes and Events (DCASE 2022), Nov. 2022.

ISBN: 978-952-03-2677-7

Table of Content

DCASE workshop 2022 sponsors

INTEGRATING ISOLATED EXAMPLES WITH WEAKLY-SUPERVISED SOUND EVENT DETECTION: A DIRECT APPROACH

Mohammad Abdollahi, Romain Serizel, Alain Rakotomamonjy, Gilles Gasso

Universite de Rouen

ABSTRACT

In an attempt to mitigate the need for high quality strong annotations for Sound Event Detection (SED), an approach has been to resort to a mix of weakly-labelled, unlabelled and a small set of representative (isolated) examples. The common approach to integrate the set of representative examples into the training process is to use them for creating synthetic soundscapes. The process of synthesizing soundscapes however could come with its own artefacts and mismatch to real recordings and harm the overall performance. Alternatively, a rather direct way would be to use the isolated examples in a form of template matching. To this end in this paper we propose to train an isolated event classifier using the representative examples. By sliding the classifier across a recording, we use its output as an auxiliary feature vector concatenated with intermediate spectro-temporal representations extracted by the SED system. Experimental results on DESED dataset demonstrate improvements in segmentation performance when using auxiliary features and comparable results to the baseline when using them without synthetic soundscapes. Furthermore we show that this auxiliary feature vector block could act as a gateway to integrate external annotated datasets in order to further boost SED system's performance.

Index Terms— sound event detection, deep learning, posteriorgrams, weakly-supervised learning

1. INTRODUCTION

Sound Event Detection (SED) is a machine listening task that addresses the questions of *What* and *When* of occurring audio events, the responses to which would be the perceptual class of an event and its location in time respectively. The necessity of such SED systems manifests itself in applications such as audio information retrieval [1], surveillance [2], bioacoustic monitoring [3] and self-driving cars [4] to a name a few. One main challenge with SED is the inherent difficulty and cost of acquiring annotated data of high quality/resolution. Hence there has been a growing body of research in development of models using lower quality annotations (lower temporal resolution, noisy labels, etc.) and un-annotated data examples potentially augmented with a small set of (isolated) representative examples of each target event class.

Isolated examples are generally used indirectly through synthesizing soundscapes [5], to generate *strongly-labelled* examples. The relative contribution of each level of annotations in this heterogeneous dataset has been studied by Turpault et al [6]. Surprisingly, their findings indicate the relative equal contribution of both sets of weak and strongly (synthetic) labelled examples to system's segmentation performance. In pre-deep learning speech recognition, *Phoneme posteriorgrams* output by pre-trained phoneme classifiers were used as features fed to the downstream blocks[7]. In spoken keyword detection, Chen et al. [8] trained a deep neural network (DNN) to classify among a number of keywords and by sliding the DNN across an audio recording used the output scores (followed by post-processing) to mark the keyword boundaries. Similarly, isolated examples could be used for *template matching* by sliding them across recordings to generate similarity scores and using these scores for detection of event boundaries.

In this paper we propose to integrate a similar approach into training the SED system. Rather than using the similarity scores directly for decision making, we will use them as auxiliary features to enhance the performance of the Weakly-Supervised SED (WSSED) system. Beyond the isolated examples, we also investigate the utility of using classifiers trained on other (non-target) external datasets (e.g. ESC-50) to generate the auxiliary features and report performance gains.

2. BASELINE SED SYSTEM

In this work, we have chosen a CNN-Transformer architecture proposed by Myazaki et al. [9] as our baseline model. Similar to a Convolutional Recurrent Neural Network (CRNN), CNN-Transformer uses a Convolutional Neural Network (CNN) for extraction of intermediate representations, but instead of a RNN, it is followed by

Thanks to ANR agency for funding the project LEAUDS.

few layers of transformer encoder for temporal dependency modelling.

Given a time-frequency representation of an audio signal as input (e.g. Mel-spectrogram) $\boldsymbol{X}_{[F^i \times T^i]}$ where F^i and T^i represent the dimensions of the input along frequency and time respectively, a CNN extracts a representation $\boldsymbol{H}_{[C \times F^e \times T^e]}$ with F^e and T^e being the resulting (downsampled) frequency and time dimensions and C the number of extracted feature maps. Merging the F^e and C dimensions, would result a T^e-long sequence of $d = C \times F^e$ dimensional feature vectors $\boldsymbol{h}_i$:

$$\boldsymbol{H} = \{\boldsymbol{h}_1, \ldots, \boldsymbol{h}_{T^e}\} \tag{1}$$

Every element of this sequence is first linearly projected to a lower dimensional space using an *Embedding layer*. The resulting sequence of embedding vectors, after being added to the respective positional encoding vectors are passed through N layers of transformer encoders.

A final single-layer Multi-layer Perceptron (MLP) is applied on all elements of the output sequence of the transformer to predict the segment-based labels. The clip-level label is obtained by performing attention-pooling [10] over all elements in the output sequence.

3. PROPOSED APPROACH

In this work we take a more direct approach in integrating isolated examples into training a WSSED system. Rather than synthesizing soundscapes with them, we propose to use them to build an isolated event classifier. The idea is that, given a large enough set of isolated examples of the target events, we could build a classifier operating on a certain time scale and slide it across test audio examples in order to generate similarity/likelihood scores. These scores could then be used (in addition to some post-processing) to detect target event boundaries. Such an approach would be very similar to the recent line of work on Keyword Spotting (KWS) [8, 11]. Unlike in the case of KWS however, our set of examples is much smaller and usually suffers from more intra-class acoustic variability. In this paper we propose to integrate a similar approach into training the SED system by using the output of an isolated event classifier as an additional feature. We leverage transfer learning to train the classifier in order to mitigate the effects of low sample-size training dataset. In fact, we use the same pre-trained backbone CNN of the CNN-transformer baseline, and train a single-layer MLP on top of it.

In our proposed system, each segment of the temporal sequence $\boldsymbol{H}$ at the CNN's output is passed to the MLP and the classifier's output scores are appended to the spectro-temporal features of that segment. The overall schematic of the proposed system is illustrated in figure 1 in which we have introduced an *Auxiliary Feature generator* block.

3.1. Event Classifier

Since the goal is to use the classifier for detection and to produce an output probability vector at a specified temporal resolution, we train a point-wise MLP operating on each $\boldsymbol{h}_t$ independently. This means we broadcast the event label to all of it sub-segments across time. Nevertheless, in order to further expand the effective receptive field input to the MLP when classifying $\boldsymbol{h}_t$, we concatenate a set of $2k+1$ dilated neighbouring feature slices centred around each $\boldsymbol{h}_t$ to be passed to the MLP:

$$\boldsymbol{o}_t = MLP(\begin{bmatrix} \boldsymbol{h}_{t-(k.d)} \\ \vdots \\ \boldsymbol{h}_t \\ \vdots \\ \boldsymbol{h}_{t+(k.d)} \end{bmatrix}) \tag{2}$$

with o_t being the classifier's output at time instance t and d the dilation factor. Similar to the SED system's output, in order for the classifier to account for event polyphony and non-target sound events, we train the classifier with a *Binary Cross-Entropy (BCE)* loss on sigmoid-activated outputs over the 10 classes of target events. We refer to the output vector in this case as a *posteriorgram*.

3.2. Auxiliary Feature Generator Block

The above trained MLP classifier is integrated into the SED system in the form of an *Auxiliary Feature Generator* (AFG) block where the trained MLP is moved across the extracted representations $\boldsymbol{h}_t$ of a recording and generates a posteriorgram $\boldsymbol{o}_t$ for each corresponding $\boldsymbol{h}_t$. We further take the logarithm of these probability scores before using them as features in the rest of the system.

Additionally, in order to emphasize the points in time where the classifiers output experience sharp changes (potential event boundaries) we propose to augment each log-probability vector with its first order dynamics $\boldsymbol{d}_t$:

$$\boldsymbol{d}_t = \sum_{n=1}^{N} n \times (\log(\boldsymbol{o}_{t+n}) - \log(\boldsymbol{o}_{t-n})) \tag{3}$$

where N is the width of the window used for the calculation of the derivative. The resulting output feature of the AFG block is the concatenation of each $\log(\boldsymbol{o}_t)$ and $\boldsymbol{d}_t$ yielding the auxiliary feature sequence $\boldsymbol{P}$:

$$\boldsymbol{P} = \{\boldsymbol{p}_1, \ldots, \boldsymbol{p}_{T^e}\} = \{\begin{bmatrix} \log(\boldsymbol{o}_1) \\ \boldsymbol{d}_1 \end{bmatrix}, \ldots, \begin{bmatrix} \log(\boldsymbol{o}_{T^e}) \\ \boldsymbol{d}_{T^e} \end{bmatrix}\} \tag{4}$$

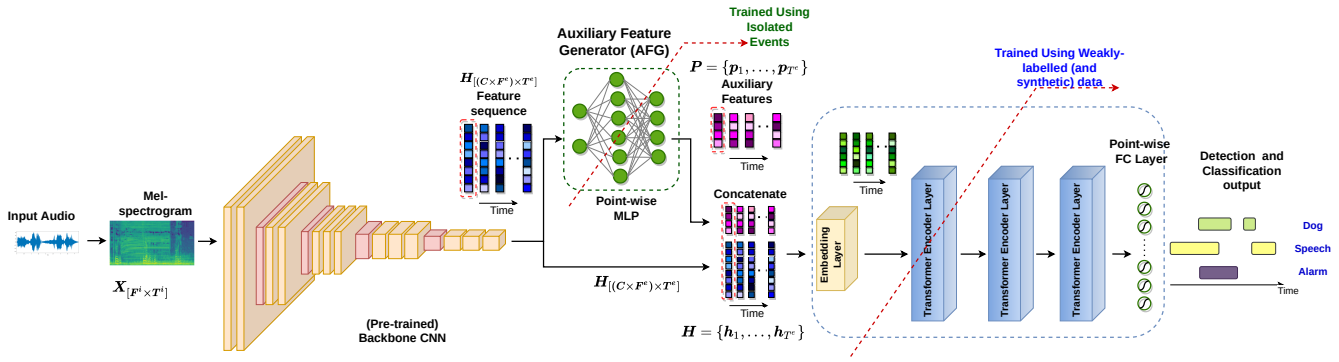


Fig. 1: General schematic of the proposed approach with the AFG block

As illustrated in figure 1, the AFG output *features* $\boldsymbol{p}_t$ are concatenated with the $\boldsymbol{h}_t$ and the sequence of tandem feature slices are batch-normalized before being passed to the embedding layer and the succeeding transformer encoder layers which will be trained using the weakly-labelled (and synthetic) examples.

3.3. Training Procedure

In this work, while training the SED system, we keep the parameters of the classifier (MLP and the backbone CNN) frozen and use the rest of weakly- and strongly-annotated examples only to train the embedding layer and the transformer layers. Given that the CNN is already pre-trained using a large dataset, this should not be a limiting factor on the system's learning capacity. Nevertheless training the entire system end-to-end, which given two tasks and two datasets would be framed as a multitask learning problem, seems more appealing and we intend to address that in our future works.

4. EXPERIMENTS

4.1. Model Parameters

In this work rather than using a shallow CNN as in the original baseline CRNN model of DCASE task 4, we use a deeper Efficientnet-B2 model [12] used in a work by by Gong et al. [13] pre-trained on Audioset [14].

Each 10-second audio clip is resampled at the rate of 16kHz and is converted to log Mel-spectrogram with 25ms window length, 10ms window hop and 128 Mel bands. The resulting 2D array is passed through the backbone CNN of the system.

The Efficientnet-B2 final convolutional layer yields a feature map of $1408 \times 4 \times 33$ dimension as its output. After average pooling along the frequency axis we end up with a temporal sequence of 33 1408-dimensional feature vectors. This sequence is passed through an embedding layer to produce a sequence of 256-dimensional embedding vectors, which in turn is passed through the transformer encoder layers. We use 3 layers of transformer encoder blocks, with 512 as the hidden dimension of the positional feed-forward network and 4 attention heads. In the proposed approach the features in this sequence are concatenated with the AFG's output features, and passed to the embedding layer to be projected to a sequence of the 256-dimensional vectors similar to the baseline model. The same pre-trained CNN is used for training the MLP in the AFG block. We chose width $k = 1$ and dilation $d = 2$ to define the temporal extent of information resulting in a 4224-dimensional input to the MLP. The MLP consists of a 512 unit hidden layer with batch-normalization followed by ReLU activation function. With the 10 output classes of the isolated classifier, the auxiliary features will be 20-dimensional vectors. Since the number and duration of examples per event class are highly variable in the set of isolated examples we adopt a weighted sampling strategy in each batch to train the classifier. For training both the SED system and the classifier we use data augmentation pipeline consisting of mixup [15], frequency and time masking [16] and random additive gaussian noise.

4.2. Datasets

4.2.1. SED Datasets

The datasets that were provided as part of DCASE 2021 task4 challenge (DESED) are listed below.

Weakly-annotated data: 1578 10-second long audio clips of domestic recordings taken from Audioset [14] with verified clip-level labels.

Unlabelled Data: 14412 clips of 10-second clips selected from Audioset. We did not use the unlabelled data in our experiments.

Soundbank of isolated events: 1009 isolated examples of target events taken from FSD50k [17] with clip durations ranging from 55ms to 83.1s. The median

	w. Synth	F1	PSDS1	PSDS2
Baseline	✗	23.21	0.097	0.336
	✓	38.92	0.251	0.512
Ours	✗	35.25	0.183	0.491
	✓	40.20	0.263	0.523

Table 1: A comparative performance evaluation of proposed AFG block on WSSED system

number of examples per class is 62 and the median total duration of examples per class is 310s.

Synthetic Soundscapes: a collection of 10000 synthetically generated soundcapes using the Scaper [18] library using the soundbank of isolated events and a selected set of background sounds from SINS dataset [19].

4.2.2. External Datasets

We also used an external dataset for auxiliary feature generation in our experiments.

ESC-50: a collection of 2000 event clips all of 5s durations across 50 sound event classes each containing 40 examples [20].

4.3. Results

Table 1 summarizes the performance of our proposed approach against the baseline CNN-Transformer system. The results are reported on the provided validation set. The performance is reported using intersection-based F1 macro score and PSDS measures. PSDS1 and PSDS2 are two specific hyperparameter settings of the PSDS measure where the former emphasizes more accurate localization while the focus of latter is on distinguishing classes from one another. In order to evaluate the potential of AFG block to complement or even replace the use of synthetic soundscapes, we report the performance of the baseline system when trained only with weakly-labelled examples alone (w/o synth) and when including the synthetic soundscapes (w. synth).

4.4. AFG, a Gateway for External Datasets

Given the generality of the AFG block, we further investigate the utility of it as a gateway to integrate external annotated datasets. The idea is that nothing restricts the MLP to be trained on the provided soundbank of isolated examples. Even if the target classes of an external dataset does not overlap with the SED task at hand, projecting slices of a recording into a classifier trained on a externally-annotated dataset could provide relevant discriminative features and hence reduce the uncertainty inherent in the weakly-labelled examples.

	w. Synth	F1	PSDS1	PSDS2
Baseline	✗	23.21	0.097	0.336
	✓	38.92	0.251	0.512
AFG: ESC-50	✗	28.33	0.131	0.451
	✓	39.56	0.227	0.536
AFG: ESC-50 + isolated examples	✗	38.22	0.202	0.553
	✓	41.31	0.251	0.574

Table 2: Effect of using external datasets for feature generation on WSSED system's performance

Table 2 lists the results of using the ESC-50 dataset to generate auxiliary features and results are reported for both when used alone and when combined with features from isolated examples. When combined, we concatenate the auxiliary features from both trained classifiers before projecting them into the transformer's embedding dimension. The results are shown for both with and without the use of synthetic dataset in training the SED system. Our results suggest this approach as a rather simple way to distil discriminative information inherent in external datasets into WSSED with minimal effort and without requiring any further annotations with respect to the target task.

5. CONCLUSIONS

In this work we have presented a different way of incorporating a small set of isolated examples into training a WSSED system. Rather than using them to create synthetic soundscapes, we proposed to use the output of a classifier built using them as an auxiliary feature for each sub-segment of an input recording. We have evaluated our approach using DCASE2021 task-4 setup and dataset. Our results suggest that by means of introducing this feature augmentation we could improve the performance of the baseline system with or without synthetic soundscapes. In addition we showed that the auxiliary features could be generated using external datasets to encode extra knowledge into the system and improve the performance.

In our future work we intend to explore other ways of auxiliary feature generation particularly memory-based methods using the advances in deep metric learning and few-shot learning. Moreover, as the auxiliary feature block and the main SED system share the backbone network, the system could be end-to-end optimized through framing it as a multi-task learning problem.

6. REFERENCES

[1] E. Wold, T. Blum, D. Keislar, and J. Wheaten, "Content-based classification, search, and retrieval

of audio," *IEEE MultiMedia*, vol. 3, no. 3, pp. 27–36, 1996.

[2] R. Radhakrishnan, A. Divakaran, and A. Smaragdis, "Audio analysis for surveillance applications," in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics, 2005.*, 2005, pp. 158–161.

[3] Justin Salamon, Juan Pablo Bello, Andrew Farnsworth, Matt Robbins, Sara Keen, Holger Klinck, and Steve Kelling, "Towards the automatic classification of avian flight calls for bioacoustic monitoring," *PLoS One*, vol. 11, no. 11, Nov. 2016.

[4] Marco Cristani, Manuele Bicego, and Vittorio Murino, "Audio-visual event recognition in surveillance video sequences," *IEEE Transactions on Multimedia*, vol. 9, no. 2, pp. 257–267, 2007.

[5] Romain Serizel, Nicolas Turpault, Ankit Shah, and Justin Salamon, "Sound event detection in synthetic domestic environments," in *Proc. ICASSP*, 2020, pp. 86–90.

[6] Nicolas Turpault and Romain Serizel, "Training sound event detection on a heterogeneous dataset," in *DCASE Workshop*, 2020.

[7] Gethin Williams and Daniel P. W. Ellis, "Speech/music discrimination based on posterior probability features," in *EUROSPEECH*, 1999.

[8] Guoguo Chen, Carolina Parada, and Georg Heigold, "Small-footprint keyword spotting using deep neural networks," in *Proc. ICASSP*, 2014, pp. 4087–4091.

[9] Koichi Miyazaki, Tatsuya Komatsu, Tomoki Hayashi, Shinji Watanabe, Tomoki Toda, and Kazuya Takeda, "Weakly-supervised sound event detection with self-attention," in *Proc. ICASSP*, 2020, pp. 66–70.

[10] Yun Wang, Juncheng Li, and Florian Metze, "A comparison of five multiple instance learning pooling functions for sound event detection with weak labeling," in *Proc. ICASSP*, 2019, pp. 31–35.

[11] Tara N. Sainath and Carolina Parada, "Convolutional neural networks for small-footprint keyword spotting," in *Proc. Interspeech 2015*, 2015, pp. 1478–1482.

[12] Mingxing Tan and Quoc Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *International conference on machine learning*. PMLR, 2019, pp. 6105–6114.

[13] Yuan Gong, Yu-An Chung, and James Glass, "PSLA: Improving audio tagging with pretraining, sampling, labeling, and aggregation," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 3292–3306, 2021.

[14] Jort F. Gemmeke, Daniel P. W. Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R. Channing Moore, Manoj Plakal, and Marvin Ritter, "Audio set: An ontology and human-labeled dataset for audio events," in *Proc. IEEE ICASSP 2017*, New Orleans, LA, 2017.

[15] Yuji Tokozume, Yoshitaka Ushiku, and Tatsuya Harada, "Learning from between-class examples for deep sound recognition," in *International Conference on Learning Representations*, 2018.

[16] Daniel S. Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D. Cubuk, and Quoc V. Le, "SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition," in *Proc. Interspeech 2019*, 2019, pp. 2613–2617.

[17] Eduardo Fonseca, Xavier Favory, Jordi Pons, Frederic Font, and Xavier Serra, "Fsd50k: An open dataset of human-labeled sound events," *IEEE/ACM Trans. Audio, Speech and Lang. Proc.*, vol. 30, pp. 829–852, jan 2022.

[18] Justin Salamon, Duncan MacConnell, Mark Cartwright, Peter Li, and Juan Pablo Bello, "Scaper: A library for soundscape synthesis and augmentation," in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, 2017, pp. 344–348.

[19] Gert Dekkers, Steven Lauwereins, Bart Thoen, Mulu Weldegebreal Adhana, Henk Brouckxon, Bertold Van den Bergh, Toon van Waterschoot, Bart Vanrumste, Marian Verhelst, and Peter Karsmakers, "The SINS database for detection of daily activities in a home environment using an acoustic sensor network," 2017, pp. 1–5, DCASE Workshop.

[20] Karol J. Piczak, "ESC: Dataset for Environmental Sound Classification," in *Proceedings of the 23rd Annual ACM Conference on Multimedia*. pp. 1015–1018, ACM Press.

IMPACT OF TEMPORAL RESOLUTION ON CONVOLUTIONAL RECURRENT NETWORKS FOR AUDIO TAGGING AND SOUND EVENT DETECTION

Wim Boes, Hugo Van hamme

ESAT, KU Leuven, Belgium

ABSTRACT

Many state-of-the-art systems for audio tagging and sound event detection employ convolutional recurrent neural architectures. Typically, they are trained in a mean teacher setting to deal with the heterogeneous annotation of the available data.

In this work, we present a thorough analysis of how changing the temporal resolution of these convolutional recurrent neural networks — which can be done by simply adapting their pooling operations — impacts their performance. By using a variety of evaluation metrics, we investigate the effects of adapting this design parameter under several sound recognition scenarios involving different needs in terms of temporal localization.

Index Terms— sound recognition, audio tagging, sound event detection, temporal resolution

1. INTRODUCTION

Recently, the popularity of research into audio-related tasks has surged because of, among other reasons, multiple versions of the Detection and Classification of Acoustic Scenes and Events (DCASE) challenge [1, 2, 3, 4, 5, 6]. The latest editions encompassed six categories, each dealing with a distinct sound recognition problem. Subtask number 4 [7] consistently covered sound event detection — joint classification and temporal localization of auditory events — in domestic environments. Submitted systems were ranked utilizing measures which represent scenarios involving variable requirements with regard to the estimation of time boundaries.

The baseline [7] supplied for the fourth problem of the DCASE 2021 and 2022 challenges, a convolutional recurrent neural network trained using the mean teacher principle [8], was based on the second-ranking system [9] of task 4 of DCASE 2019. Its output temporal resolution was predefined and fixed.

Our submission [10] for the fourth subtask of the DCASE 2021 challenge [7] demonstrated that simply adapting the amount of pooling in this network, which is directly linked to its temporal resolution, can significantly impact its performance.

In this work, we provide a more thorough and complete analysis of this interesting finding. We substantially supplement the discussion in multiple ways, as outlined in the two paragraphs below.

Firstly, we examine more than the specific evaluation scenarios prescribed in task 4 of the DCASE 2021 and 2022 challenges. Particularly, we also investigate what happens when the temporal resolutions of convolutional recurrent networks are adapted in the context of audio tagging. In this situation, the goal of the models is to perform clip-level auditory event classification. Unlike for sound event detection, temporal localization is not required in this case.

This work was supported by a PhD Fellowship of Research Foundation Flanders (FWO-Vlaanderen) and the Flemish Government under "Onderzoeksprogramma AI Vlaanderen".

Secondly, for sound event detection, we inspect what happens using multiple types of measures. More specifically, in addition to the intersection-based scores employed in subtask 4 of the DCASE 2021 and 2022 challenges, we also utilize segment-based and event-based metrics, which are commonly used in this subdomain of machine learning as well. This is further elaborated upon in Section 4.

To this end, the following approach is taken: We start with a suitable baseline, which forms the basis for many state-of-the-art architectures for both audio tagging and sound event detection, such as [11], [12] and [13]. We change the temporal resolution of this model by adapting its pooling operations. We then analyze the results obtained by this modified system in a variety of experimental configurations, representing different evaluation scenarios.

In Section 2, we expand upon the baseline system. Afterwards, in Section 3, we describe how the pooling operations of the considered architecture can be adapted to modify the temporal resolution of the model. Then, in Section 4, we elaborate upon the experimental setup. Next, in Section 5, we analyze the results of the performed experiments, and finally, we draw a conclusion in Section 6.

2. BASELINE

In this section, the baseline system, which is nearly the same as in [14], is elaborated upon. It is a slightly adapted version of the baseline supplied for task 4 of the DCASE 2021 and 2022 challenges [7].

2.1. Architecture

A schematic visualization of the baseline model is given in Figure 1.

The input to the baseline is a spectral map of an audio recording. The amount of frequency bins is predefined and set to 128.

The first component of the architecture is a convolutional neural network (CNN) made up of seven blocks. Each of those consists of the following five layers: a convolutional layer, a batch normalization layer [15], a ReLU activation layer, a dropout layer [16] with a dropout rate of 33%, and lastly, an average pooling layer.

All convolutional layers use a square kernel of size 3 and utilize a stride of 1. For the first three blocks, the number of output channels of the convolutional operations is equal to 16, 32 and 64 respectively. For the last four blocks, this number is equal to 128.

The hyperparameters of the pooling operations are given per model block in Table 1. The first and second numbers of each tuple are connected to the time and frequency axes respectively.

After the last block, the frequency-related dimension of the spectral (auditory) input map has been brought down to one and the corresponding axis can therefore be squeezed, i.e., removed.

Afterwards, a bidirectional gated recurrent (BiGRU) unit, consisting of two layers with hidden sizes equal to 128, is used to model potential temporal relationships in the auditory data.

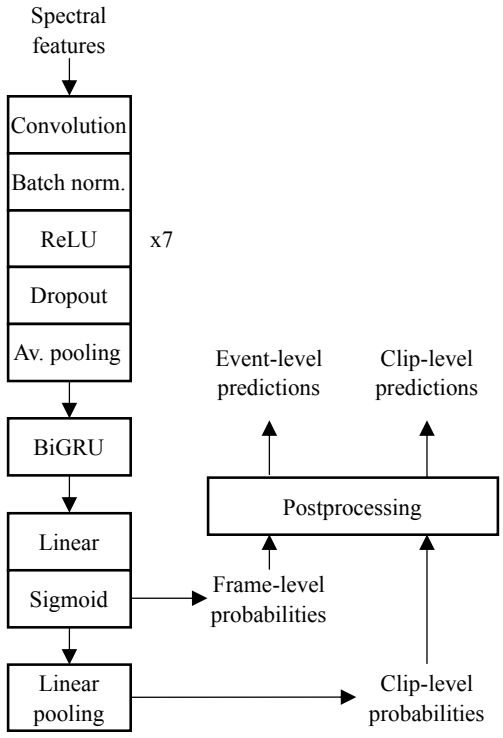


Figure 1: Schematic representation of the baseline

The output of this recurrent layer is followed by a linear projection and application of the sigmoid function to obtain multi-label frame-level probabilities. These values indicate per temporal frame which sound categories are active, and can be postprocessed to obtain event-level predictions of sounds, which are relevant for sound event detection, as explained in the next subsection.

Finally, the frame-level sound probabilities are aggregated to get clip-level auditory event probabilities, which are relevant for the task of audio tagging, by performing linear (softmax) pooling [17].

2.2. Postprocessing

The clip- and frame-level probabilities are converted into binary values by enforcing a fixed threshold. The frame-level decisions are also passed through a smoothing median filter with a window of about 500 ms. Preliminary experiments showed that more complicated postprocessing (e.g., class-wise optimization on validation data) is unnecessary as it provides relatively insignificant benefits.

The binary clip-level decisions can readily be used to perform audio tagging. On the contrary, to produce outputs suitable for sound event detection, an extra step has to be taken: The frame-level decisions are converted into event-level predictions by performing a merging operation with a maximum gap tolerance of 200 ms.

2.3. Mean teacher training

As expanded upon in Section 4, the training data employed in this research project is heterogeneously annotated: Some of the samples include clip-level or weak labels, some come with event-level or strong labels, and the rest is unlabeled. To deal with this difficulty, the mean teacher training principle [8] is applied.

Table 1: Kernel sizes and strides of pooling layers in baseline CNN

Block	**Kernel size = stride**
0-1	(2, 2)
2-3-4-5-6	(1, 2)

In the mean teacher training framework, two models called the student and the teacher are utilized. They share the same architecture, but their parameters are updated completely differently.

The student system is trained in a regular fashion: A differentiable loss function is optimized by an optimization algorithm such as Adam [18]. This is not the case for the teacher counterpart: The weights of this model are computed as the exponential moving average of the student parameters with a multiplicative decay factor of 0.999 per training iteration, also explaining the name of the method.

The loss used to train the student consists of four terms: The first two are clip-level and frame-level binary cross entropy functions, which are only calculated for the weakly and strongly labeled data samples respectively. The remaining two components are mean-squared error consistency costs between the clip-level and frame-level output probabilities of the student and teacher models, which can be computed for all examples, including the unlabeled ones. The classification and consistency terms are summed with weights equal to 1 and 2 respectively to obtain the final objective.

3. ADAPTATION OF TEMPORAL RESOLUTION

The goal of this project is to investigate how modifying the output temporal resolution of the considered convolutional recurrent neural network affects its sound recognition performance under different circumstances. In this section, we describe which parts of this model were changed to achieve these specific adaptations.

In the baseline model outlined in Section 2, the pooling layers in the first and second blocks of the CNN halve input feature maps along the temporal dimension, while the others do not change anything in this regard. This results in a total pooling factor of 4.

As explained in more detail in Section 4, most of the audio recordings used in this project have a duration of 10 seconds. They are fed to the baseline convolutional recurrent neural network as spectral feature maps consisting of 608 time frames. This model applies a temporal reduction factor of 4, and hence, the frame-level probabilities contain 152 values per sample. This comes down to a temporal resolution of about one class prediction vector per 65 ms.

We create adaptations of the baseline convolutional recurrent network in the following way: Instead of only allowing the first two pooling layers to apply a reduction in the number of temporal frames, we create model versions of which the first x pooling layers perform this halving operation, with x ranging from 1 up to 6. This results in systems with time reduction factors of 2, 4, 8, 16 and 32 respectively, which are equivalent to temporal resolutions of about one frame-level prediction bin per 32.5, 65, 130, 260 and 520 ms.

Important to note is that changing the amount of pooling in the considered convolutional recurrent neural network does not change the number of trainable parameters, and thus, its modeling capacity.

4. EXPERIMENTAL SETUP

In this section, we provide full details on the setup used during the experiments, of which the results are analyzed in Section 5.

4.1. Data

The data set for problem 4 of DCASE 2021 [7] is a multi-label collection consisting of audio recordings with a maximum length of 10 seconds. There are 10 possible sound event categories, related to domestic environments, which are not mutually exclusive.

The partition used to train models consists of three subsets:

- 1578 real samples with weak (clip-level) annotation
- 10000 synthetic samples with strong (event-level) annotation
- 14412 real samples without annotation

For evaluation purposes, two partitions were available: the so-called validation and public evaluation sets, holding 1168 and 692 real recordings respectively. Both subsets include strong or event-level labels of the active environmental auditory events.

4.2. Preprocessing

To get spectral maps of the available audio recordings, which are used as input features for the considered convolutional recurrent nets as explained in Section 2, we resampled all clips to 22050 Hz and performed peak amplitude normalization. Then, log mel spectrograms with 128 frequency bins were extracted using a Hamming window with a size of 2048 samples and a hop length of 363 samples. Lastly, per-frequency bin standardization was carried out.

As mentioned before, for a 10-second audio clip, these steps resulted in a spectral feature map consisting of 608 temporal frames.

4.3. Data augmentation

In order to avoid the risk of overfitting, we employed data augmentation during the training of the considered models. In particular, we used mixup [19], which comes down to creating extra learning examples (and associated labels) by linearly interpolating the original samples. We employed this method with a probability of 50% of applying it. The mixing ratios used in this algorithm were randomly sampled from a beta distribution with shape parameters set to 0.2.

Unlike for our related submission [10] for task 4 of the DCASE 2021 challenge [7], we did not employ time and frequency masking [20] in this project. This choice was based on two observations made during initial experiments: Firstly, the best hyperparameters for these techniques seemed to depend on the situation, e.g., used metric. Secondly, these methods only led to minimal improvements. As the focus of this work is on analysis rather than trying to get optimized performance, they were excluded for the sake of clarity.

4.4. Training and evaluation

All models were trained and evaluated using PyTorch [21].

4.4.1. Training

All models were trained for 200 epochs. Per epoch, 250 batches of 48 samples were given to the networks. Each batch contained 12 weakly labeled, 12 strongly labeled and 24 unlabeled examples.

Adam [18] was employed to train the weights of the student models. Learning rates were ramped up exponentially from 0 to 0.001 for the first 12500 optimization step. Thereafter, they decayed multiplicatively at a rate of 0.99995 per training iteration.

4.4.2. Evaluation

For audio tagging, we used the (micro-averaged) clip-based F1 measure [22], which was also utilized in task 4 of the DCASE 2017 challenge [23]. This score was computed for a single operating point, namely, the situation in which probabilities were converted into binary decisions by enforcing a 50% threshold.

For sound event detection, we employed multiple types of metrics. To start with, we utilized the intersection-based measures used for ranking in task 4 of the DCASE 2021 and 2022 challenges [7]. More specifically, we used two polyphonic sound event detection scores [24] representing distinct evaluation scenarios, denoted as PSDS 1 and 2. The former imposes strict requirements on the temporal localization accuracy, the latter is more lenient in this regard.

The hyperparameters utilized for calculating these PSDS measures are summarized in Table 2. These scores were computed using 50 operating points, in which thresholds linearly distributed from 0.01 to 0.99 were used to convert probabilities into binary decisions.

Details on the procedure for calculating PSDS scores and a discussion on the hyperparameters can be found in [24].

Table 2: PSDS hyperparameters

Hyperparameter	**PSDS 1**	**PSDS 2**
Detection tolerance criterion	0.7	0.1
Ground truth intersection criterion	0.7	0.1
Cross-trigger tolerance criterion	N/A	0.3
Cost of class instability	1	1
Cost of cross-triggers	0	0.5
Maximum false positive rate	100	100

Furthermore, we used the segment-based (micro-averaged) F1 score based on chunks of 1 s [22] to gauge the performance of the considered models. This metric was also employed in task 4 of the DCASE 2017 challenge [23]. Because of the long segment length, this measure quantifies systems in terms of their ability to perform more coarse-grained sound event detection, not unlike PSDS 2.

Lastly, we also employed the (macro-averaged) event-based F1 score with tolerances of 200 ms for onsets and 20% of the audio event lengths (up to a max of 200 ms) for offsets [22]. This measure was used in task 4 of the DCASE 2018, 2019 and 2020 challenges [7, 25]. This metric quantifies models in terms of their ability to perform fine-grained sound event detection, similar to PSDS 1.

These segment-based and event-based scores were computed for a single operating point, in which a threshold of 0.5 was enforced to convert frame-level probabilities into binary decisions.

All measures were computed using the clip-level or frame-level probabilities of the student models after the last training epoch.

5. EXPERIMENTAL RESULTS

In this section, we analyze the results obtained by the convolutional recurrent neural systems with varying output resolutions, as explained previously. We report the metrics for audio tagging and sound event detection elaborated upon in Section 4 after applying the following method to significantly diminish the variability of the results: For each experimental configuration, we train 20 models with independent initializations and average the scores they achieve on the public evaluation partition of the employed data set.

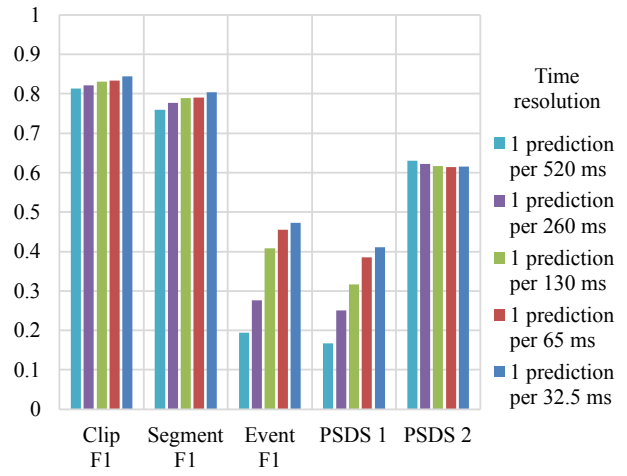


Figure 2: Results on public evaluation subset

The scores achieved on the public evaluation set are summarized in Figure 2. Precise numerical values are not essential to the interpretation below and are not disclosed due to a lack of space.

Only one of the metrics appears to display a (very slight) negative correlation with the temporal output resolution of the considered models, namely, PSDS 2. Intuitively, this behavior is not unexpected as this intersection-based score was designed to gauge relatively coarse-grained sound event detection performance.

All other measures correlate positively with respect to the time resolution. For the event-based F1 score and PSDS 1, the scores demanding precise estimations of the boundaries of sounds, this seems logical: Naturally, if the length associated with an output prediction bin is too long, this becomes more challenging or even impossible.

However, this positive relationship also appears to hold for the clip-based and segment-based F1 metrics, which inherently require much less accurate temporal differentiation — in the former case, this is not even needed at all. This counterintuitive result is all the more unanticipated given the trend of PSDS 2: Apparently, there is a certain discrepancy in the reaction of the examined types of measures for audio tagging and coarse-grained sound event detection to the temporal output resolution of the considered models.

The sensitivity of the performance of convolutional recurrent neural networks to their output resolutions depends on the criterion. In particular, scores designed for more fine-grained sound event detection show a high responsiveness in this regard: PSDS 1 and the event-based F1 measure drop sharply as the temporal resolution of the models at hand decreases. On the contrary, the clip- and segment-based F1 metrics as well as PSDS 2, used to gauge performance with regard to audio tagging and coarse-grained sound event detection, remain more stable as this hyperparameter varies.

It is also possible to study how the performance and sensitivity of the models differs across sound categories by inspecting class-wise scores. For F1-based measures, these can easily be obtained. For the PSDS metrics, they can also be computed, but unlike for their aggregated counterparts, the cost of class instability (see Section 4) must be disregarded. Exact numerical results are not crucial to the following analysis and are not included due to a lack of space.

In nearly all instances, the class-wise scores correlate to the temporal resolution in the same direction (positively/negatively) as their global versions. Nevertheless, the absolute performance and responsiveness strikingly vary across metrics and sound categories.

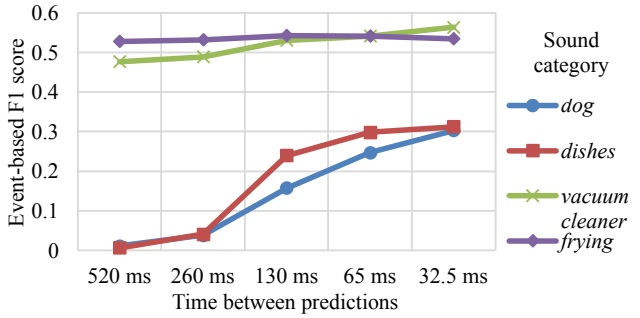


Figure 3: Subset of class-wise event-based F1 scores

For all of the measures, there is a substantial link between the scores and the examined kind of sound. As an example, the best-performing category (*speech*) consistently outperforms the worst (*running water*) by a large margin. This behavior can be explained by multiple factors, e.g., qualitative differences between types of audio events and imbalance within the data used for training.

As is the case for the aggregated scores, the class-wise sensitivities are only strongly pronounced for fine-grained sound event detection measures. In these cases, they also greatly depend on the considered audio category. This is exemplified for the event-based F1 metric in Figure 3. For events which are short and localized (e.g., *dog*, *dishes*), reducing the temporal resolution leads to severe deterioration in terms of performance. For more long-lived sounds (e.g., *vacuum cleaner*), this dependency is much less outspoken. In the most extreme case of *frying*, there is even no correlation at all.

6. CONCLUSION

Numerous state-of-the-art sound recognition models are based on convolutional recurrent neural architectures. They are usually optimized in a mean teacher framework to deal with the heterogeneous labeling of the supplied data samples. In this work, we provided deeper insight into how changing the temporal resolution of such nets impacts their performance. We analyzed the effect of modifying pooling operations on a multitude of metrics, designed for distinct audio tagging and sound event detection scenarios.

The experiments showed that the performance of the considered models is highly susceptible to changes in terms of their pooling operations. More specifically, metrics designed for fine-grained sound event detection showed a strong, positive relation with respect to the temporal output resolutions of the systems at hand. For sound recognition measures involving less focus on temporal differentiation (among others, F1 score for audio tagging), the direction of the correlation was mixed and contingent on the scoring method.

The responsiveness appeared to depend on the considered evaluation scenario. Generally, measures devised for fine-grained sound event detection displayed a much higher reactivity than metrics built for audio tagging and coarse-grained sound event detection.

A class-wise study showed that for the latter, the sensitivity to temporal resolution was low across all sound types. Conversely, for fine-grained sound event detection scores, the reactivity was decidedly stronger for shorter than for longer-lasting events.

7. REFERENCES

[1] A. Mesaros, T. Heittola, E. Benetos, P. Foster, M. Lagrange, T. Virtanen, and M. D. Plumbley, "Detection and Classification of Acoustic Scenes and Events: Outcome of the DCASE 2016 Challenge," *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, vol. 26, no. 2, pp. 379–393, 2018.

[2] T. Virtanen, A. Mesaros, T. Heittola, A. Diment, E. Vincent, E. Benetos, and B. M. Elizalde, *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2017 Workshop (DCASE2017)*, 2017.

[3] M. D. Plumbley, C. Kroos, J. P. Bello, G. Richard, D. P. Ellis, and A. Mesaros, *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*, 2018.

[4] M. Mandel, J. Salamon, and D. P. Ellis, *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2019)*, 2019.

[5] N. Ono, N. Harada, Y. Kawaguchi, A. Mesaros, K. Imoto, Y. Koizumi, and T. Komatsu, *Proceedings of the 5th Workshop on Detection and Classification of Acoustic Scenes and Events (DCASE 2020)*, 2020.

[6] F. Font, A. Mesaros, D. P. Ellis, E. Fonseca, M. Fuentes, and B. Elizalde, *Proceedings of the 6th Workshop on Detection and Classification of Acoustic Scenes and Events (DCASE 2021)*, 2021.

[7] N. Turpault, R. Serizel, A. Parag Shah, and J. Salamon, "Sound event detection in domestic environments with weakly labeled data and soundscape synthesis," in *Workshop on Detection and Classification of Acoustic Scenes and Events*, 2019.

[8] A. Tarvainen and H. Valpola, "Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results," in *Advances in Neural Information Processing Systems*, 2017, pp. 1195–1204.

[9] L. Delphin-Poulat and C. Plapous, "Mean teacher with data augmentation for dcase 2019 task 4," Orange Labs Lannion, Tech. Rep., 2019.

[10] W. Boes and H. Van hamme, "Optimizing temporal resolution of convolutional recurrent neural networks for sound event detection," ESAT, Tech. Rep., 2021.

[11] N. K. Kim and H. K. Kim, "Self-training with noisy student model and semi-supervised loss function for DCASE 2021 challenge task 4," Gwangju Institute of Science and Technology, Tech. Rep., 2021.

[12] X. Zheng, H. Chen, and Y. Song, "Zheng USTC team's submission for DCASE2021 TASK4 - semi-supervised sound event detection," University of Science and Technology of China, Tech. Rep., 2021.

[13] J. Ebbers and R. Haeb-Umbach, "Pre-Training And Self-Training For Sound Event Detection In Domestic Environments," Paderborn University, Tech. Rep., 2022.

[14] W. Boes and H. Van hamme, "Audiovisual transfer learning for audio tagging and sound event detection," *arXiv preprint arXiv:2106.05408*, 2021.

[15] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," in *Proceedings of ICML*, 2015, pp. 448–456.

[16] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[17] Y. Wang, J. Li, and F. Metze, "A comparison of five multiple instance learning pooling functions for sound event detection with weak labeling," in *2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 31–35.

[18] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[19] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," *arXiv preprint arXiv:1710.09412*, 2017.

[20] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "SpecAugment: A simple data augmentation method for automatic speech recognition," *arXiv preprint arXiv:1904.08779*, 2019.

[21] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "PyTorch: An Imperative Style, High-Performance Deep Learning Library," in *Advances in Neural Information Processing Systems*, 2019, pp. 8024–8035.

[22] A. Mesaros, T. Heittola, and T. Virtanen, "Metrics for Polyphonic Sound Event Detection," *Applied Sciences*, vol. 6, no. 6, p. 162, 2016.

[23] A. Mesaros, T. Heittola, A. Diment, B. Elizalde, A. Shah, E. Vincent, B. Raj, and T. Virtanen, "DCASE 2017 challenge setup: Tasks, datasets and baseline system," in *DCASE 2017-Workshop on Detection and Classification of Acoustic Scenes and Events*, 2017.

[24] Ç. Bilen, G. Ferroni, F. Tuveri, J. Azcarreta, and S. Krstulović, "A framework for the robust evaluation of sound event detection," in *2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 61–65.

[25] Serizel, Romain and Turpault, Nicolas and Eghbal-Zadeh, Hamid and Shah, Ankit Parag, "Large-scale weakly labeled semi-supervised sound event detection in domestic environments," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*, 2018.

CONFIDENCE REGULARIZED ENTROPY FOR POLYPHONIC SOUND EVENT DETECTION

Won-Gook Choi

Department of Electronic Engineering
Hanyang University, Seoul, Republic of Korea
onlyworld94@hanyang.ac.kr

*Joon-Hyuk Chang**

Department of Electronic Engineering
Hanyang University, Seoul, Republic of Korea
jchang@hanyang.ac.kr

ABSTRACT

One of the main issues of polyphonic sound event detection (PSED) is the class imbalance problem caused by the proportions of active and inactive frames. Since the target sounds occasionally appear, binary cross-entropy makes the model mainly fit on inactive frames. This paper introduces an effective objective function, confidence regularized entropy, which regularizes the confidence level to prevent overfitting of the dominant classes. The proposed method exhibits less overfitted samples and better detection performance than the binary cross-entropy. Also, we compare our method with the other objective function, the asymmetric focal loss also designed to solve the class imbalance problem in PSED. The two objective functions show different system characteristics. From an end-user perspective, we suggest choosing a proper objective function for the purposes.

***Index Terms*—** Polyphonic sound event detection, class imbalance problem

1. INTRODUCTION

Polyphonic sound event detection (PSED) is one of the acoustic classification and detection tasks that detects the target sound and timestamps in an audio signal. For many years, PSED has had following several challenges:

- Difficult to gather strongly labeled recordings.
- The subjectivity problem of manual labeling.
- Hard to find an analytic model that can cover the various sound patterns.
- The class imbalance problem due to the proportion of active and inactive frames.

The first problem has been solved with two approaches: semi-supervised learning approaches using both labeled and unlabeled data and training with synthetic audio mixed background noise and target sounds. The second problem could be relieved by choosing the metric when comparing the system with others [1]. And the third problem has been solved by deep neural netoworks using improved convolutional neural networks (CNNs), Transformer, attention mechanisms, etc [2].

This study focuses on the last problem of class imbalance. Most inactive frames (background sound) dominate an audio clip, so the problem arises when detecting a target sound frame by frame (Fig. 1). This phenomenon is not a problem presented only in PSED task; the image object detection also has suffered from the background-foreground class imbalance [3]. For the image object detection, a solution is the focal loss that controls the weight parameter of cross-entropy so that train the model well for the target object, but vice versa for the background images. Motivated by the focal loss, the previous study in PSED proposed asymmetric focal loss (AFL) [4] that could control the focal weights of entropies for the inactive and active terms, respectively. AFL successfully controlled the imbalance problem, but an adverse effect arose: the system detected repetitive impulsive sounds as a long-duration sound.

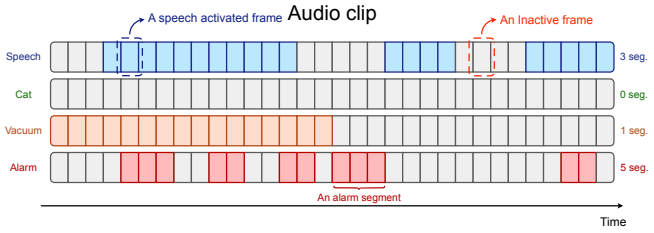


Figure 1: Simple illustration of an audio clip with target sounds.

In this study, we propose confidence regularized entropy (CRE), which set the confidence threshold to the binary cross entropy (BCE). When calculating the BCE, samples are eliminated for the backpropagation during training steps if the detected results are over the threshold. The proposed method keeps the samples less overfitted, especially for the inactive frames. Compared to the AFL, the proposed entropy resulted in a system that can detect the onsets and offsets of target sounds well. Both CRE and AFL relieved the class imbalance problem for PSED. However, they showed a different system characteristic: the CRE-based system was advantageous in detecting a target event's precise localization on frames, whereas the AFL-based system showed strength in detecting whether a target sound appeared. The details will be discussed in Section 5.2.

2. CLASS IMBALANCE PROBLEM WITH PSED

A class imbalance problem is one of the considerations for building and training a neural network. If the class imbalance problem remains unsolved, the model could remain ungeneralized [3]. When collecting data from real world, the target sound would appear intermittently rather than often; thus, one of the factors that cause class imbalance is the imbalance between the number of active and inactive frames [4] in dealing with the PSED tasks. Additionally, the imbalance among the target sounds could appear since each event's duration is entirely different, and the amount of recorded sound is

*corresponding author.

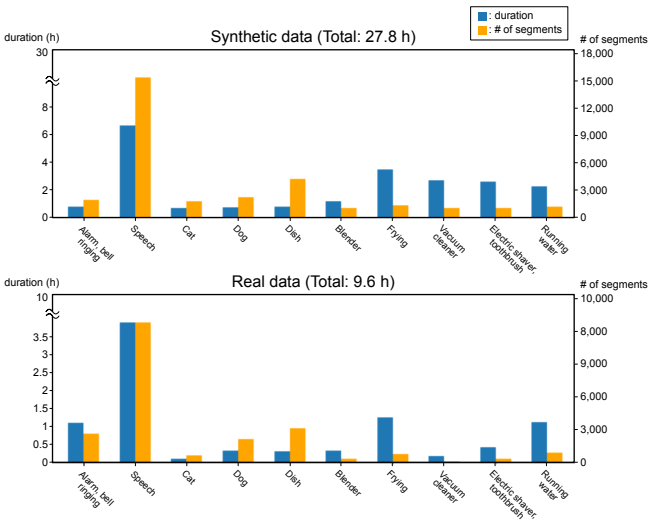


Figure 2: Bar graphs representing the duration of active sounds (blue) and the number of active segments (yellow). Considering the total audio length, the proportion of inactive frames is large.

also diverse according to the datasets.

The total duration and number of segments for each target sound composing the domestic environment sound event detection (DESED) dataset are shown in Fig. 2. In both synthetic and real data, inactive frames have large proportions and, all the events except speech have very low proportions.

Imoto *et al.* proposed asymmetric focal loss (AFL) [4] to control the entropies of active and inactive frames. The entropy between a ground truth $y_{n,c}$ and a model output $\hat{y}_{n,c}$ is described:

$$AFL = -\sum_{n,c=0}^{N,C} \{\underbrace{(1-\hat{y}_{nc})^{\gamma}\, y_{nc} \log(\hat{y}_{nc})}_{\text{Active term}} + \underbrace{(\hat{y}_{nc})^{\zeta}\, (1-y_{nc}) \log(1-\hat{y}_{nc})}_{\text{Inactive term}}\} \quad (1)$$

where γ and ζ denote the parameters to control the entropies of active and inactive frames in each, and N and C denote the number of frames and target sounds, respectively. If γ and ζ are set to 0, the entropy is same as the binary cross entropy, and the higher the values, the less focal. Imoto *et al.* set γ and ζ to 0.0625 and 1, respectively, which means that the objective function focuses more on the active frames.

3. CONFIDENCE REGULARIZED ENTROPY

Suppose that there are lot proportion of speech-activated and inactive frames among the datum. If then, the inactive points are converged earlier than the other target sounds (e.g., cat, vacuum cleaner, etc. in Fig. 2). Even if the training epoch is processed enough, the inactive points are still converging more closely to one or zero, whereas the network is less optimized for the other target sounds. To concentrate on training the network for the false positive and false negative data, we propose the confidence regularized entropy (CRE) that can regularize confidences so that they could not converge beyond the threshold.

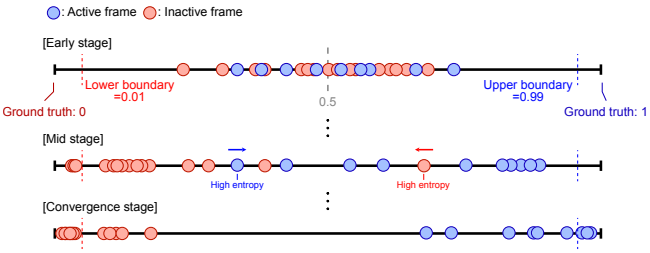


Figure 3: Training scenario when CRE is used for objective function.

$$CRE = -\frac{1}{NC}\sum_{n,c=0}^{N,C} \mathbb{I}_{|\hat{y}_{nc}-y_{nc}|>\gamma}(\hat{y}_{nc}) \cdot \{y_{nc} \cdot \log\hat{y}_{nc} + (1-y_{nc}) \cdot \log(1-\hat{y}_{nc})\}, \quad (2)$$

where $\mathbb{I}(\cdot)$ denotes an indicator function. We set γ to 0.01; the frames are excluded on each optimizing step, if those of confidence are either over the 0.99 or under 0.01. Generally, the mixup augmentation [5] is widely used for training the PSED network, and Eq. (2) also can be used whether the mixup is applied. If the mixup is used, confidences that are too close to the mixed labels are excluded during the training. In Section 5.1, the experimental results will demonstrate that a system that applied both CRE and mixup surpasses the system without either of them.

4. EXPERIMENTS

4.1. Dataset

To validate our proposed method, we used DESED database[1] [6]. There were ten sound events that could occur in domestic environments. For the training set, there were 10,000 synthetic clips with strong annotations, 3,470 recorded clips with strong labels coming from the Audioset [7], 1,578 recorded clips with weak labels, and 14,412 unlabeled-recorded clips. For the evaluation set, there were 1,168 recorded clips. Each clip had a 10 s duration and was provided either 16 kHz or 44.1 kHz and single or dual channel. All clips were down-mixed to 16 kHz and extracted to log-mel spectrograms. For the details, window size and shift size were used 2048 and 255 samples, respectively, and 128 mel-filter banks.

4.2. CNN networks

The CNN architecture for the experiments is shown in Fig. 5. The group size of convolutional layer was 4, and output channel sizes were 32, 64, 128, 256, 256, 256 and 128, respectively. To reduce a temporal size of feature map without temporal pooling, we stacked frames in 4 layers. Also, we designed the axis-wise attention module (AWAM) inspired by parallel temporal-spectral attention [8] to improve the baseline model [9]. AWAM is a module that calculates the sigmoid-based score for each axis and adds to the input feature map, and it was adopted after the 2nd, 4th, and 6th convolutional blocks. The detail of AWAM architecture is shown in Figs. 4. The RNN network was same to the baseline CRNN introduced in [9].

[1] Strong labeled real recordings were newly released in DCASE 2022 challenge task 4. https://github.com/DCASE-REPO/DESED_task/tree/master/recipes/dcase2022_task4_baseline

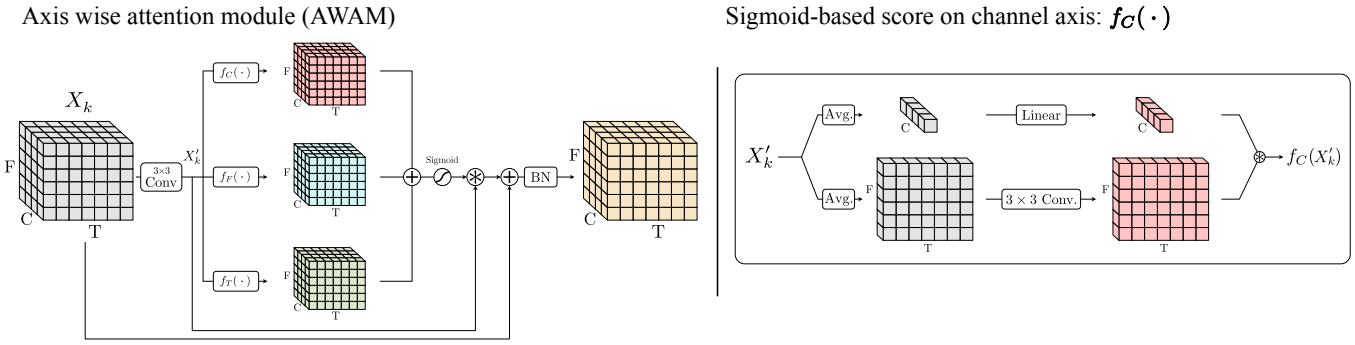


Figure 4: Axis wise attention module. X_k denotes the output of the k-th convolutional block in Fig. 5. The architecture of $f_F(\cdot)$ and $f_T(\cdot)$ are same to $f_C(\cdot)$, but are performed on frequency and time axis, respectively.

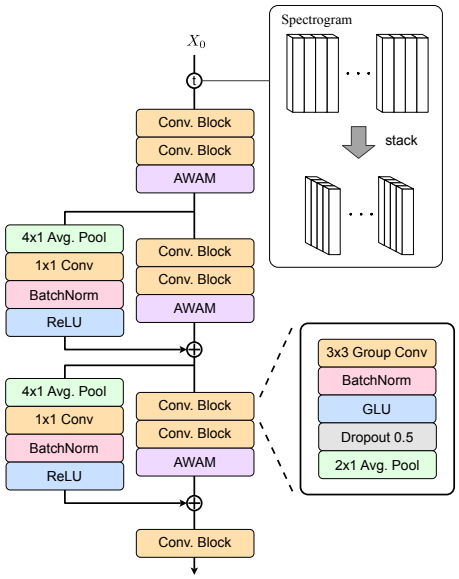


Figure 5: Block diagram of CNN architecture.

4.3. Experimental setup

We adopted the mean teacher [9, 10], one of the semi-supervised learning strategies to train the detection model using the unlabeled data. A minibatch consists of synthetic, strong, weak, and unlabeled-recorded clips with batch sizes of 8, 8, 4, and 40 each. All networks were optimized with the AdamW [11] optimizer and the cosine-annealing learning rate scheduler for 50 epochs after warming up the first 50 epochs from 0 to 0.001. Also, we set the weight decay and dropout to 0.001 and 0.5, respectively. We used event-based f1 score [12] and polyphonic sound detection score (PSDS) [13] for the evaluation metrics[2].

[2]The specific parameters settings for all metrics were same to the recent DCASE challenge.
https://dcase.community/challenge2022/task-sound-event-detection-in-domestic-environments

5. RESULTS AND DISCUSSION

5.1. Effect of confidence regularization

The experimental results according to the objective functions are compared in Table 1. If the other conditions are same except objective function, the systems built with CRE showed great performances under the event-f1 and PSDS1 metrics (CRE $>$ BCE $>$ AFL). Whereas, the systems built with AFL showed better performances under the PSDS2 metric (AFL $>$ BCE $>$ CRE), and the system with BCE showed medium performances for all metrics. Also, the proposed confidence regularization method was applicable with the mixup augmentation. The results demonstrate that if the detection performances of a mixup-applied system with BCE improved more than the system without the mixup, the mixup-applied system with CRE also improved. Although sounds and labels are mixed up, Eq. (2) keeps an output not too much fitting to the mixed label.

Confidence of detected sound event versus number of frames graphs are shown in Fig. 7. In the aspect of detection as the inactive frame, many frames with detection results close to 0 when BCE was used for the objective function. Most of the detection results of the CRE-based system were also close to 0 but more spread from 0 to 0.02 than the system with BCE. In other words, CRE made the model less overfitted to inactive frames, which shows the class imbalance problem was relieved. Whereas, the detection confidences of the AFL-based system were evenly distributed rather than biased towards zero.

In the aspect of detection as the active frame, all systems show similar results to each other but have a little difference. The peak of the CRE-based system's curve was left-biased due to the threshold; however, the peak of the AFL-based system's curve was right-biased since the focal weight was set to train well for the active frames. It demonstrates that the CRE-based network is trained well up to the regularization threshold and is prevented from overfitting when the confidences come over the threshold. On the other hand, according to the focal weights, the AFL-based network is trained well focused for the active frames but less focused for the inactive frames.

5.2. Discussion: system characteristics and PSDS

As shown in the experimental result, the system's scores are different according to the evaluation metrics. For instance, the system trained with CRE outperformed the system with AFL on PSDS1 but vice versa on PSDS2. Then which system should we choose or

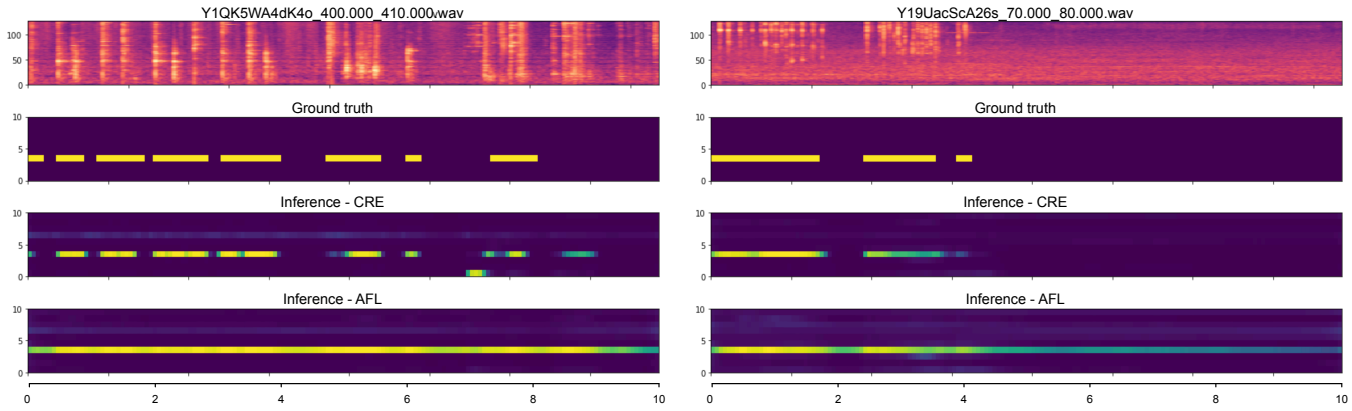


Figure 6: Two detection examples of CRE- and AFL-based systems. (From up to bottom: log-mel spectrogram, ground truth, CRE-based system, AFL-based system)

Table 1: Comparison of the detection performances among the different objective functions.

Network	Loss	Event-F1	PSDS1	PSDS2
CRNN w/o mixup	BCE	43.27	33.78	59.91
	AFL	40.63	31.99	**65.05**
	CRE	**45.50**	**33.98**	57.30
CRNN w/ mixup	BCE	44.93	34.79	58.60
	AFL	41.90	32.79	**60.92**
	CRE	**46.40**	**35.08**	57.82
CRNN+AWAM w/ mixup	BCE	49.17	37.51	66.34
	AFL	45.16	34.03	**66.88**
	CRE	**51.11**	**38.12**	64.33

which is better? As discussed in [1], PSDS1 is an effective metric for whether the system could detect the sound's timestamp correctly; whereas it has a severe problem related to the labeler's subjectivity. PSDS2 has strength in relieving the labeler's subjectivity; however it is hard to detect event localization precisely, and highly depends on the long-duration sounds.

For further description, we analyze the different detection patterns of the systems as shown in Fig. 6. Just as people label subjectively according to their background, systems have different characteristics under the objective functions. The CRE-based system tends to detect onsets and offsets precisely, whereas the AFL-based system tends to detect sound longer than the ground truth. In other words, the AFL-based system is proper to detect whether a sound appears in a clip rather than timestamps. The more general analysis is shown in the top graph of Fig. 7. The blue line shows that the confidences are more fitted to zero for inactive frames than green line. Instead, the green line is spread evenly without being biased to one side.

From the standpoint of user experience, the CRE-based system (system having high PSDS1 but low PSDS2) is required for users or environments that need to detect the target sound's onset and offset precisely. Whereas, the AFL-based system (system having high PSDS2 but low PSDS1) is more suitable in the environments for whether the appearance of target sounds is more important than detection resolution.

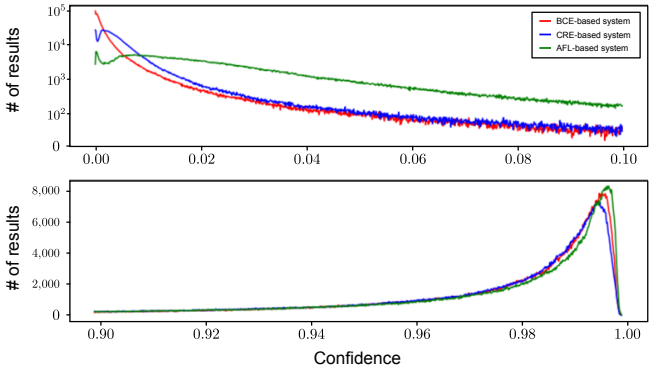


Figure 7: A graph of the number of frames over confidences of detected target sounds. Since there are a large number of inactive frames in the dataset, the graph of confidence near zero was log-scaled. (Top: confidence > 0.9, bottom: confidence < 0.1)

6. CONCLUSION

In this paper, we introduced the confidence regularized BCE that could avoid overfitting inactive frames. Compared to AFL, CRE performed better under the event-based f1 score and PSDS1. Furthermore, we suggested choosing the proper objective function according to the user's requirements. Of course, the system having good performance in both PSDS1 and PSDS2 is the best, but in a situation where you have to choose between the two, we can design a more suitable system by controlling the objective function.

7. ACKNOWLEDGEMENT

This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No.2021-0-00456, Development of Ultra-high Speech Quality Technology for Remote Multi-speaker Conference System)

8. REFERENCES

[1] G. Ferroni, N. Turpault, J. Azcarreta, F. Tuveri, R. Serizel, Ç. Bilen, and S. Krstulović, "Improving sound event detection metrics: insights from dcase 2020," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 631–635.

[2] H. Sundar, M. Sun, and C. Wang, "Event specific attention for polyphonic sound event detection," in *Proc. Interspeech*, 2021, pp. 566–570.

[3] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proc. IEEE international conference on computer vision*, 2017, pp. 2980–2988.

[4] K. Imoto, S. Mishima, Y. Arai, and R. Kondo, "Impact of sound duration and inactive frames on sound event detection performance," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 860–864.

[5] H. Zhang, M. Cissé, Y. N. Dauphin, and D. Lopez-Paz, "Mixup: Beyond empirical risk minimization," in *Proc. International Conference on Learning Representations (ICLR)*, 2018.

[6] S. Hershey, D. P. Ellis, E. Fonseca, A. Jansen, C. Liu, R. C. Moore, and M. Plakal, "The benefit of temporally-strong labels in audio event classification," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 366–370.

[7] J. F. Gemmeke, D. P. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio set: An ontology and human-labeled dataset for audio events," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017, pp. 776–780.

[8] H. Wang, Y. Zou, D. Chong, and W. Wang, "Environmental sound classification with parallel temporal-spectral attention," in *Proc. Interspeech*, 2020, pp. 821–825.

[9] N. Turpault, R. Serizel, J. Salamon, and A. P. Shah, "Sound event detection in domestic environments with weakly labeled data and soundscape synthesis," in *Proc. Detection and Classification of Acoustic Scenes and Events Workshop (DCASE)*, New York University, NY, USA, October 2019, pp. 253–257.

[10] A. Tarvainen and H. Valpola, "Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results," *Advances in neural information processing systems*, vol. 30, 2017.

[11] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," *arXiv preprint arXiv:1711.05101*, 2017.

[12] A. Mesaros, T. Heittola, and T. Virtanen, "Metrics for polyphonic sound event detection," *Applied Sciences*, vol. 6, no. 6, p. 162, 2016.

[13] Ç. Bilen, G. Ferroni, F. Tuveri, J. Azcarreta, and S. Krstulović, "A framework for the robust evaluation of sound event detection," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 61–65.

EDANSA-2019: THE ECOACOUSTIC DATASET FROM ARCTIC NORTH SLOPE ALASKA

Enis Berk Çoban[1], *Megan Perra*[2], *Dara Pir*[3], *Michael I Mandel*[1,4]

[1] The Graduate Center, CUNY, New York, NY, USA, ecoban@gradcenter.cuny.edu
[2] Institute of Arctic Biology, UAF, Fairbanks, AK, USA, meperra@alaska.edu
[3] Guttman Community College, CUNY, New York, NY, USA, dpir@gradcenter.cuny.edu
[4] Brooklyn College, CUNY, Brooklyn, NY, USA, mim@sci.brooklyn.cuny.edu

ABSTRACT

The arctic is warming at three times the rate of the global average, affecting the habitat and lifecycles of migratory species that reproduce there, like birds and caribou. Ecoacoustic monitoring can help efficiently track changes in animal phenology and behavior over large areas so that the impacts of climate change on these species can be better understood and potentially mitigated. We introduce here the Ecoacoustic Dataset from Arctic North Slope Alaska (EDANSA-2019), a dataset collected by a network of 100 autonomous recording units covering an area of 9000 square miles over the course of the 2019 summer season on the North Slope of Alaska and neighboring regions. We labeled over 27 hours of this dataset according to 28 tags with enough instances of 9 important environmental classes to train baseline convolutional recognizers. We are releasing this dataset and the corresponding baseline to the community to accelerate the recognition of these sounds and facilitate automated analyses of large-scale ecoacoustic databases.

Index Terms— Ecoacoustics, audio dataset, labeled data, baseline, biophony, anthrophony, geophony, convolutional network

1. INTRODUCTION

The Arctic Coastal Plain is an ecosystem in northern Alaska and Canada that hosts over 180 migratory bird species from nearly every continent on the planet. The health of this ecosystem is inextricably linked to other habitats across the globe [1] and is undergoing rapid change due to global warming [2, 3] and land-use change [4]. This region has rich oil and gas resources; extraction and transportation of these fossil fuels increase the usage of machinery and vehicles. As a result, anthrophony from industrial activity or aircraft overflights may change the acoustic environment in the area. Aircraft overflights associated with this activity have been a community concern in the region. One village on the Coastal Plain, Nuiqsut, experiences air traffic equivalent to a city 95 times its size [5]. Past acoustic monitoring studies in Alaska have been smaller in geographic scope and utilized coarse acoustic indices or manual labeling [6, 7]. While some of this research has addressed anthrophony [7], our dataset is the first to account for both developed and undeveloped regions across the Arctic Coastal Plain. Such recordings are valuable for understanding the natural state of the Arctic acoustic environment, how development changes that state, and how that change affects wildlife.

Passive acoustic monitoring is an effective tool to monitor this system—and many others—because acoustic data can tell us about changes in wildlife populations, including phenology [8], biodiversity [9], community structure [10], and distribution [11]. Because the volume of data produced by acoustic studies makes manual data processing prohibitively expensive, researchers have recently employed convolutional neural networks (CNN) to label the contents of large ecoacoustic datasets [12, 13, 14]. To train a CNN in a supervised fashion, researchers must label a small subset of data to train the model, and that labeled data is what we have provided in this paper. We used an earlier version of this dataset (batch-1, described in Subsection 2.3) in our research to understand the advantages of self-supervised learning and data valuation for audio classification [15]. We are providing the best performing model from that work as the baseline in this paper, which utilizes data augmentation [16, 17, 18] and global temporal pooling [19].

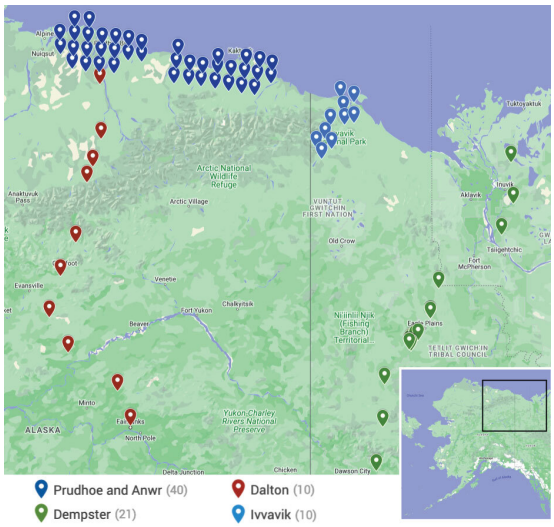


Figure 1: Audio recording device locations.

2. CORPUS

2.1. Monitoring sites

Samples were taken at latitudes between 64° and 70° N, and longitudes between 139° to 150° W, covering predominately the Arctic Coastal Plain but also spanning tundra, shrub, and boreal forest ecosystems on the north and south of the Brooks Mountain Range in northern Alaska. A map of the recording sites is shown in Figure 1. We used 40 recording devices to cover the Prudhoe Bay oilfields and the 1002 portions of the Arctic National Wildlife Refuge (ANWR) in a grid separating locations by 20 km with a random

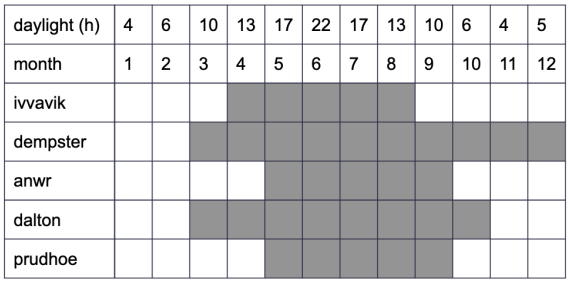


Figure 2: Audio recording times for monitors in each region. The top row displays the number of daylight hours for each corresponding month shown below it on the second row. The colored cells show the months where recordings were made for each region.

Label	b-1	b-2	b-3	b-4	Total
Biophony	4107	500	776	886	6269
Bird	3821	493	78	52	4444
Songbird	2210	238	5	6	2459
Waterfowl	573	126	2	3	704
Upland Bird	386	44	2	2	434
Insect	372	36	734	846	1988
Anthrophony	328	217	1367	1165	3077
Aircraft	100	93	731	769	1693
Silence	1146	325	32	19	1522
Total	5566	1045	2133	2038	10782

Table 1: Number of samples per label in each of the four batches, batch-1 (b-1) through batch-4 (b-4). The last row shows the total number of samples in each batch.

offset. We also had 20 recording devices at sites along the length of the Dempster and Dalton Highways (10 devices each) in the Yukon Territories and Alaska, respectively, and an additional 10 recording devices placed at existing wildlife monitoring sites throughout Ivvavik National Park in the Yukon. Devices are deployed over 3-5 days in each region, starting from the 16th and 23rd of March, 2nd, 3rd, and 4th of May, respectively, in Dalton, Dempster, Ivvavik, Prudhoe, and Anwr. Recordings are saved locally on the device and collected manually. The acoustic recording units (ARUs) were SM4 wildlife recorders from Wildlife Acoustics, sampling at a rate of 48 kHz with gains set to 16 dB. ARUs recorded 150 minutes of audio at a time, with rotating breaks of 120 to 150 minutes in order to cycle through every hour of the day within a 4-day period. In total, devices recorded 2,161 days of audio data throughout 2019.

Recording periods for each region are shown in Figure 2. Recordings from the Prudhoe Bay Oilfields, ANWR, and Ivvavik were able to capture wildlife activity on the Arctic Coastal Plain and in the foothills of the Brooks Range, which serve as the primary breeding grounds for a majority of migratory species in the area. Sites along the Dempster and Dalton Highway captured the arrival and departure of those migratory birds that use the major north-south flyways that converge on the Coastal Plain.

2.2. Taxonomy

Labels of our dataset are members of three taxonomic ranks: coarse, medium, and fine. The coarse rank is the highest rank of the taxonomic tree and contains the four most general labels: *"anthrophony"*, *"biophony"*, *"geophony"*, and *"silence"*. A higher rank contains more general labels compared to the lower ranks. The medium rank contains more specific labels for each of the coarse rank labels and includes *"bird"*, *"insect"*, and *"aircraft"*. The fine rank consists of the most specific labels, each belonging to one of the medium rank categories and includes *"songbird"*, *"waterfowl"* (which in our dataset includes ducks, geese, and swans), and *"upland bird"* (including grouse and ptarmigan). Labels used in our baseline system are shown on the leftmost column of Table 1.

For example, a sample that contains a birdsong event is annotated with the *"biophony"*, *"bird"*, and *"songbird"* tags representing labels from the coarse, medium, and fine ranks, respectively. Annotating a sample with a child label will automatically annotate it with the parent label, however, it is possible for a sample to be annotated only with a parent label. Our annotators could assign nearly all samples a designation from the coarse labels. Fine labels under the bird category were not uncommon, but fine labels under other categories were relatively rare or absent.

2.3. Sampling and labeling

Our dataset is composed of four batches differentiated by the sampling and labeling methods used, which are described below. We initially labeled data with broad coverage of time and space in our dataset so as to capture as many sound classes as possible without bias. We pulled our initial batch, batch-1, of random samples from each site within the Arctic National Wildlife Refuge and the adjacent Oilfields (sites 11-50). We selected a random 150-minute contiguous recording within each site, visually examined each recording's spectrogram in Audacity [20], and labeled all visually identified sounds present in that recording via listening. We excluded sound recordings that were inaudible due to wind-related clipping. We describe the details of four examples in Subsection 2.4.

The same expert labeler labeled sound clips in all batches based on the taxonomy described in Subsection 2.2 In almost all cases, sounds could be identified at the coarsest scale (e.g., *"anthrophony"* or *"biophony"*), and more specific labels were added as they could be identified. This generated 3083 separately labeled sounds that ranged in length from a few seconds to a few minutes. To generate equal-length samples, they were then split into 5566 non-overlapping, 10-second clips. However, not all clips' lengths were divisible by 10, generating clips less than 10 seconds. Clips less than 2 seconds were discarded, and clips between 2 and 10 seconds were zero-padded to the full 10 seconds.

We divided the whole set of samples into training, validation, and test sets so that all samples from a given site were confined to one of these three sets rather than split between them. To determine which site went to which set, we used a multiple knapsack problem detailed in [15]. This split ensured that we were measuring generalization across sites, and thus to future recordings.

In order to increase the number of examples in the rare class *"anthrophony"*, we labeled more samples that we expected to be relevant from sites where this class was more common using predictions of a model trained on the training set of batch-1. At those sites, we pulled 500 10-second clips not tied to model confidence and 500 clips where model confidence for anthrophony was 0.75 or higher. We split the season into 12 weekly periods starting on May 7 and pulled $\sim$80 samples from each weekly period, 40 of which were not tied to model confidence and were just the first 40 samples from that week, and the other 40 of which were tied to model confidence and tended to be distributed more throughout the week.

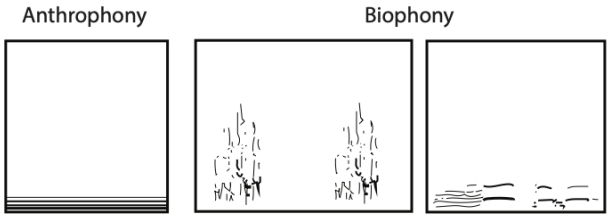


Figure 3: The depictions in this figure highlight the visual characteristics of spectrograms from four examples produced in Audacity. From left to right, these represent a truck driving on a road, a bird singing, and a loon singing near an acoustic recording unit (in sites 26, 50, and 11, respectively).

For each week, we randomly chose one of the training sites (30, 29, 25, 24, 22, 21, 19, 16, 13, 49, 48, 44, 41, 40, 38, 37, 33) to pull samples from. The proportion of all 1000 samples taken from each training site depended on the proportion of anthrophony that was identified at that site in batch-1. So if 30% of all anthrophony in batch-1 had come from one site, 30% of our samples for batch-2 were also pulled from that site. This meant that, in some cases, we had to use multiple sites to fill our quota of samples for a given week. This process created a second batch of clips, batch-2. All of the clips from this 1000-sample set were labeled by trained undergraduate students, and then their labels were reviewed and corrected by an expert labeler before being finalized.

To ensure the accuracy of our original samples, plus the additional anthrophony samples, we built a user interface (UI) in Python that allowed users to quickly listen to a sample, view its spectrogram, and label it. The expert labeler checked or unchecked boxes next to each possible label to validate the original labels associated with a sample. All 6616 10-second clips that we had previously labeled were reviewed using this process. Note that all numbers provided in Table 1 are after this relabeling.

The UI made it considerably more efficient to review clips, allowing us to label additional samples from the *"aircraft"* sound class by selecting high-confidence predictions from the baseline model trained on batches 1 and 2. This created our third batch, batch-3. Note that batch-3 is only used for training, so we are less concerned that this selection process might bias the labels. Using this process, we were able to label an additional 2133 clips.

Performance was still below what we had hoped for the sound classes *"insect"*, *"anthrophony"* and *"aircraft"*, so we decided to collect a final batch of data, batch-4. In this batch, we labeled samples for the validation and test sets. Since selecting samples to label using a single model's confidence scores could lead to choosing only the type of samples that are successfully recognized by this model, we used an ensemble of 7 different iterations of our sound labeling model, including architectural and training variants, developed with earlier versions of the dataset. We normalized the confidence scores of each model across time to be between 0 and 1. Then assigned the maximum confidence across models to each label on each clip. Again, we pulled clips where this combined confidence was high, though this differed for each sound class as follows: 0.7 for anthrophony, 0.25 for aircraft, and 0.99 for the insect. The expert labeler used the UI to review these clips and was able to identify an additional 1,874 instances of labels so that the final set of label-clip associations was over 10,000. Table 1 displays the number of samples per label in each of the four batches of our dataset.

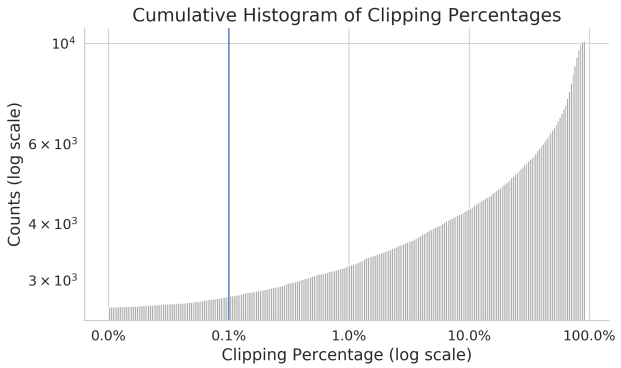


Figure 4: Number of clips out of 10,000 randomly sampled that have at most a particular clipping percentage (percentage of samples at maximum or minimum value).

2.4. Labeling examples

When labeling batch-1, we viewed the 150-minute contiguous recordings as spectrograms in Audacity so we could locate sounds visually. The spectrogram was shown up to 24 kHz so that all possible sounds were visible. While the harmonics of birdsong could extend to 10-12 kHz, a majority of the visible signal on the spectrogram was at 9 kHz or below. The presence of anthrophony was generally indicated by lower frequency, uniform partials that tend to have a consistent pitch and long duration. The presence of biophony presents as harmonic signals with clear partials that have a wide frequency range and short duration, though they are often part of a larger temporal pattern, or song. Rain was the most common form of geophony and was clearly indicated by extremely brief, nonharmonic signals on the spectrogram that look similar to 'clicks', i.e., they have a vertical, broadband form that does not follow an intentional temporal pattern.

Figure 3 displays depictions that highlight the visual characteristics of spectrograms from three examples produced in Audacity. From left to right, the first depiction, labeled *"anthrophony"*, represents a truck driving on a road, near an acoustic recording unit (site 26) in the oilfields. The lines highlight the visible, flat partials at the lower frequencies that are characteristic of anthrophony sounds. The second depiction represents a bird singing near an acoustic recording unit (site 50) in the Arctic National Wildlife Refuge and is labeled *"songbird"*. The patterns highlight the signals created by the birdsong, which show a complex pattern of partials that extends from low to high frequencies; this complex song is repeated multiple times to produce a 'singing bout'. The third depiction represents a loon singing near an acoustic recording unit (site 11) in the oilfields and is labeled *"waterfowl"*. The patterns highlight partials of the loon call, which show a definitive temporal pattern.

2.5. Clipping

Clipping is distortion caused by loud sounds. Recorders have a high and low limit for the amplitude of the sounds they can process; if these thresholds are passed, the data gets corrupted. We count the proportion of sample values in a 10-second interval that are at the maximum or minimum observed levels in a given recording.

An acceptable clipping percentage depends on the specific application. To pick a threshold of acceptable clipping level, we lis-

tened to random 10-second clips and observed that a clipping proportion less than 0.1% is almost unnoticeable to a listener. We therefore removed samples with more than 0.1% clipping before labeling in batch-2, batch-3, and batch-4 of our dataset. Figure 4 shows for 10,000 randomly selected clips, how many had clipping less than a given percentage. In a similar fashion, for batch-1, the expert labeler checked the recordings' spectrogram and only labeled those without any visible clipping artifacts. 80% of these samples from batch-1 have a clipping percentage lower than the 0.1% threshold.

3. BASELINE

Our initial labeling included examples of 28 unique categories in all levels of the hierarchy, out of 41 we thought we might encounter. Of these, only 9 had more than 100 examples in batch-1 and were used to train our baseline model: *"biophony"*, *"bird"*, *"songbird"*, *"waterfowl"*, *"upland bird"*, *"insect"*, *"athrophony"*, and *"silence"*.

As a baseline system[1], we provide the best system described in [15]. Our baseline system employs the Ecoacoustic Dataset from Arctic North Slope Alaska (EDANSA-2019)[2] and uses convolutional neural networks (CNNs) together with global temporal pooling and data augmentation. We share our code with MIT and our dataset under Creative Commons 4.0 licenses, which are highly permissive. We decided to use CNNs with hyperparameters inherited from AlexNet [21] due to their common success in sound event detection experiments [22]. Each sample is a 10-second clip, preprocessed and turned into a mel-spectrogram with a hop size of 23 ms, a window size of 42 ms, and 128 mel-frequency bins. We use a stack of 4 convolutional layers where all kernels are 5×5, followed by two fully connected layers. We train our model for 1600 epochs and keep the one with the highest mean AUC score over all labels on the validation set. Table 2 shows the AUC per label of the baseline model on the validation and test sets.

4. PREVIOUS WORK

There are a number of open-source datasets, similar to ours, shared along with their research findings. The CityNet dataset, which is collected from London, has diverse anthropogenic classes but the biophony classes are limited to only general labels like *"bird"*, *"insect"*, *"vegetation"*, and *"wing beats"* [23]. Another soundscape dataset consists of 5 hours of recordings collected from Sonoma County, California, USA and samples are labeled with *"anthropophony"*, *"biophony"*, *"geophony"*, *"quiet"*, and *"interference"* [14]. The main difference between these datasets and ours is that ours is recorded in remote locations and over a much larger area. Our dataset consists of 29 hours of labeled data, compared to 19 hours in CityNet and 5 hours from Sonoma County.

There are large datasets focusing on bird calls, which are challenging to model and of high scientific interest. BIRDCLEF is a family of such datasets focusing on short targeted recordings as opposed to long-term continuous recordings. It consists of sound recordings collected by the Xeno-canto community and new versions with different purposes have been released every year since 2014. The latest, 2022 version, consists of 15k recordings, totaling over 190 hours covering 152 species from Hawaii, specially designed for modeling calls of rare and endangered bird species with small amounts of training data [24]. Another dataset with 385 minutes of dawn chorus recordings was collected from Eastern North America, including 48 species and 16,052 annotations [25]. Some of the other datasets with bird calls are BirdVox [26], Nips4Bplus [27], Freefield1010 [28], Warblrb10k and PolandNFC [29].

Larger general-purpose datasets have been extracted from YouTube such as Audio Set [30] and VGGSOUND [31] and include bioacoustic classes as a small part of their corpus. There are also continuously recorded open-source sound datasets without bioacoustic labels, such as SONYC-UST-V2, which is the output of an urban noise monitoring project and it is a multi-labeled [32]. This dataset is ∼51 hours long in total and labeled with 8 main tags that are common in city environments, such as engine, music, and the human voice.

Label	Validation	Test
Biophony	0.95	0.96
Bird	0.96	0.98
Songbird	0.90	0.96
Waterfowl	0.87	0.90
Upland bird	0.87	0.93
Insect	0.90	0.83
Anthrophony	0.88	0.88
Aircraft	0.96	0.88
Silence	0.96	0.93
Average	0.92	0.92

Table 2: AUC per label of the baseline on validation and test sets.

5. CONCLUDING REMARKS

This paper presented the Ecoacoustic Dataset from Arctic North Slope Alaska (EDANSA-2019), collected by autonomous recording units during the summer of 2019, and its corresponding baseline. We provided detail on the recordings and the sampling and labeling methods used to generate the four batches of our dataset. This work should help facilitate the analysis of large-scale ecoacoustic recordings made in arctic conditions, and it would be interesting to examine the extent to which models trained on this data can generalize to data collected in other environments and ecosystems.

6. ACKNOWLEDGMENT

We are grateful to Scott Leorna, Dr. Todd Brinkman, and Dr. Natalie T. Boelman for help with data collection and Eleanor Davol for help with labeling. This work is supported by the National Science Foundation (NSF) grant OPP-1839185. Any opinions, findings, and conclusions or recommendations are those of the author(s) and do not necessarily reflect the views of the NSF.

7. REFERENCES

[1] B. K. Sullender. (2019, Apr) Migratory birds in the heart of the arctic: A journey through the arctic national wildlife refuge.

[2] J. E. Walsh and B. Brettschneider, "Attribution of recent warming in alaska," *Polar Sci.*, vol. 21, pp. 101–109, 2019.

[3] P. Fauchald, T. Park, H. Tømmervik, R. Myneni, and V. H. Hausner, "Arctic greening from warming promotes declines

[1] https://github.com/speechLabBcCuny/EDANSA-2019

[2] https://zenodo.org/record/6824272

in caribou populations," *Science Advances*, vol. 3, no. 4, p. e1601365, 2017.

[4] M. K. Raynolds, D. A. Walker, K. J. Ambrosius, J. Brown, K. R. Everett, M. Kanevskiy, G. P. Kofinas, V. E. Romanovsky, Y. Shur, and P. J. Webber, "Cumulative geoecological effects of 62 years of infrastructure and climate change in ice-rich permafrost landscapes, prudhoe bay oilfield, alaska," *Glob. Ch. Bio.*, vol. 20, no. 4, pp. 1211–1224, 2014.

[5] T. R. Stinchcomb, T. J. Brinkman, and D. Betchkal, "Extensive aircraft activity impacts subsistence areas: acoustic evidence from arctic alaska," *Environmental Research Letters*, vol. 15, no. 11, p. 115005, oct 2020.

[6] R. Buxton, E. Brown, L. Sharmin, C. M. Gabriele, and M. F. McKenna, "Using bioacoustics to examine shifts in songbird phenology," *Eco. & Evol.*, vol. 6, pp. 4697–4710, 05 2016.

[7] H. Vincelette, R. Buxton, N. Kleist, M. F. McKenna, D. Betchkal, and G. Wittemyer, "Insights on the effect of aircraft traffic on avian vocal activity," *Ibis*, vol. 163, no. 2, pp. 353–365, 2021.

[8] R. Y. Oliver, D. P. W. Ellis, H. E. Chmura, J. S. Krause, J. H. Pérez, S. K. Sweet, L. Gough, J. C. Wingfield, and N. T. Boelman, "Eavesdropping on the Arctic: Automated bioacoustics reveal dynamics in songbird breeding phenology," *Sci Adv*, vol. 4, no. 6, p. eaaq1084, 06 2018.

[9] D. Proppe, C. Sturdy, and C. St. Clair, "Anthropogenic noise decreases urban songbird diversity and may contribute to homogenization," *Glob. Ch. Bio.*, vol. 19, pp. 1075–84, 04 2013.

[10] C. D. Francis, C. P. Ortega, and A. Cruz, "Noise pollution filters bird communities based on vocal frequency," *PLOS ONE*, vol. 6, no. 11, pp. 1–8, 11 2011.

[11] N. J. Kleist, R. P. Guralnick, A. Cruz, and C. D. Francis, "Sound settlement: noise surpasses land cover in explaining breeding habitat selection of secondary cavity-nesting birds," *Ecol Appl*, vol. 27, no. 1, pp. 260–273, 01 2017.

[12] S. S. Sethi, N. S. Jones, B. D. Fulcher, L. Picinali, D. J. Clink, H. Klinck, C. D. L. Orme, P. H. Wrege, and R. M. Ewers, "Characterizing soundscapes across diverse ecosystems using a universal acoustic feature set," *PNAS*, vol. 117, no. 29, pp. 17 049–17 055, 2020.

[13] J. LeBien, M. Zhong, M. Campos-Cerqueira, J. P. Velev, R. Dodhia, J. L. Ferres, and T. M. Aide, "A pipeline for identification of bird and frog species in tropical soundscape recordings using a convolutional neural network," *Ecological Informatics*, vol. 59, p. 101113, 2020.

[14] C. A. Quinn, P. Burns, G. Gill, S. Baligar, R. L. Snyder, L. Salas, S. J. Goetz, and M. L. Clark, "Soundscape classification with convolutional neural networks reveals temporal and geographic patterns in ecoacoustic data," *Ecological Indicators*, vol. 138, p. 108831, 2022.

[15] E. B. Çoban, A. R. Syed, D. Pir, and M. I. Mandel, "Towards large scale ecoacoustic monitoring with small amounts of labeled data," in *Proc. WASPAA*, 2021, pp. 181–185.

[16] T. Ko, V. Peddinti, D. Povey, and S. Khudanpur, "Audio augmentation for speech recognition," in *Proc. INTERSPEECH*, 2015, pp. 3586–3589.

[17] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," *arXiv preprint arXiv:1710.09412*, 2017.

[18] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "Specaugment: A simple data augmentation method for automatic speech recognition," *arXiv preprint arXiv:1904.08779*, 2019.

[19] Q. Kong, Y. Cao, T. Iqbal, Y. Wang, W. Wang, and M. D. Plumbley, "Panns: Large-scale pretrained audio neural networks for audio pattern recognition," *Tr. ASLP*, vol. 28, pp. 2880–2894, 2020.

[20] A. Team, 2019. [Online]. Available: https://www.audacityteam.org

[21] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *NeurIPS*, vol. 25, pp. 1097–1105, 2012.

[22] K. Choi, G. Fazekas, and M. B. Sandler, "Automatic tagging using deep convolutional neural networks," in *ISMIR*, 2016.

[23] A. J. Fairbrass, M. Firman, C. Williams, G. J. Brostow, H. Titheridge, and K. E. Jones, "Citynet—deep learning tools for urban ecoacoustic assessment," *Methods in Ecology and Evolution*, vol. 10, no. 2, pp. 186–197, 2019.

[24] S. Kahl, A. Navine, T. Denton, H. Klinck, P. Hart, H. Glotin, H. Goëau, W.-P. Vellinga, R. Planqué, and A. Joly, "Overview of birdclef 2022: Endangered bird species recognition in soundscape recordings," *Working Notes of CLEF*, 2022.

[25] L. M. Chronister, T. A. Rhinehart, A. Place, and J. Kitzes, "An annotated set of audio recordings of eastern north american birds containing frequency, time, and species information," *Ecology*, vol. 102, 2021.

[26] V. Lostanlen, J. Salamon, A. Farnsworth, S. Kelling, and J. P. Bello, "Birdvox-full-night: A dataset and benchmark for avian flight call detection," in *ICASSP*, 2018, pp. 266–270.

[27] V. Morfi, Y. Bas, H. Pamuła, H. Glotin, and D. Stowell, "Nips4bplus: a richly annotated birdsong audio dataset," *PeerJ Computer Science*, vol. 5, p. e223, 2019.

[28] D. Stowell and M. D. Plumbley, "An open dataset for research on audio field recording archives: freefield1010," *arXiv preprint arXiv:1309.5275*, 2013.

[29] F. Berger, W. Freillinger, P. Primus, and W. Reisinger, "Bird audio detection-dcase 2018," *DCASE*, 2018.

[30] J. F. Gemmeke, D. P. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio set: An ontology and human-labeled dataset for audio events," in *ICASSP*, 2017, pp. 776–780.

[31] H. Chen, W. Xie, A. Vedaldi, and A. Zisserman, "Vggsound: A large-scale audio-visual dataset," in *ICASSP*, 2020, pp. 721–725.

[32] M. Cartwright, J. Cramer, A. E. M. Mendez, Y. Wang, H.-H. Wu, V. Lostanlen, M. Fuentes, G. Dove, C. Mydlarz, J. Salamon, *et al.*, "Sonyc-ust-v2: An urban sound tagging dataset with spatiotemporal context," *arXiv preprint arXiv:2009.05188*, 2020.

ENSEMBLE OF MULTIPLE ANOMALOUS SOUND DETECTORS

Yufeng Deng[1], *Anbai Jiang*[1], *Yuchen Duan*[1], *Jitao Ma*[3], *Xuchu Chen*[1], *Jia Liu*[1,2], *Pingyi Fan*[1], *Cheng Lu*[3], *Wei-Qiang Zhang*[1]

[1] Department of Electronic Engineering, Tsinghua University, Beijing, China
[2] Tsinghua AI Plus, Beijing, China
[3] School of Economics and Management, North China Electric Power University, Beijing, China
dyf20@mails.tsinghua.edu.cn, {liuj, fpy, wqzhang}@tsinghua.edu.cn, lucheng1983@163.com

ABSTRACT

This paper presents our submission to DCASE 2022 Challenge Task 2, which aims to detect anomalous machine status via sounding by using machine learning methods, where the training dataset itself does not contain any examples of anomalies. We build six subsystems, including three self-supervised classification methods, two probabilistic methods and one generative adversarial network (GAN) based method. Our final submissions are four ensemble systems, which are different combinations of the six subsystems. The best official score of the ensemble systems can achieve 86.81% on the development dataset, whereas the corresponding Autoencoder-based baseline and the MobileNetV2-based baseline are with scores of 52.61% and 56.01%, respectively. In addition, our methods rank top on the development dataset and fourth on the evaluation dataset in this challenge.

***Index Terms*—** DCASE, anomaly detection, domain generalization, machine condition monitoring, machine health monitoring

1. INTRODUCTION

The DCASE 2022 Challenge Task 2 is concerned with detecting anomalous state of the target machine using the sounding data. Unlike the acoustic scene classification, the available training data in this task contains samples of only one class — the normal-state class, but the aim is to detect whether a test sample is in another class, refer to as anomaly class, which may include various anomalous situations. A further complication added to this challenge is that the distributions of the training data and of the test data are different. This is called as domain shift. In the literature, there are some works investigating how to solve the domain shift problem by using machine learning methods and reducing the performance gap between the training and test data [1]. Although these techniques achieved impressive performance on image classification, they did not generally gain the expected and comparable results in the machine status detection via sounding up to now.

In this paper, we present six subsystems, the first three are self-supervised classifiers trained by using the supervision information provided by the metadata, similar to the approach taken by several teams at DCASE 2021 [2, 3] and DCASE 2020 [4]. The fourth and fifth models are probabilistic models. For the fourth model, inspired by the probabilistic model in [2], we employ normalizing flows to estimate the conditional density of the feature vectors for each section where the Mel spectrograms are used as the input of the pooling layers. Those output above the defined threshold will be marked as anomaly. The fifth model estimates the conditional density of spectrogram target frames conditioned on remaining frames using an RNN based model and a GMM loss. The sixth model is a GAN based model.

We next present the results of the baseline systems in Section 2, and then describe each of our subsystems in detail in Section 3. For each subsystem, we will describe how it is trained and present its results on the development dataset. After that, we present our ensemble systems and their detection results in Section 4. Finally, conclusions are drawn in Section 5.

2. BASELINE RESULTS

In order to give a clear picture of the Challenge Task 2, we include the baseline scores on Table 1 and Table 2. To present the results succinctly, the results in all tables in this paper present only the harmonic mean of the source AUC, target AUC and pAUC for each machine type on the development dataset. Here the harmonic mean is denoted as h-mean. The data used in this challenge is 16 kHz, single-channel audio. For more details, please see [5, 6, 7].

Table 1: baseline-AE results

bearing	fan	gearbox	slider	ToyCar	ToyTrain	valve	h-mean
54.80%	58.47%	63.07%	57.99%	51.06%	39.61%	50.59%	52.61%

Table 2: baseline-MobileNetV2 results

bearing	fan	gearbox	slider	ToyCar	ToyTrain	valve	h-mean
59.16%	57.21%	59.91%	50.26%	54.23%	51.18%	62.42%	56.01%

3. APPROACHES

The general idea of the first three approaches is to first train a neural network to extract the embeddings of the samples by classifying labels extracted from the metadata, and then use the outlier detection algorithm to score how abnormal the embeddings are. The input to the first two models is a spectrogram with or without a Mel transformation, the difference between the first two models is that the first model uses a single index loss detection, while the second model uses a multiple indices loss detection. For the third model, the inputs are the embeddings extracted by the pre-trained wav2vec [8],

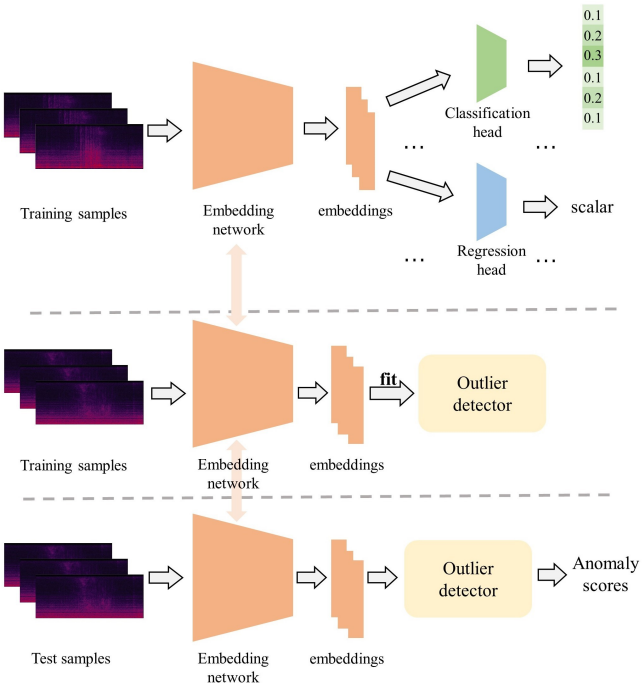


Figure 1: Overview of the classification based method.

which is trained with a partial Audioset [9]. The loss function for the first three models is ArcFace [10]. The fourth and fifth models differ from the first three in that they are not trained using any metadata information, which makes them completely unsupervised. The fourth model attempts to learn several distributions of some feature vector bins conditioned on the remaining bins. The fifth model estimates the distribution of target frames for the spectrogram conditioned on the remaining frames. The sixth model is a GAN based model. After describing these six subsystems, we present four ensembles of these subsystems, which are our final submission.

3.1. Classification Based Methods

In this subsection, we describe the first three subsystems, which we call SC (**S**ection **C**lassification), MHCR (**M**ulti-**h**ead **C**lassification & **R**egression) and SC-wav2vec. All of the three subsystems follow the overview shown in Figure 1.

The overview shown in Figure 1 is divided into three processes. First, we use the training samples to train the embedding network, and then use the trained embedding network to extract the embeddings of the training samples. Later on, we use these embeddings to train the outlier detector. Finally, the trained embedding network is used to extract the embeddings of the test samples, and the trained outlier detector is used to score the abnormality of these embeddings. In Figure 1, the embedding is extracted from the output of the last or penultimate layer of the embedding network. Since the ArcFace [10] loss function increases the inter-class distance and decreases the intra-class distance so that the network learns a better representation of the data, an ArcFace [10] layer is usually chosen for the classification head. The regression header is a fully connected layer.

3.1.1. Features & Training

The input feature of the embedding network used in the SC and MHCR is STFT spectrogram with or without a Mel transformation. For the SC-wav2vec, the input feature is the embedding extracted from a pre-trained wav2vec [8], which is trained using a partial AudioSet [9]. The logarithm is taken for the Mel spectrogram but not taken for the STFT spectrogram. We select the optimal STFT frame length, hop length, and number of frames based on the results on the development dataset and decide whether to use the Mel transform for each machine type. The specific feature parameters are shown in Table 3.

For the SC, the embedding network is trained to predict the section IDs using ArcFace [10] loss function. Since only the section ID metadata is used, there is only one classification head in the SC. For the MHCR, we additionally use other tags in the filenames to design classification (factory noise, microphone position, etc.) or regression (speed, weight, etc.) tasks. In order to achieve multi-label classification or regression, multiple parallel classification or regression heads are used. Different tasks are trained simultaneously. The hyperparameter λ is used to balance multiple losses, as shown in (1).

$$\mathcal{L} = \sum_i \lambda_i \mathcal{L}_i \tag{1}$$

where $\lambda_i > 0$ and $\mathcal{L}_i$ is the loss of label i. For the SC-wav2vec, we first train wav2vec [8] model using Fairseq [11] on the balanced AudioSet [9], while the features extracted using the pre-trained wav2vec [8] are input to the embedding network. The supervised label for the SC-wav2vec is the section IDs, hence, only one classification head is used in the SC-wav2vec.

Based on various experiments using the training dataset provided in section 0-2, we observe that the architecture of the embedding network has a significant impact on the performance and the optimal network architecture is different for different machine types, so we select the best performing network from MobileFaceNet (MFN) [12], MFNSE, Ecapa-tdnn [13] and Cnn6 for each machine type. The MFNSE is an improved network from the MFN, the difference between the MFNSE and MFN is that we add a squeeze-excitation [14] block to the bone block of the MFN. The Cnn6 is a 6-layer convolutional network used in [15].

For the training of embedding network, we adopt the AdamW optimizer with the default learning rate of 1×10^{-3}, weight decay 1×10^{-4}, and 25 epochs for training, where all the training data is drawn from the development and evaluation datasets. When training the wav2vec [8] model on the balanced AudioSet [9], we use the initial learning rate of 1×10^{-7} and linearly increase the learning rate to 5×10^{-3} in the first 500 updates, then decay to 1×10^{-6} along the cosine curve.

3.1.2. Anomaly Detection Algorithms

Because the training samples only contain the normal ones, outlier detection algorithms are used to detect anomalous samples. Once the training is completed, the embedding network is used to extract the embeddings of the input samples, which are used to fit the outlier detectors. For the outlier detection, we employ four well known algorithms, k-NN [16], LOF [17], cosine distance and Mahalanobis distance. For the cosine distance and Mahalanobis distance, we compute the average embedding using the embeddings of the training samples, and additionally compute the covariance matrix for the Mahalanobis distance. In the test phase, the average

embedding and covariance matrix are used to compute the cosine and Mahalanobis distances of the test embedding, and use them to present the anomaly scores.

3.1.3. Results

Table 3 shows the results of the SC, Table 4 shows the results of the MHCR, while Table 5 shows the results of the SC-wav2vec.

Table 3: The results of SC

	bearing	fan	gearbox	slider	ToyCar	ToyTrain	valve
feature	STFT	STFT	STFT	STFT	logmel	STFT	STFT
nMels	-	-	-	-	64	-	-
nffts	2048	2048	2048	2048	2048	2048	2048
nFrames	192	192	192	192	192	192	192
network	MFN	MFNSE	MFN	MFNSE	Ecapa-tdnn	Cnn6	MFNSE
h-mean	82.20%	80.64%	83.20%	88.27%	77.50%	75.92%	96.84%

Table 4: The results of MHCR

bearing	fan	gearbox	slider	ToyCar	ToyTrain	valve	h-mean
72.07%	72.16%	84.01%	83.18%	71.10%	68.14%	95.28%	77.01%

3.2. Probabilistic Models

In this subsection, we describe the fourth and fifth subsystems, which we call WSP-NFCDEE and IMDN (**I**nterpolation **M**ixture **D**ensity **N**etwork).

The WSP-NFCDEE is built on the NFCDEE proposed in [2]. The difference between the NFCDEE and the WSP-NFCDEE is that we add a **W**eighted **S**tatistic **P**ooling (WSP) layer before the normalizing flows, which improves the performance on most machine types, especially on the slider. Hence, we call this method WSP-NFCDEE. Let $\boldsymbol{X} \in \boldsymbol{R}^{M \times T}$ is a Mel spectrogram, where M is the number of Mel bins and T is the number of frames. The WSP computes the mean and standard deviation of $\boldsymbol{X}$ along time axis to obtain the mean vector $\boldsymbol{y} \in \boldsymbol{R}^{M}$ and the standard deviation vector $\boldsymbol{z} \in \boldsymbol{R}^{M}$. The output of the WSP is $\alpha \cdot \boldsymbol{y} + \beta \cdot \boldsymbol{z}$, where the α and β are two trainable parameters satisfying the constraints (2).

$$\alpha + \beta = 1, \alpha, \beta > 0 \tag{2}$$

For the IMDN, we adopt three network structures mainly based on CNN and GRU, and a special density estimation loss function that combines the IDNN structure and Gaussian mixture model. We employ three networks named IGNN, LRCGNN, and a simplified DeepFilterNet [18]. Sufficient experiments had shown that networks based on RNN structure perform well on time series inputs. Hence, we select the 3D input with this form, (batch size, nFrames, nMels), which will be later propagated forward in the time dimension by the GRU, while CNN is mainly in the frequency dimension. We first select two lightweight network architectures, IGNN and LRCGNN. These two networks are with relatively low computation complexity, but achieve significant improvements over the Autoencoder-based baseline. The architecture of IGNN is shown in Table 6. The LRCGNN additionally add a Conv1d layer after the first fully connected layer of IGNN.

Table 5: The results of SC-wav2vec

	bearing	fan	gearbox	slider	ToyCar	ToyTrain	valve
Scoring	maha	maha	k-NN	k-NN	LOF	k-NN	cosine
h-mean	58.54%	57.16%	58.01%	66.77%	71.90%	65.03%	68.89%

Table 6: The architecture of IGNN

layer_name	parameters
Fully Connected	(nMels, 128)
3 × GRU	(128, 128)
Fully Connected	(128, 32)
Fully Connected	(32, 128)
3 × GRU	(128, 128)
Fully Connected	((nF-1)×128, 2×nC×nMels)

To achieve a better performance on more complex data, we employ DeepFilterNet (DFnet) [18], a more complex Unet-like network proposed in speech enhancement. The computational complexity of the original DeepFilterNet [18] is high, so we use two fully connected layers in place of the original two convolutional layers, which greatly reduces the amount of computation. Additionally, we add a fully connected layer to match the number of Mel bins.

IDNN [19] was demonstrated to achieve significant improvement on non-stationary signals, which predicted the center frame of the input Mel spectrogram. We also adopt this idea. Another modification is that the Gaussian mixture model is adopted as the loss function. Let $\boldsymbol{x}_p$ is the p-th frame to be predicted. By mapping input features $\boldsymbol{X}$ to the parameters of the GMM, we obtain the frame predictions in probabilistic form (In the test system, the component weights are selected the same value):

$$p\left(\boldsymbol{x}_p|\boldsymbol{X}, D, E\right) = \sum_{m=1}^{C} p_m\left(\boldsymbol{x}_p|\boldsymbol{X}, D, E\right) \tag{3}$$

where C is the number of Gaussian components and p_m is the density of the m-th Gaussian component. Since the aim is to find the maximum probability density of the predicted frame, the LSE function is as follows.

$$\mathcal{L}_{LSE} = -\log \sum_{i=1}^{M} \exp\left(\sum_{m=1}^{C} \log p_m\left(x_p|\boldsymbol{x}, D, E\right)\right) \tag{4}$$

where M is the number of Mel bins. When the covariance matrix of GMM is taken as diagonal matrix, this loss function is proved to be equivalent to maximization of the log likelihood of network with the given data.

3.2.1. Features & Training

The input feature of the WSP-NFCDEE and the IMDN is STFT spectrogram with Mel transformation and the logarithm is taken for the Mel spectrogram. For different machine types we tune the values of nMels, nffts, nFrames.

3.2.2. Results

Table 7 shows the results of the WSP-NFCDEE, while Table 8 shows the results of the IMDN.

Table 7: The results of WSP-NFCDEE

	bearing	fan	gearbox	slider	ToyCar	ToyTrain	valve
nMels	128	128	64	64	64	64	64
nffts	2048	2048	2048	2048	2048	2048	1024
nFrames	100	60	100	100	100	100	30
network	LRCGNN	LRCGNN	DFnet	LRCGNN	LRCGNN	DFnet	IGNN
h-mean	68.12%	68.54%	79.96%	78.52%	65.23%	60.29%	68.27%

Table 8: The results of IMDN

	bearing	fan	gearbox	slider	ToyCar	ToyTrain	valve
nMels	128	128	64	64	64	64	64
nffts	2048	2048	2048	2048	2048	2048	1024
nFrames	100	60	100	100	100	100	30
network	LRCGNN	LRCGNN	DFnet	LRCGNN	LRCGNN	DFnet	IGNN
h-mean	60.92%	62.24%	67.73%	73.17%	68.29%	65.07%	86.19%

3.3. AEGAN-AD

In this subsection, we describe the sixth subsystem, which we call AEGAN-AD.

We design an autoencoder which reconstructs Mel spectrograms and complement it with a discriminator, resulting in a GAN [20] model. Inspired by [21], we adopt a DCGAN [22]-like autoencoder, with the discriminator being our encoder and the generator being our decoder. As deconvolution suffers from checker-board effect, yet this effect is somehow resulting from the periodicity of the spectrogram, which makes reconstruction better than an "upsample-conv" structure. BNs are substituted by LNs in order to promote the detection in the target domain. Since most samples in a batch are from the source domain, it is likely that the network is misled by the biased statistics and only learns the distribution of source domain, resulting in a poor performance for the target domain. LN, which normalizes each sample independently, can learn to transform spectrograms into domain-invariant features. As for ToyCar and gearbox, we pass the latent variable through an adaptive LN which does different affine transformations for different sections. This observation indicates that it could help to transform reconstructed samples to their respective styles as in [23] so that features of different sections can be better represented. The loss function is selected as the MSE. Anomaly detection is conducted not only in input space, but also in the latent space, which is done by sending the reconstructed samples back to the encoder to obtain their latent representations. L1 norm, L2 norm and cosine are utilized to measure the difference of each spectrogram and the overall anomaly score is the mean/min/max of these. We select the best performing metric among these metrics.

For gearbox and slider, a discriminator is introduced to promote the reconstruction, while the autoencoder becomes the generator. The discriminator has the similar architecture with the encoder. It is trained to do a feature level discrimination on the reconstructed samples as a complement for MSE loss. Loss function for the discriminator is WGAN-GP [24] and loss function for the generator is a combination of MSE and feature matching loss [25]. Both of them are shown as (5) and (6), respectively.

$$\begin{aligned}\mathcal{L}_D =&\mathbb{E}_{\hat{x}\sim P_g}[D(\hat{x})] - \mathbb{E}_{x\sim P_r}[D(x)]+ \\ &\lambda\mathbb{E}_{\tilde{x}\sim P_{\tilde{x}}}[(\|\nabla_{\tilde{x}}D(\tilde{x})\|_2 - 1)^2]\end{aligned} \tag{5}$$

$$\mathcal{L}_G = \mathbb{E}_{x\sim P_r}[\|x - G(x)\|_2^2] + \mu\|\mathbb{E}_{x\sim P_r}[f(x)] - \mathbb{E}_{\hat{x}\sim P_g}[f(\hat{x})]\|_2^2 \tag{6}$$

Table 9: The results of AEGAN-AD

bearing	fan	gearbox	slider	ToyCar	ToyTrain	valve	h-mean
75.78%	65.83%	71.50%	75.02%	79.16%	58.71%	52.52%	67.04%

where P_r and P_g denote the real distribution and the reconstructed distribution respectively. $P_{\tilde{x}}$ is the linear combination of P_r and P_g. $f(x)$ is the output of the last convolution layer in D. This embedding is also extracted during test time and it is compared with average embedding using k-NN [16], LOF [17], cosine and Mahalanobis distances. We simply choose the best performing metric from both G-based and D-based metrics.

All input for the model is 128×128 Mel spectrogram computed with 2048-point FFT and 512 hop-length. Logarithm is taken first and a MinMaxScaler then scales spectrograms to [-1, 1]. We use an Adam optimizer with a learning rate of 2×10^{-4}. The batch size is set to 512. The model is trained on both development set and evaluation set. The performance is shown in Table 9.

4. SUBMISSION RESULTS

In this subsection, we present the results of ensembles. For the ensembles, we combine the six subsystems by first standardizing the training data scores and then searching over a grid of convex combinations, similar to [2].

The difference between submission-1 and submission-2 is that for submission-1, we additionally train domain classifiers with Cnn6 for sections with obvious domain differences to predict whether the test samples belong to the source domain or the target domain. The test scores for both domains are normalized respectively. For the submission-3, we combine the top performing two or three subsystems for each machine type. For the submission-4, we only combine the SC and WSP-NFCDEE for each machine type. Table 10 shows the results.

Table 10: The results of ensembles

method	bearing	fan	gearbox	slider	ToyCar	ToyTrain	valve	h-mean
submission-1	87.75%	84.98%	87.55%	89.35%	83.52%	78.81%	98.10%	86.81%
submission-2	87.72%	84.34%	87.55%	88.77%	83.48%	78.15%	98.06%	86.51%
submission-3	87.62%	84.73%	87.54%	89.33%	82.25%	77.94%	98.10%	86.40%
submission-4	82.76%	83.88%	84.09%	89.25%	78.81%	76.57%	97.02%	84.18%

5. CONCLUSION

We have outlined our submission to the DCASE 2022 Challenge Task 2, which features a domain shift between the training and test distributions.

In this challenge, we build six subsystems and four ensemble systems in which four new unsupervised models, namely WSP-NFCDEE, IGNN, LRCGNN, and AEGAN-AD are integrated. All these four new models are employed with unsupervised training. Our methods are expected to be promising because they clearly match the data sampling characteristics of practical application scenarios for machine working condition detection in Industry 4.0. Moreover, our best official score of ensembles can achieve 86.81% on the development dataset, which is 30.80% higher than the best baseline. Finally, our method ranks fourth on the evaluation dataset.

6. ACKNOWLEDGEMENT

This work was supported by the National Natural Science Foundation of China under Grant No. U1836219 and No. 62276153.

7. REFERENCES

[1] G. Wilson and D. J. Cook, "A survey of unsupervised deep domain adaptation," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 11, no. 5, pp. 1–46, 2020.

[2] J. A. Lopez, G. Stemmer, P. Lopez Meyer, P. Singh, J. Del Hoyo Ontiveros, and H. Cordourier, "Ensemble of complementary anomaly detectors under domain shifted conditions," in *Proceedings of the 6th Detection and Classification of Acoustic Scenes and Events 2021 Workshop (DCASE2021)*, Barcelona, Spain, November 2021, pp. 11–15.

[3] K. Morita, T. Yano, and K. Tran, "Anomalous sound detection using cnn-based features by self supervised learning," DCASE2021 Challenge, Tech. Rep., July 2021.

[4] R. Giri, S. V. Tenneti, F. Cheng, K. Helwani, U. Isik, and A. Krishnaswamy, "Self-supervised classification for detecting anomalous sounds," in *Detection and Classification of Acoustic Scenes and Events Workshop (DCASE)*, 2020, pp. 46–50.

[5] K. Dohi, T. Nishida, H. Purohit, R. Tanabe, T. Endo, M. Yamamoto, Y. Nikaido, and Y. Kawaguchi, "MIMII DG: Sound dataset for malfunctioning industrial machine investigation and inspection for domain generalization task," *In arXiv e-prints: 2205.13879*, 2022.

[6] N. Harada, D. Niizumi, D. Takeuchi, Y. Ohishi, M. Yasuda, and S. Saito, "ToyADMOS2: Another dataset of miniature-machine operating sounds for anomalous sound detection under domain shift conditions," in *Proceedings of the 6th Detection and Classification of Acoustic Scenes and Events 2021 Workshop (DCASE2021)*, Barcelona, Spain, November 2021, pp. 1–5.

[7] K. Dohi, K. Imoto, N. Harada, D. Niizumi, Y. Koizumi, T. Nishida, H. Purohit, T. Endo, M. Yamamoto, and Y. Kawaguchi, "Description and discussion on DCASE 2022 challenge task 2: Unsupervised anomalous sound detection for machine condition monitoring applying domain generalization techniques," *In arXiv e-prints: 2206.05876*, 2022.

[8] S. Schneider, A. Baevski, R. Collobert, and M. Auli, "wav2vec: Unsupervised pre-training for speech recognition," *arXiv preprint arXiv:1904.05862*, 2019.

[9] J. F. Gemmeke, D. P. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio set: An ontology and human-labeled dataset for audio events," in *2017 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2017, pp. 776–780.

[10] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, "Arcface: Additive angular margin loss for deep face recognition," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 4690–4699.

[11] M. Ott, S. Edunov, A. Baevski, A. Fan, S. Gross, N. Ng, D. Grangier, and M. Auli, "fairseq: A fast, extensible toolkit for sequence modeling," in *Proceedings of NAACL-HLT 2019: Demonstrations*, 2019.

[12] S. Chen, Y. Liu, X. Gao, and Z. Han, "Mobilefacenets: Efficient cnns for accurate real-time face verification on mobile devices," in *Chinese Conference on Biometric Recognition*. Springer, 2018, pp. 428–438.

[13] B. Desplanques, J. Thienpondt, and K. Demuynck, "Ecapa-tdnn: Emphasized channel attention, propagation and aggregation in tdnn based speaker verification," *arXiv preprint arXiv:2005.07143*, 2020.

[14] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7132–7141.

[15] Q. Kong, Y. Cao, T. Iqbal, Y. Wang, W. Wang, and M. D. Plumbley, "Panns: Large-scale pretrained audio neural networks for audio pattern recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 2880–2894, 2020.

[16] S. Ramaswamy, R. Rastogi, and K. Shim, "Efficient algorithms for mining outliers from large data sets," in *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, 2000, pp. 427–438.

[17] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "Lof: identifying density-based local outliers," in *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, 2000, pp. 93–104.

[18] H. Schroter, A. N. Escalante-B, T. Rosenkranz, and A. Maier, "Deepfilternet: A low complexity speech enhancement framework for full-band audio based on deep filtering," in *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022, pp. 7407–7411.

[19] K. Suefusa, T. Nishida, H. Purohit, R. Tanabe, T. Endo, and Y. Kawaguchi, "Anomalous sound detection based on interpolation deep neural network," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 271–275.

[20] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," *Advances in neural information processing systems*, vol. 27, 2014.

[21] S. Akcay, A. Atapour-Abarghouei, and T. P. Breckon, "Ganomaly: Semi-supervised anomaly detection via adversarial training," in *Asian conference on computer vision*. Springer, 2018, pp. 622–637.

[22] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.

[23] X. Huang and S. Belongie, "Arbitrary style transfer in real-time with adaptive instance normalization," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 1501–1510.

[24] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of wasserstein gans," *Advances in neural information processing systems*, vol. 30, 2017.

[25] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," *Advances in neural information processing systems*, vol. 29, 2016.

MIMII DG: SOUND DATASET FOR MALFUNCTIONING INDUSTRIAL MACHINE INVESTIGATION AND INSPECTION FOR DOMAIN GENERALIZATION TASK

Kota Dohi, Tomoya Nishida, Harsh Purohit, Ryo Tanabe, Takashi Endo
Masaaki Yamamoto, Yuki Nikaido, and Yohei Kawaguchi

Research and Development Group, Hitachi, Ltd.
1-280, Higashi-koigakubo, Kokubunji, Tokyo 185-8601, Japan
{kota.dohi.gr, yohei.kawaguchi.xk}@hitachi.com

ABSTRACT

We present a machine sound dataset to benchmark domain generalization techniques for anomalous sound detection (ASD). Domain shifts are differences in data distributions that can degrade the detection performance, and handling them is a major issue for the application of ASD systems. While currently available datasets for ASD tasks assume that occurrences of domain shifts are known, in practice, they can be difficult to detect. To handle such domain shifts, domain generalization techniques that perform well regardless of the domains should be investigated. In this paper, we present the first ASD dataset for the domain generalization techniques, called MIMII DG. The dataset consists of five machine types and three domain shift scenarios for each machine type. The dataset is dedicated to the domain generalization task with features such as multiple different values for parameters that cause domain shifts and introduction of domain shifts that can be difficult to detect, such as shifts in the background noise. Experimental results using two baseline systems indicate that the dataset reproduces domain shift scenarios and is useful for benchmarking domain generalization techniques.

Index Terms— Machine sound dataset, Anomalous sound detection, Unsupervised learning, Domain shift, Domain generalization

1. INTRODUCTION

Anomalous sound detection systems (ASD) are automatic inspection systems that identify anomalous sounds emitted from machines [1–8]. Because these systems use microphones to conduct inspections, contactless inspections of anomalies inside the machines can be realized, unlike the vibration monitoring systems [9–11].

For the widespread application of ASD systems, researchers have mainly tackled two types of challenges. First, in real-world cases, only a few anomalous samples are available or provided anomalous samples do not cover all possible types of anomalies. Therefore, unsupervised anomaly detection methods are often adopted so that the system can detect anomalies by training with only normal samples. MIMII [12] and ToyADMOS [13] are the first datasets that contain machine sounds in real factory environments, and are used for benchmarking the performance of unsupervised ASD methods.

Second, the detection performance of the system degrades due to changes in the distribution of normal sounds (i.e., domain shifts). Domain shifts for an ASD task can be classified into two categories; operational domain shifts caused by changes in states of a machine and environmental domain shifts caused by changes in the background noise or in the recording environment. One solution for handling domain shifts is to use domain adaptation techniques and adapt the model to the new data. MIMII DUE [14] and ToyADMOS2 [15] were developed for benchmarking domain adaptation techniques, while an unsupervised scenario was also assumed.

However, in some real-world cases, domain generalization techniques [16–18] rather than domain adaptation techniques can be preferred. For example, if the operational domain shifts occur too frequently, adaptation of the model can be difficult. This is because only a small amount of data can be used for adaptation and frequent adaptation can be too costly. For another example, if domain shifts are difficult to detect, such as the domain shifts in the background noise, adaptation of the model can also be difficult. In these cases, domain generalization can be useful for handling domain shifts. Because these techniques aim at generalizing the model to detect anomalies regardless of the domains, adaptation of the model during the operation is not necessary. Therefore, domain generalization techniques for ASD task should be investigated for handling domain shifts that are too frequent to adapt or too difficult to detect.

To benchmark the domain generalization techniques for ASD task, a new dataset dedicated to the domain generalization task should be developed. This is because the data required for domain generalization and domain adaptation can be different. For example, generalization of the model may require a larger number of sets of data recorded under different conditions. Also, because domain generalization techniques are likely to be used for domain shifts that can be difficult to detect, this type of shifts should be included in the dataset for domain generalization tasks.

In this paper, we present a new dataset for benchmarking ASD methods using domain generalization techniques. The dataset consists of five different machine types; fan, gearbox, bearing, slide rail, and valve. Each machine type includes three sections, each of which corresponds to a type of domain shift. Each section consists of the source domain data to be used for generalizing the model and the target domain data for evaluating the domain generalization performance. The source domain has at least two different sets of values that cause domain shifts to generalize the model. Also, domain shifts that can only be handled with domain generalization techniques are included in the dataset. The dataset is freely available at https://zenodo.org/record/6529888 and is a subset of the dataset for Task 2 of the DCASE 2022 Challenge.

2. RECORDING ENVIRONMENT AND SETUP

We prepared five types of machines (fan, gearbox, bearing, slide rail, and valve), three types of factory noise data (factory noise A, B, and C), and three different domain shift scenarios for each machine type. The types of machines and domain shift scenarios were chosen on the basis of our experiences building ASD systems for real-world commercial solutions. Here, we identify each scenario of domain shifts by **section IDs**. The details of the type of domain shift for each section and the values of the parameters that shift between domains, the domain shift parameters, are described in Table 1.

We then recorded sound data of each machine to reproduce the domain shift scenarios we assumed. We recorded both normal and anomalous sounds for each domain, where to reproduce anomalous sounds, we used deliberately damaged machines or operated machines in an incorrect manner. For recording, we used a TAMAGO-03 microphone manufactured by *System In Frontier Inc.* [19]. The recording was conducted either in a sound-proof room (Fan and Valve) or in an anechoic chamber (Gearbox, Bearing, Slide rail). Although the microphone has eight channels, we only used the first channel for the dataset. Recorded sound clips are 16-bit audio with a sampling rate of 16 kHz and are 10 seconds long. Examples of spectrograms for each machine type are shown in Figure 1. A short description and recording procedures of each machine type are as follows.

Fan An industrial fan used to keep gas or air flowing in a factory. Operational conditions were kept the same between source and target domains, since Fan was dedicated to environmental domain shifts. Anomaly types include wing damage, unbalanced, clogging, and over voltage.

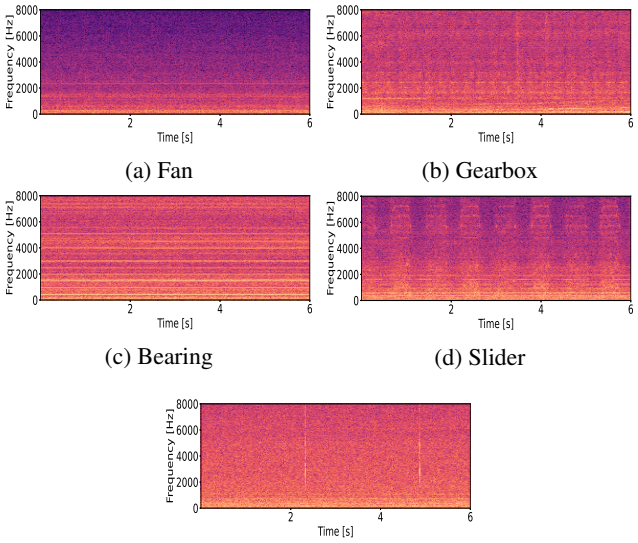
Gearbox A gearbox that links a direct current (DC) motor to a slider-crank mechanism, transmitting the power generated by the rotation of the motor at a constant speed to the slider-crank mechanism. The slider-crank mechanism then converts the rotational motion into a linear motion and raises and lowers its weight. We changed the operation voltage and mass of the weight to cause domain shifts. Anomaly types include gear damage and over voltage.

Bearing Two ball-type bearings are attached to a shaft with a spindle motor, and the sound is emitted from the bearing as it supports the rotating shaft. We changed the rotation speed of the shaft and the location of the microphones to cause domain shifts. Anomaly types include eccentricity in the bearing for two different directions.

Slide rail (slider) A linear slide system consisting of a moving platform and a staging base that repeats a pre-programmed operation pattern. We changed the operation velocity and acceleration to cause domain shifts. Anomaly types include cracks on the rail, removal of grease, and a loose belt for a belt-type slide rail.

Valve A solenoid valve that repeatedly opens and closes in accordance with a pre-programmed operating pattern and is connected to a pump to control air or water flow. We changed the operating pattern and location of the panels surrounding the valve. Anomaly types include contamination in the valve.

After recording the machine sounds, we mixed the prerecorded



(a) Fan (b) Gearbox

(c) Bearing (d) Slider

(d) Valve

Figure 1: Examples of spectrograms for each machine type.

factory noise A, B, or C as the background noise to simulate real-world environments. The factory noise A, B, and C were recorded in different real factories and consisted of sounds of various machinery. The noise-mixed data of each section was generated by the following steps.

1. The average power over all clips in the section, a was calculated.
2. For each clip i from the section,
   (a) the signal-to-noise ratio (SNR) γ dB for the clip was set to the value shown in Table 1,
   (b) a background-noise clip j was randomly selected, and its power b_j was tuned so that $\gamma = 10\log_{10}(a/b_j)$, and
   (c) the noise-mixed data was generated by mixing the machine sound clip i and the power-tuned background-noise clip j.

Here, the background-noise clip j was randomly selected from predetermined types of factory noise, depending on the domain shift scenario. For Fan section 01, Bearing section 02, and Slide rail section 02, factory noise A and B were used for source domain and factory noise C was used for target domain. For other sections, factory noise A and B were used for both source and target domain. Also, for Fan section 00, we additionally mixed sound data of pumps from MIMII DUE.

The complete dataset consists of normal and anomalous operating sounds of five different types of industrial machines, and each machine type has three sections with source and target domain samples. Table 2 lists the number of samples in each section. The training data have 990 source domain samples and ten target domain samples for each section. We prepared ten target domain samples for training data so that the users can utilize a small number of target samples for generalization if the generalization of the model was too difficult. The test data have 50 normal samples and 50 anomalous samples for both domains.

Table 1: Type of domain shift, values of domain shift parameter, and SNR for each section. Values of domain shift parameters represent machines the sound of which are mixed in Fan section 00, levels of noise in Fan section 02, and locations of microphone in Bearing section 01.

Machine type / section ID		SNR [dB]	Type of Domain shift [Domain shift parameter]	Parameter values for source-domain	Parameter values for target-domain
Fan	00	-6.0	Mixing of different machine sound [machine sound index]	W, X	Y, Z
	01	-12.0	Mixing of different factory noise [factory noise index]	A, B	C
	02	N/A	Different levels of noise [noise level (SNR [dB])]	L1 (3), L2 (-9)	L3 (-3), L4 (-15)
Gearbox	00	-6.0	Different operation voltage [V]	1.0, 1.5, 2.0, 2.5, 3.0	0.6, 0.8, 1.3, 1.8, 2.3, 2.3, 3.3, 3.5
	01	-12.0	Different weight attached to the gearbox [g]	0, 50, 100, 150, 200	30, 80, 130, 180, 230, 250
	02	-12.0	Different gearbox ID [machine ID]	05, 08, 13	00, 02, 11
Bearing	00	12.0	Different rotation speed [krpm]	6, 10, 14, 18, 22	2, 4, 8, 12, 16, 20, 24, 26
	01	12.0	Different microphone location [location of the mic.]	A, B, C, D	E, F, G, H
	02	12.0	Mixing of different factory noise [factory noise index]	A, B	C
Slide rail	00	-6.0	Different operation velocity [mm/s]	300, 500, 700, 900, 1100	100, 200, 400, 600, 800, 1000, 1200, 1300
	01	-3.0	Different acceleration $[\mathrm{m/s^2}]$	0.03, 0.05, 0.07, 0.09, 0.11	0.01, 0.02, 0.04, 0.06, 0.08, 0.10, 0.12, 0.14
	02	-12.0	Mixing of different factory noise [factory noise index]	A, B	C
Valve	00	0.0	Different open/close operation patterns [pattern index]	00, 01	02, 03
	01	0.0	Different number and location of panels [panel locations]	open (no panels), bs-c (back-side closed)	b-c (back closed), s-c (side closed)
	02	0.0	Different number of valves [(valve1 pattern index, valve2 pattern index)]	(v1 04), (v1 05), (v2 04), (v2 05)	(v1 04, v2 04), (v1 04, v2 05), (v1 05, v2 04), (v1 05, v2 05)

Table 2: Number of samples in each section

	Source domain		Target domain	
	normal	anomaly	normal	anomaly
Train	990	0	10	0
Test	50	50	50	50

3. RELATION TO MIMII DUE AND TOYADMOS2

While MIMII DUE and ToyADMOS2 were developed for domain adaptation tasks, MIMII DG in this paper is for domain generalization tasks. As described in Sec. 1, the domain generalization techniques are promised for handling domain shifts that domain adaptation techniques may not be applicable. We created a new dataset dedicated to the domain generalization tasks because the dataset for domain generalization tasks and domain adaptation tasks should be different in some points. We included these points as three main features that characterize differences from MIMII DUE and ToyADMOS2.

- The number of values the domain shift parameter (a parameter that causes domain shift) takes has increased to at least three for each type of domain shift. This change is crucial because domain generalization techniques may require multiple sets of data obtained from different domain shift parameter values to generalize the model [20]. For example, for the velocity shift in Slide rail, we increased the number of values of the velocity from four in MIMII DUE to 13 in MIMII DG. Also, with the increased number of sets, users can adjust the difficulty of the generalization task.

- Domain shifts that can be difficult to detect are introduced. As described in Sec. 1, domain generalization techniques are preferred for domain shifts that can be unnoticed. Therefore, we introduced difficult-to-detect domain shifts such as differences in states of a machine operating in the background.

- Domain shift parameters become easier to access and utilize. To generalize the model, not only the sound data but additional information such as the domain shift parameters and other attributes can be useful. Therefore, easy access to these additional information is crucial. Unlike MIMII DUE, we specified domain shift parameters in file names and attribute files for both the source and target domain. With domain shift parameters in the target domain, users can evaluate the detection performance for each value of the domain shift parameter.

4. EXPERIMENT

In this section, we use MIMII DG to benchmark the domain generalization performance of two baseline systems.

4.1. Baseline systems

We used two ASD systems for benchmarking; an autoencoder-based system and a MobileNetV2-based system. These systems are provided as the baseline systems in Task 2 of the DCASE 2022 Challenge, and Python implementations of the systems are available at `https://github.com/Kota-Dohi/dcase2022_task2_baseline_ae` for the autoencoder-based system and `https://github.com/Kota-Dohi/dcase2022_task2_baseline_mobile_net_v2` for the MobileNetV2-based system.

The autoencoder-based system is often used as an unsupervised ASD system. Sound data were first converted to log-Mel spectrogram with a frame size of 1024, a hop size of 512, and 128 Mel bins. Five frames with four overlappings were successively concatenated to generate 640-dimensional input feature vectors. The model had four linear layers with 128 dimensions for the encoder, one bottleneck layer with eight dimensions, and four linear layers with 128 dimensions for the decoder. The model was trained to minimize the error between the input feature vector $\mathbf{x}$ and the reconstruction $\mathbf{x}'$. We trained the model for 100 epochs using the Adam optimizer [21] with a learning rate of 0.0001 and a batch size of 128. The anomaly scores were calculated by the averaged reconstruction error.

The MobileNetV2-based system uses an auxiliary task to improve the detection performance of an unsupervised ASD system [22, 23]. 64 frames with 48 overlappings were successively concatenated to generate input feature vectors. For the model, we used a MobileNetV2 [24] with a multiplier parameter of 0.5. The model was trained to classify section IDs for each machine type. We trained the model for 20 epochs using the Adam optimizer with a learning rate of 0.0001 and a batch size of 128. The anomaly scores were calculated by the averaged negative logit of the predicted probabilities for the correct section.

4.2. Metric

We used the area under the receiver operating characteristic curve (AUC) for evaluation. Because the domain generalization task requires detecting anomalies even when the occurance of domain shifts can be difficult to detect, the anomaly detector is expected to work with the same threshold regardless of the domain. Therefore, we calculated the AUC using both the source and target domain data. Also, to evaluate the anomaly detection performance for each domain, the AUC was computed for each domain. The AUC for each domain, section, and machine type was calculated as

$$\mathrm{AUC} = \frac{1}{N_d^- N_n^+} \sum_{i=1}^{N_d^-} \sum_{j=1}^{N_n^+} \mathcal{H}(\mathcal{A}_\theta(x_j^+) - \mathcal{A}_\theta(x_i^-)), \qquad (1)$$

where n represents the index of a section, $d \in \{\mathrm{source}, \mathrm{target}\}$ represents a domain, and $\mathcal{H}(x)$ returns 1 when $x > 0$ and 0 otherwise. $\mathcal{A}_\theta(x)$ is the anomalous score of a sound clip x, where θ is the parameters of the system. Here, $\{x_i^-\}_{i=1}^{N_d^-}$ is normal test clips in the domain d in the section n and $\{x_j^+\}_{j=1}^{N_n^+}$ is anomalous test clips in the section n in the machine type m. N_d^- is the number of normal test clips in the domain d and N_n^+ is the number of anomalous test clips in the section n.

Table 3: AUC (%) of each domain for each section.

Machine type / section ID		Autoencoder		MobileNetV2	
		source	target	source	target
Fan	00	84.69	39.35	71.07	62.13
	01	71.69	44.74	76.26	35.12
	02	80.54	63.49	67.29	58.02
Gearbox	00	64.63	64.79	63.54	67.02
	01	67.66	58.12	66.68	66.96
	02	75.38	65.57	80.87	43.15
Bearing	00	57.48	63.07	67.85	60.17
	01	71.03	61.04	59.67	64.65
	02	42.34	52.91	61.71	60.55
Slide rail	00	81.92	58.04	87.15	80.77
	01	67.85	50.30	49.66	32.07
	02	86.66	38.78	72.70	32.94
Valve	00	54.24	52.73	75.26	43.60
	01	50.45	53.01	54.78	60.43
	02	51.56	43.84	76.26	78.74
Average		67.21	53.99	68.72	56.42

4.3. Results

Baseline results are shown in Table 3. On average, the AUC for the target domain data was lower than the source domain data at 13.2% with the autoencoder-based system and 12.3% with the MobileNetV2-based system. In some sections, the AUC of the target domain was slightly higher than that of the source domain. This could be because the target domain data happened to be similar to the source domain data of other sections. Overall, the fact that models trained with the source-domain tended to show lower performance for the target data indicate that there is a significant difference between the source-domain data and the target-domain data. This suggests that domain shift scenarios have been successfully reproduced. Thus, the dataset is useful for benchmarking the performance of domain generalization techniques.

5. CONCLUSION

We presented a new dataset, MIMII DG, which was developed for benchmarking domain generalization techniques for ASD. The dataset has normal and anomalous operating sounds of five different types of industrial machines with domain shifts. Experimental results using two ASD systems demonstrate that the detection performance significantly degrades for the target domain.

6. REFERENCES

[1] Y. Koizumi, S. Saito, H. Uematsu, and N. Harada, "Optimizing acoustic feature extractor for anomalous sound detection based on Neyman-Pearson lemma," in *Proc. 25th European Signal Processing Conference (EUSIPCO)*, 2017, pp. 698–702.

[2] Y. Kawaguchi and T. Endo, "How can we detect anomalies from subsampled audio signals?" in *Proc. 27th IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, 2017.

[3] Y. Koizumi, S. Saito, H. Uematsu, Y. Kawachi, and N. Harada, "Unsupervised detection of anomalous sound based on deep learning and the Neyman-Pearson lemma," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 27, no. 1, pp. 212–224, Jan. 2019.

[4] Y. Kawaguchi, R. Tanabe, T. Endo, K. Ichige, and K. Hamada, "Anomaly detection based on an ensemble of dereverberation and anomalous sound extraction," in *Proc. 44th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 865–869.

[5] Y. Koizumi, S. Saito, M. Yamaguchi, S. Murata, and N. Harada, "Batch uniformization for minimizing maximum anomaly score of DNN-based anomaly detection in sounds," in *Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, 2019, pp. 6–10.

[6] K. Suefusa, T. Nishida, H. Purohit, R. Tanabe, T. Endo, and Y. Kawaguchi, "Anomalous sound detection based on interpolation deep neural network," in *Proc. 45th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 271–275.

[7] H. Purohit, R. Tanabe, T. Endo, K. Suefusa, Y. Nikaido, and Y. Kawaguchi, "Deep autoencoding GMM-based unsupervised anomaly detection in acoustic signals and its hyperparameter optimization," in *Proc. 5th Workshop on Detection and Classification of Acoustic Scenes and Events (DCASE)*, 2020, pp. 175–179.

[8] K. Dohi, T. Endo, H. Purohit, R. Tanabe, and Y. Kawaguchi, "Flow-based self-supervised density estimation for anomalous sound detection," in *Proc. 46th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 336–340.

[9] E. P. Carden and P. Fanning, "Vibration based condition monitoring: a review," *Structural health monitoring*, vol. 3, no. 4, pp. 355–377, 2004.

[10] G. Toh and J. Park, "Review of vibration-based structural health monitoring using deep learning," *Applied Sciences*, vol. 10, no. 5, p. 1680, 2020.

[11] A. Khademi, F. Raji, and M. Sadeghi, "Iot enabled vibration monitoring toward smart maintenance," in *2019 3rd International Conference on Internet of Things and Applications (IoT)*, 2019, pp. 1–6.

[12] H. Purohit, R. Tanabe, T. Ichige, T. Endo, Y. Nikaido, K. Suefusa, and Y. Kawaguchi, "MIMII dataset: Sound dataset for malfunctioning industrial machine investigation and inspection," in *Proc. 4th Workshop on Detection and Classification of Acoustic Scenes and Events (DCASE)*, 2019, pp. 209–213.

[13] Y. Koizumi, S. Saito, H. Uematsu, N. Harada, and K. Imoto, "ToyADMOS: A dataset of miniature-machine operating sounds for anomalous sound detection," in *Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, 2019, pp. 313–317.

[14] R. Tanabe, H. Purohit, K. Dohi, T. Endo, Y. Nikaido, T. Nakamura, and Y. Kawaguchi, "MIMII DUE: Sound dataset for malfunctioning industrial machine investigation and inspection with domain shifts due to changes in operational and environmental conditions," in *Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, 2021.

[15] N. Harada, D. Niizumi, D. Takeuchi, Y. Ohishi, M. Yasuda, and S. Saito, "ToyADMOS2: Another dataset of miniature-machine operating sounds for anomalous sound detection under domain shift conditions," in *Workshop on Detection and Classification of Acoustic Scenes and Events (DCASE)*, 2021.

[16] K. Zhou, Z. Liu, Y. Qiao, T. Xiang, and C. Change Loy, "Domain generalization: A survey," *arXiv e-prints*, pp. arXiv–2103, 2021.

[17] Y. Wang, H. Li, and A. C. Kot, "Heterogeneous domain generalization via domain mixup," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 3622–3626.

[18] T. Dissanayake, T. Fernando, S. Denman, H. Ghaemmaghami, S. Sridharan, and C. Fookes, "Domain generalization in biosignal classification," *IEEE Transactions on Biomedical Engineering*, vol. 68, no. 6, pp. 1978–1989, 2021.

[19] System In Frontier Inc. (https://www.sifi.co.jp/product/microphone-array/).

[20] K. Dohi, T. Endo, and Y. Kawaguchi, "Disentangling physical parameters for anomalous sound detection under domain shifts," *arXiv preprint arXiv:2111.06539*, 2021.

[21] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. 3rd International Conference on Learning Representations (ICLR)*, 2015.

[22] R. Giri, S. V. Tenneti, F. Cheng, K. Helwani, U. Isik, and A. Krishnaswamy, "Self-supervised classification for detecting anomalous sounds," in *Proc. 5th Workshop on Detection and Classification of Acoustic Scenes and Events (DCASE)*, 2020, pp. 46–50.

[23] P. Primus, V. Haunschmid, P. Praher, and G. Widmer, "Anomalous sound detection as a simple binary classification problem with careful selection of proxy outlier examples," in *Proc. 5th Workshop on Detection and Classification of Acoustic Scenes and Events (DCASE)*, 2020, pp. 170–174.

[24] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proc. 31st IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 4510–4520.

DESCRIPTION AND DISCUSSION ON DCASE 2022 CHALLENGE TASK 2: UNSUPERVISED ANOMALOUS SOUND DETECTION FOR MACHINE CONDITION MONITORING APPLYING DOMAIN GENERALIZATION TECHNIQUES

Kota Dohi[1]*, Keisuke Imoto*[2]*, Noboru Harada*[3]*, Daisuke Niizumi*[3]*, Yuma Koizumi*[4]*, Tomoya Nishida*[1]*, Harsh Purohit*[1]*, Takashi Endo*[1]*, Masaaki Yamamoto*[1]*, and Yohei Kawaguchi*[1]

[1] Hitachi, Ltd., Japan, kota.dohi.gr@hitachi.com
[2] Doshisha University, Japan, keisuke.imoto@ieee.org
[3] NTT Corporation, Japan, noboru.harada.pv@hco.ntt.co.jp
[4] Google, Japan, koizumiyuma@google.com

ABSTRACT

We present the task description and discussion on the results of the DCASE 2022 Challenge Task 2: "Unsupervised anomalous sound detection (ASD) for machine condition monitoring applying domain generalization techniques". Domain shifts are a critical problem for the application of ASD systems. Because domain shifts can change the acoustic characteristics of data, a model trained in a source domain performs poorly for a target domain. In DCASE 2021 Challenge Task 2, we organized an ASD task for handling domain shifts. In this task, it was assumed that the occurrences of domain shifts are known. However, in practice, the domain of each sample may not be given, and the domain shifts can occur implicitly. In 2022 Task 2, we focus on domain generalization techniques that detects anomalies regardless of the domain shifts. Specifically, the domain of each sample is not given in the test data and only one threshold is allowed for all domains. Analysis of 81 submissions from 31 teams revealed two remarkable types of domain generalization techniques: 1) domain-mixing-based approach that obtains generalized representations and 2) domain-classification-based approach that explicitly or implicitly classifies different domains to improve detection performance for each domain.

Index Terms— anomaly detection, acoustic condition monitoring, domain shift, domain generalization, DCASE Challenge,

1. INTRODUCTION

Anomalous sound detection (ASD) [1–7] is the task of identifying whether the sound emitted from a target machine is normal or anomalous. Automatic detection of mechanical failure is essential in the fourth industrial revolution, which involves artificial intelligence (AI)–based factory automation. Prompt detection of machine anomalies by observing sounds is useful for machine condition monitoring.

One challenge regarding the application scope of ASD systems is that anomalous samples for training can be insufficient both in number and type. In 2020, we organized the fundamental ASD task in Detection and Classification of Acoustic Scenes and Event (DCASE) Challenge 2020 Task 2 [8]; "*unsupervised ASD*" that was aimed to detect unknown anomalous sounds using only normal sound samples as the training data [1–7]. For the wide spread application of ASD systems, advanced tasks such as handling of domain shifts should be tackled. Domain shifts are differences in acoustic characteristics between the source and target domain data caused by differences in a machine's operational conditions or environmental noise. Because these shifts are caused by factors other than anomalies, the detection performance of models trained with the source domain data can degrade for the target domain data. Therefore, in 2021, we organized DCASE Challenge 2021 Task 2 [9], "*unsupervised ASD under domain shifted conditions*" that focused on handling domain shifts using domain adaptation techniques.

The task in 2021 involved the use of domain adaptation techniques under two assumptions. First, all domain shifts have been detected in advance, and the domain of each sample is known. Second, the domain shifts do not occur too frequently for the model to adapt. However, these assumptions may not hold for certain real-world scenarios. For example, a machine's background sound can be affected by various sound sources surrounding the machine, and it can be difficult to identify the cause of changes and attribute the changes to the domain shift. Also, because the operational conditions of the machine can change within a short period, adapting the model every time can be too costly. Therefore, methods have to be investigated such that the detection of domain shifts is unnecessary and frequent occurrences of domain shifts can be handled.

To solve the problem described above, we designed DCASE challenge 2022 Task 2 "*Unsupervised Detection of Anomalous Sounds for Machine Condition Monitoring Applying Domain Generalization Techniques*". This task is aimed at developing domain generalization techniques to handle domain shifts. The task involves the use of domain generalization techniques so that the developed ASD systems do not require detection of the domain shifts or adaptation of the model. Specifically, to evaluate the generalization performance, the domain of each sample is not provided in the test data. To enhance generalization of the model, attributes that caused domain shifts are also provided in the training data.

We received 81 submissions from 31 teams. By analyzing these submissions, we found two types of domain generalization techniques: 1) domain-mixing-based approach and 2) domain-classification-based approach. The domain-mixing-based approach aims at obtaining generalized representations across domains by mixing data from different domains. In contrast, the domain-classification-based approach differentiates different domains so that the model can be specialized for each domain.

2. UNSUPERVISED ANOMALOUS SOUND DETECTION APPLYING DOMAIN GENERALIZATION TECHNIQUES

Let the L-sample time-domain observation $\boldsymbol{x} \in \mathbb{R}^L$ be an audio clip that includes a sound emitted from a machine. The ASD task is a task to determine whether a machine is in a normal or anomalous state using an anomaly score $\mathcal{A}_\theta(\boldsymbol{x})$ calculated by an anomaly score calculator $\mathcal{A} : \mathbb{R}^L \to \mathbb{R}$ with parameters θ. The machine is determined to be anomalous when $\mathcal{A}_\theta(\boldsymbol{x})$ exceeds a pre-defined threshold ϕ as

$$\text{Decision} = \begin{cases} \text{Anomaly} & (\mathcal{A}_\theta(\boldsymbol{x}) > \phi) \\ \text{Normal} & (\text{otherwise}). \end{cases} \quad (1)$$

The primary difficulty in this task is to train $\mathcal{A}$ with only normal sounds. This is because anomalies are rarely obtained in practice.

Domain-shift is another major issue in real-world applications. Domain shifts mean a difference in conditions between training and testing. The conditions are machine's operational conditions such as its speed, load, and temperature, or the environmental conditions such as the type of environmental noise, level of the noise, and location of the microphone. Differences in these conditions change the distribution of data and degrades the detection performance. Let us define two domains: **source domain** and **target domain**, where the source domain is the original condition with enough training clips and the target domain is another condition with zero or a few training clips. Also, let $\mathcal{D}_S$, $\mathcal{D}_T$, $\mathcal{D}_{SA}$, and $\mathcal{D}_{TA}$ be the distributions of $\boldsymbol{x}$ under the normal condition in the source domain, normal condition in the target domain, anomalous condition in the source domain, and anomalous condition in the target domain, respectively.

The task in DCASE 2021 Task 2 involved two tasks. One was to detect anomalies in the source domain: determine whether $\boldsymbol{x_s}$ is from $\mathcal{D}_S$ or $\mathcal{D}_{SA}$ using an anomaly score calculator $\mathcal{A}_{\theta_s}(\boldsymbol{x})$ and a threshold ϕ_s. The other was detection in the target domain: whether $\boldsymbol{x_t}$ is generated from $\mathcal{D}_T$ or $\mathcal{D}_{TA}$ using an anomaly score calculator $\mathcal{A}_{\theta_t}(\boldsymbol{x})$ and a threshold ϕ_t. The task was set to develop domain adaptation techniques so that the detection performance on the target domain can be improved by adaptation on the model trained with the source domain data. Although this problem setting assumes that the domain (source/target) of each sample is known, in practice, the detection of domain shifts can be difficult and the domain may not be available. Also, the use of domain adaptation techniques can be too costly if the domain shifts occur too frequently.

We show four types of real-world scenarios for these problems.

Domain shifts due to differences in machine's conditions
Characteristics of a machine sound can change due to changes in the machine's operational conditions. Although these shifts can be detected, if these conditions change within a short period of time, it can be too costly to adapt the model every time.

Domain shifts due to differences in environmental conditions
Because characteristics of background noise can be affected by various factors, it is difficult to detect these shifts. Therefore, a model that is unaffected by these shifts is desirable.

Domain shifts due to maintenance
Characteristics of a machine sound can change after maintenance or parts replacement. Though these shifts can be detected, adapting the model every time can be costly.

Domain shifts due to differences in recording devices
In real-world scenarios, many microphones are installed at different locations, and these microphones may be from different manufacturers. Although these shifts can be detected, adapting the model for each location or microphone can be too costly.

As a possible solution to handle these problems, domain generalization techniques should be investigated. Domain generalization techniques for ASD aims at detecting anomalies from different domains with a single threshold. These techniques, unlike domain adaptation techniques, do not require detection of domain shifts or adaptation of the model in the testing phase. Therefore, domain generalization techniques can be used for handling domain shifts that are difficult to detect or too costly to adapt.

The DCASE 2022 Task 2 is set to develop domain generalization techniques for ASD. Because the domain generalization techniques are expected to work regardless of the domains, the domain of each sample is not given in the test data. The task is to determine if $\boldsymbol{x}$ is from the normal condition $\mathcal{D}_S \cup \mathcal{D}_T$ or anomalous condition $\mathcal{D}_{SA} \cup \mathcal{D}_{TA}$ using an anomaly score calculator $\mathcal{A}_\theta(\boldsymbol{x})$ and ϕ. Because the differences in operational or environmental conditions make $\mathcal{D}_S \neq \mathcal{D}_T$, the decision must be executed without being affected by the differences between different domains.

3. TASK SETUP

3.1. Dataset

We used ToyADMOS2 [10] and MIMII DG [11] to generate the dataset. The dataset consists of normal/anomalous operating sounds from seven types of toy/real machines (ToyCar, ToyTrain, fan, gearbox, bearing, slide rail, and valve).

Each recording is a single-channel and 10-sec-long audio with a sampling rate of 16 kHz. We mixed machine sounds recorded at laboratories and the environmental noise recorded at real-world factories to create the training/test data. Details of the recording procedure can be found in [10] and [11].

In this dataset, **Machine type** means the type of machine. **Section** is defined as a subset of the data within a machine type and corresponds to a type of domain shift scenario.

We provide three datasets: **development dataset**, **additional training dataset**, and **evaluation dataset**. The **development dataset** consists of three sections (Sections 00, 01, and 02), which are sets of the training and test data. Each section provides (i) 990 normal clips from a source domain for training, (ii) 10 normal clips from a target domain for training, (iii) 100 normal clips and 100 anomalous clips from both domains for the test. We provided domain information (source/target) in the test data for the convenience of participants. Attributes represent the operational or environmental conditions, e.g. velocity of slide rail and level of noise (SNR) mixed in fan data. The **additional training dataset** provides training clips for three sections (Sections 03, 04, and 05). Each section consists of (i) 990 normal clips in a source domain for training and (ii) 10 normal clips in a target domain for training. Attributes are also provided. The **evaluation dataset** provides test clips for three sections (Sections 03, 04, and 05). Each section consists of 200 test clips, none of which have a condition label (i.e., normal or anomaly) or the domain information. Attributes are not provided. The main difference from our task in 2021 is that the domain information is not given in the evaluation dataset. Thus, the participants have to develop a system that performs well regardless of the domains.

3.2. Evaluation metrics

This task is evaluated with the area under the receiver operating characteristic (ROC) curve (AUC) and the partial AUC (pAUC).

The pAUC is calculated as the AUC over a low false-positive-rate (FPR) range $[0, p]$. In this task, we used $p = 0.1$.

Because the domain generalization task requires detecting anomalies using the same threshold between domains, the pAUC has to be calculated for each section, not for each domain. We calculated the AUC for each domain and pAUC for each section as

$$\mathrm{AUC}_{m,n,d} = \frac{1}{N_d^- N_n^+} \sum_{i=1}^{N_d^-} \sum_{j=1}^{N_n^+} \mathcal{H}(\mathcal{A}_\theta(x_j^+) - \mathcal{A}_\theta(x_i^-)), \quad (2)$$

$$\mathrm{pAUC}_{m,n} = \frac{1}{P_n^- N_n^+} \sum_{i=1}^{P_n^-} \sum_{j=1}^{N_n^+} \mathcal{H}(\mathcal{A}_\theta(x_j^+) - \mathcal{A}_\theta(x_i^-)), \quad (3)$$

where $P_n^- = \lfloor p N_n^- \rfloor$, m represents the index of a machine type, n represents the index of a section, $d = \{\mathrm{source}, \mathrm{target}\}$ represents a domain, $\lfloor \cdot \rfloor$ is the flooring function, and $\mathcal{H}(x)$ returns 1 when $x > 0$ and 0 otherwise. Here, $\{\mathcal{A}_\theta(x_i^-)\}$ and $\{\mathcal{A}_\theta(x_j^+)\}$ are sets of anomaly scores of normal and anomalous test clips, ordered in descending power, respectively. N_d^- is the number of normal test clips in domain d, N_n^- and N_n^+ are the number of normal and anomalous test clips in section n, respectively. We calculated $\mathrm{AUC}_{m,n,d}$ to evaluate the contribution of each domain to $\mathrm{AUC}_{m,n}$, as it holds that $\mathrm{AUC}_{m,n} = \sum_d \mathrm{AUC}_{m,n,d}$ if $N_{source}^- = N_{target}^-$.

The official score Ω for ranking submitted systems is given by the harmonic mean of the AUC and pAUC scores over all machine types and sections as follows:

$$\begin{aligned} \Omega \quad = \quad & h\left\{\mathrm{AUC}_{m,n,d},\ \mathrm{pAUC}_{m,n} \quad | \right. \\ & \left. m \in \mathcal{M},\ n \in \mathcal{S}(m),\ d \in \{\mathrm{source}, \mathrm{target}\}\right\}, \end{aligned} \quad (4)$$

where $h\{\cdot\}$ represents the harmonic mean (over all machine types, sections, and domains), $\mathcal{M}$ represents the set of machine types, and $\mathcal{S}(m)$ represents the set of sections for machine type m.

Participants are required to submit the anomaly score and normal/anomaly decision result of each test clip. Even though the official score can be calculated with only the anomaly scores, decision results are also required because we must determine the threshold in real-world applications.

3.3. Baseline systems and results

The task organizers provide an autoencoder (AE)-based and a MobileNetV2-based baseline systems.

The AE-based system calculates the anomaly score as the reconstruction error of the sound. To determine the threshold, we assume that anomaly scores of normal sound follows a gamma distribution. The parameters of the gamma distribution are estimated from the anomaly scores of normal sound in the training data, and the threshold is calculated by the 90th percentile of the gamma distribution. A test clip is determined to be anomalous if its anomaly score exceeds the threshold.

In the MobileNetV2-based system [12–14], classifiers such as the MobileNetV2 [15] are trained to identify from which section the observed signal was generated. The anomaly score is calculated as the averaged negative logit of the predicted probabilities for the correct section. The threshold is calculated in the same manner as in the AE-based baseline.

Tables 1 and 2 show the AUC and pAUC for the two baselines. Because the results produced with a GPU are generally non-deterministic, the average and standard deviations from five independent trials are also shown in the tables.

Table 1: Results of the AE-based baseline

Section		AUC [%]		pAUC [%]
		Source	Target	
ToyCar	00	86.42 ± 1.10	41.48 ± 6.11	51.31 ± 1.34
	01	89.85 ± 1.39	41.93 ± 5.36	54.08 ± 1.84
	02	98.84 ± 0.52	26.50 ± 13.52	52.79 ± 1.04
ToyTrain	00	67.54 ± 0.97	33.68 ± 3.12	52.72 ± 1.63
	01	79.32 ± 0.82	29.87 ± 5.62	50.64 ± 2.33
	02	84.08 ± 0.38	15.52 ± 14.90	48.33 ± 2.33
Bearing	00	67.85 ± 19.61	60.17 ± 7.24	54.41 ± 5.72
	01	59.67 ± 12.67	64.65 ± 12.63	55.09 ± 3.36
	02	61.71 ± 33.52	60.55 ± 35.10	64.18 ± 19.79
Fan	00	84.69 ± 1.74	39.35 ± 9.35	59.95 ± 2.00
	01	71.69 ± 0.69	44.74 ± 1.79	51.12 ± 0.55
	02	80.54 ± 1.42	63.49 ± 2.36	62.88 ± 1.55
Gearbox	00	64.63 ± 0.88	64.79 ± 1.06	60.93 ± 2.31
	01	67.66 ± 0.51	58.12 ± 0.38	53.74 ± 0.56
	02	75.38 ± 0.75	65.57 ± 0.82	61.51 ± 0.69
Slide rail	00	81.92 ± 0.81	58.04 ± 1.22	61.65 ± 1.22
	01	67.85 ± 0.53	50.30 ± 1.25	53.06 ± 0.53
	02	86.66 ± 0.39	38.78 ± 5.13	53.44 ± 1.18
Valve	00	54.24 ± 0.68	52.73 ± 1.93	52.15 ± 0.25
	01	50.45 ± 3.67	53.01 ± 1.73	49.78 ± 0.19
	02	51.56 ± 2.89	43.84 ± 1.11	49.24 ± 0.65

Table 2: Results of the MobileNetV2-based baseline

Section		AUC [%]		pAUC [%]
		Source	Target	
ToyCar	00	47.40 ± 7.22	56.40 ± 4.11	49.96 ± 2.56
	01	62.02 ± 11.07	56.38 ± 11.31	50.92 ± 2.52
	02	74.19 ± 7.94	45.64 ± 11.32	56.51 ± 6.07
ToyTrain	00	46.02 ± 12.21	49.41 ± 15.14	50.25 ± 1.49
	01	71.96 ± 5.72	45.14 ± 13.66	52.97 ± 4.61
	02	63.23 ± 25.60	44.34 ± 21.50	51.54 ± 4.34
Bearing	00	67.85 ± 19.61	60.17 ± 7.24	54.41 ± 5.72
	01	59.67 ± 12.67	64.65 ± 12.63	55.09 ± 3.36
	02	61.71 ± 33.52	60.55 ± 35.10	64.18 ± 19.79
Fan	00	71.07 ± 19.84	62.13 ± 12.50	55.40 ± 11.29
	01	76.26 ± 4.95	35.12 ± 13.38	52.14 ± 4.08
	02	67.29 ± 10.34	58.02 ± 7.46	65.14 ± 1.09
Gearbox	00	63.54 ± 9.46	67.02 ± 13.50	62.12 ± 11.66
	01	66.68 ± 12.29	66.96 ± 8.92	56.85 ± 4.47
	02	80.87 ± 7.85	43.15 ± 16.12	50.62 ± 7.73
Slide rail	00	87.15 ± 2.71	80.77 ± 4.53	71.57 ± 5.28
	01	49.66 ± 30.46	32.07 ± 46.84	48.21 ± 2.73
	02	72.70 ± 11.67	32.94 ± 19.77	49.69 ± 1.63
Valve	00	75.26 ± 4.84	43.60 ± 14.38	55.37 ± 5.86
	01	54.78 ± 5.37	60.43 ± 5.08	54.69 ± 3.87
	02	76.26 ± 1.02	78.74 ± 2.64	85.74 ± 0.08

4. CHALLENGE RESULTS

4.1. Results for evaluation dataset

We received 81 submissions from 31 teams, and 22 teams outperformed the MobileNetV2-based baseline in the official score. In Figure 1, the harmonic means of the AUCs are shown for top 10 teams [16–25]. Although the AUCs change drastically between different machine types and teams, these highly ranked teams outperformed the baselines for most of the machine types. It is worth noting that, for these teams, the source-domain AUC did not correlate with the official rank (correlation coefficient was -0.033) while the target-domain AUC did (correlation coefficient was -0.862). This indicates that handling domain shifts and generalizing the model was the key to better ranks among highly ranked teams.

We find that domain generalization approaches adopted by the participants can be categorized into two types: domain-mixing-based approach and domain-classification-based approach. These methods achieved the aim of the task by generalizing the model using data with different attributes. Figure 2 shows the average source-domain and target-domain AUC of the top 20 teams.

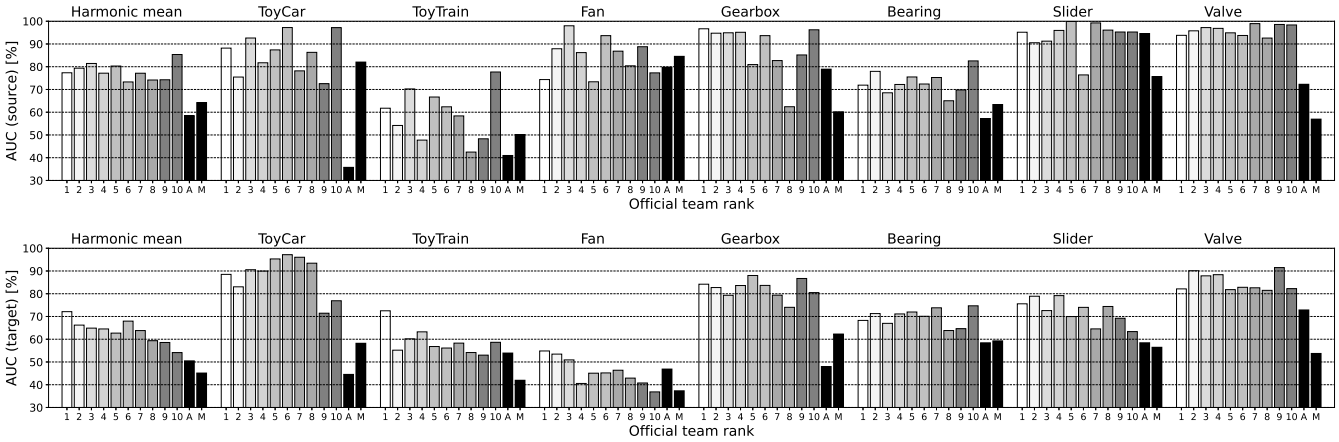


Figure 1: Evaluation results of top 10 teams in the ranking. Average source-domain AUC (Top) and target-domain AUC (bottom) for each machine type. Label "A" and "M" on the x-axis denote AE-based and MobileNetV2-based baselines, respectively.

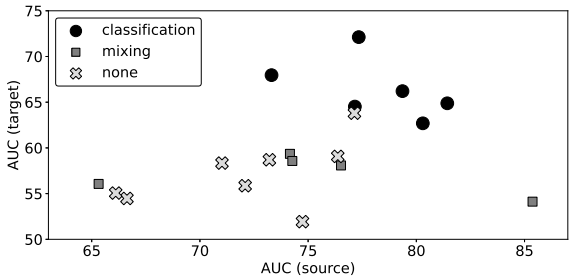


Figure 2: Average source-domain AUC and target-domain AUC of the top 20 teams. "classification" denotes teams that used domain-classification-based approaches, "mix-up" denotes teams that used domain-mixing-based approaches, and "none" denotes teams that did not use particular domain generalization techniques.

Domain-classification-based approaches outperformed other approaches especially for the target domain. However, these approaches may be specialized for the types of target domain data provided in both the training and test data, and thus may not perform well for those not included in the train data. We describe the details in the following.

4.2. Domain-mixing-based approach

Domain-mixing-based approach extracts common representations between domains. These include batch mixing that use data from both domains in a batch to train a model [17, 23], Mixup [26] that synthesizes data from both domains to obtain intermediate representations [17, 23, 25, 27], and data augmentation techniques to obtain robust representations [24]. These techniques use the target domain data to expand the normal conditions for the model so that the model can be generalized to better handle domain shifts. However, as shown in Figure 2, they have been outperformed by domain-classification-based approaches. This can be that, for the Mixup and data augmentation, synthesized data was not useful for representing the target domain data. One future direction can be on obtaining meaningful synthetic representations with the aid of external information such as the attribute information.

4.3. Domain-classification-based approach

Domain-classification-based approach distinguishes the source and target domain data to obtain better detection performance for each domain. The 1st and 6th place teams [16, 21] used distances between the embedding of a domain and that of the test data to calculate anomaly scores. Because the domain of each sample can be estimated by the domain with shorter distance, this approach can be regarded as implicitly classifying the domain of each sample. The 2nd, 3rd, 4th, and 5th place teams [17–20] explicitly trained a classifier to distinguish the attributes or the domains. The 5th place teams trained an attribute classifier and a section classifier so that both the domain-wise information from the attribute classifier and the domain-independent information from the section classifier can be obtained.

As shown in Figure 2, the domain-classification-based approach outperformed the domain-mixing-based approach. This can be because the normal conditions are defined for each specific domain, unlike the domain-mixing-based approach that defines normal conditions over all domains. However, this approach assumes that the target domain data in the train data includes all types of the target domain data in the test data. If the target domain data in the test data contains too many types of data not included in the train data, the classifier may fail to distinguish domains, which can degrade the detection performance. Therefore, further investigation is needed to examine the ability of this approach to handle completely unseen target domain data.

5. CONCLUSION

This paper presented an overview of the task and analysis of the solutions submitted to DCASE 2022 Challenge Task 2. To handle domain shifts that occur implicitly, the task was dedicated to developing domain generalization techniques. The organization of the task revealed two approaches that can be useful for domain generalization task: domain-mixing-based approach and domain-classification-based approach. For the former approach, obtaining more meaningful synthetic representations from multiple domains is left for future works. For the latter approach, future works can focus on analyzing the effect of this approach on completely unseen types of target domain data.

6. REFERENCES

[1] Y. Koizumi, S. Saito, H. Uematsu, and N. Harada, "Optimizing acoustic feature extractor for anomalous sound detection based on Neyman-Pearson lemma," in *Proc. 25th European Signal Processing Conference (EUSIPCO)*, 2017, pp. 698–702.

[2] Y. Kawaguchi and T. Endo, "How can we detect anomalies from subsampled audio signals?" in *Proc. 27th IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, 2017.

[3] Y. Koizumi, S. Saito, H. Uematsu, Y. Kawachi, and N. Harada, "Unsupervised detection of anomalous sound based on deep learning and the Neyman-Pearson lemma," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 27, no. 1, pp. 212–224, Jan. 2019.

[4] Y. Kawaguchi, R. Tanabe, T. Endo, K. Ichige, and K. Hamada, "Anomaly detection based on an ensemble of dereverberation and anomalous sound extraction," in *Proc. 44th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 865–869.

[5] Y. Koizumi, S. Saito, M. Yamaguchi, S. Murata, and N. Harada, "Batch uniformization for minimizing maximum anomaly score of DNN-based anomaly detection in sounds," in *Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, 2019, pp. 6–10.

[6] K. Suefusa, T. Nishida, H. Purohit, R. Tanabe, T. Endo, and Y. Kawaguchi, "Anomalous sound detection based on interpolation deep neural network," in *Proc. 45th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 271–275.

[7] H. Purohit, R. Tanabe, T. Endo, K. Suefusa, Y. Nikaido, and Y. Kawaguchi, "Deep autoencoding GMM-based unsupervised anomaly detection in acoustic signals and its hyper-parameter optimization," in *Proc. 5th Workshop on Detection and Classification of Acoustic Scenes and Events (DCASE)*, 2020, pp. 175–179.

[8] Y. Koizumi, Y. Kawaguchi, K. Imoto, T. Nakamura, Y. Nikaido, R. Tanabe, H. Purohit, K. Suefusa, T. Endo, M. Yasuda, and N. Harada, "Description and discussion on DCASE2020 challenge task2: Unsupervised anomalous sound detection for machine condition monitoring," in *Proc. 5th Workshop on Detection and Classification of Acoustic Scenes and Events (DCASE)*, 2020, pp. 81–85.

[9] Y. Kawaguchi, K. Imoto, Y. Koizumi, N. Harada, D. Niizumi, K. Dohi, R. Tanabe, H. Purohit, and T. Endo, "Description and discussion on DCASE 2021 challenge task 2: Unsupervised anomalous detection for machine condition monitoring under domain shifted conditions," in *Proc. 6th Workshop on Detection and Classification of Acoustic Scenes and Events (DCASE)*, 2021, pp. 186–190.

[10] N. Harada, D. Niizumi, D. Takeuchi, Y. Ohishi, M. Yasuda, and S. Saito, "Toyadmos2: Another dataset of miniature-machine operating sounds for anomalous sound detection under domain shift conditions," in *Proc. 6th Workshop on Detection and Classification of Acoustic Scenes and Events (DCASE)*, 2021, pp. 1–5.

[11] K. Dohi, T. Nishida, H. Purohit, R. Tanabe, T. Endo, M. Yamamoto, Y. Nikaido, and Y. Kawaguchi, "MIMII DG: Sound dataset for malfunctioning industrial machine investigation and inspection for domain generalization task," *arXiv preprint arXiv:2205.13879*, 2022.

[12] R. Giri, S. V. Tenneti, F. Cheng, K. Helwani, U. Isik, and A. Krishnaswamy, "Self-supervised classification for detecting anomalous sounds," in *Proc. 5th Workshop on Detection and Classification of Acoustic Scenes and Events (DCASE)*, 2020, pp. 46–50.

[13] P. Primus, V. Haunschmid, P. Praher, and G. Widmer, "Anomalous sound detection as a simple binary classification problem with careful selection of proxy outlier examples," in *Proc. 5th Workshop on Detection and Classification of Acoustic Scenes and Events (DCASE)*, 2020, pp. 170–174.

[14] T. Inoue, P. Vinayavekhin, S. Morikuni, S. Wang, T. H. Trong, D. Wood, M. Tatsubori, and R. Tachibana, "Detection of anomalous sounds for machine condition monitoring using classification confidence," in *Proc. 5th Workshop on Detection and Classification of Acoustic Scenes and Events (DCASE)*, 2020, pp. 66–70.

[15] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proc. 31st IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 4510–4520.

[16] Y. Zeng, H. Liu, L. Xu, Y. Zhou, and L. Gan, "Robust anomaly sound detection framework for machine condition monitoring," DCASE2022 Challenge, Tech. Rep., 2022.

[17] I. Kuroyanagi, T. Hayashi, K. Takeda, and T. Toda, "Two-stage anomalous sound detection systems using domain generalization and specialization techniques," DCASE2022 Challenge, Tech. Rep., 2022.

[18] F. Xiao, Y. Liu, Y. Wei, J. Guan, Q. Zhu, T. Zheng, and J. Han, "The dcase2022 challenge task 2 system: Anomalous sound detection with self-supervised attribute classification and gmm-based clustering," DCASE2022 Challenge, Tech. Rep., 2022.

[19] Y. Deng, J. Liu, and W.-Q. Zhang, "Aithu system for unsupervised anomalous detection of machine working status via sounding," DCASE2022 Challenge, Tech. Rep., 2022.

[20] S. Venkatesh, G. Wichern, A. Subramanian, and J. Le Roux, "Disentangled surrogate task learning for improved domain generalization in unsupervised anomalous sound detection," DCASE2022 Challenge, Tech. Rep., 2022.

[21] Y. Wei, J. Guan, H. Lan, and W. Wang, "Anomalous sound detection system with self-challenge and metric evaluation for dcase2022 challenge task 2," DCASE2022 Challenge, Tech. Rep., 2022.

[22] K. Morita, T. Yano, and K. Tran, "Comparative experiments on spectrogram representation for anomalous sound detection," DCASE2022 Challenge, Tech. Rep., 2022.

[23] J. Bai, Y. Jia, and S. Huang, "Jless submission to dcase2022 task2: Batch mixing strategy based method with anomaly detector for anomalous sound detection," DCASE2022 Challenge, Tech. Rep., 2022.

[24] S. Verbitskiy, M. Shkhanukova, and V. Vyshegorodtsev, "Unsupervised anomalous sound detection using multiple time-frequency representations," DCASE2022 Challenge, Tech. Rep., 2022.

[25] K. Wilkinghoff, "An outlier exposed anomalous sound detection system for domain generalization in machine condition monitoring," DCASE2022 Challenge, Tech. Rep., 2022.

[26] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," in *International Conference on Learning Representations*, 2018. [Online]. Available: https://openreview.net/forum?id=r1Ddp1-Rb

[27] I. Nejjar, J. P. J. Meunier-Pion, G. M. Frusque, and O. Fink, "Dcase challenge 2022: Self-supervised learning pre-training, training for unsupervised anomalous sound detection," DCASE2022 Challenge, Tech. Rep., 2022.

CONVOLUTIONAL NEURAL NETWORK FOR AUDIBILITY ASSESSMENT OF ACOUSTIC ALARMS

François Effa[1,2,3]*, Romain Serizel*[3]*, Jean-Pierre Arz*[1]*, Nicolas Grimault*[2]*,*

[1] Institut National de Recherche et de Sécurité, F-54000 Nancy, France
[2] Centre de Recherche en Neurosciences de Lyon, CNRS, F-69500 Bron, France
[3] Université de Lorraine, CNRS, Inria, Loria, F-54000 Nancy, France

ABSTRACT

In noisy workplaces, the audibility of acoustic alarms is essential to ensure worker safety. In practice, some criteria are required in international standards to make sure that the alarms are "clearly audible". However, the recommendations may lead to overly loud alarms, thereby exposing workers to unnecessary high sound levels, especially when ambient sound levels are high themselves. For this reason, it appears necessary to properly assess the audibility of alarms at design stage. Existing psychoacoustical methods rely on repeated subjective measurements at different sound levels and therefore require time-consuming procedures. In addition, they must be repeated each time the alarm or sound environment changes. To overcome this issue, we propose a data-driven approach to estimate the audibility of new alarm signals without having to test each new condition experimentally. In this study, a convolutional neural network model is trained to perform a binary classification task on short sound clips labeled with the outcomes of psychoacoustical experiments. We propose a proof of concept of this approach and analyze its performance depending on the data used at training and the temporal context used by the networks to predict the audibility of the alarm.

***Index Terms*—** Acoustic Scene Classification, Warning signals, Psychophysics, Machine learning

1. INTRODUCTION

The audibility of acoustic signals indicating a danger – for instance, the reverse alarm of a construction machine – is essential to ensure the safety of workers and to prevent the risk of accidents. The international standard dedicated to auditory danger signals for public and work areas requires for the alarms to be "clearly audible" [1]. In order to meet this requirement, several criteria related to the ambient noise levels are proposed. In particular an overall level criterion imposes a minimum signal-to-noise ratio (SNR) of 15 dB. This recommendation is questionable on two counts. First, the proposed criteria appear to lead in practice to excessive sound levels when the ambient noise levels are high. Żera and Nagórski have found that, when asking listeners to adjust the level of alarms so that they are judged to be clearly audible, the required SNR varies continuously from 15 to −2 dB as the noise level increases from 60 dB to 90 dB [2]. An SNR of 15 dB could therefore be too conservative or even harmful at high noise levels. Second, there is no scientific or formal definition of what "clearly audible" means. We can easily understand the difference between a sound that is simply detectable and another, qualified as clearly audible. The alarm must not only be perceptible but also "loud" enough to provide an effective warning of danger. This judgment can however vary greatly from one individual to another. It should also be noted that such consideration is dependent on the sound environment in which the alarm occurs.

In the presence of background noise, our ability to detect a given sound is reduced. It translates into an increase in the audibility threshold of the target sound, meaning that the level at which the sound is just audible is higher in noise than in silence [3, 4]. The noise in question is called *masker*, and we refer to the audibility threshold when a masker is present as the *masked threshold*. The masking mechanism is the basis for current alarm design methods. With the understanding that the alarm level must be well above the masked threshold to ensure reliable audibility, it has been suggested that alarms should exceed the masked threshold by 10 to 15 dB [5]. This recommendation is also included in the standard [1]. Masked thresholds can be measured experimentally [6], but such approach is not the most convenient, since a psychoacoustical experiment requires the involvement of multiple human subjects, and measuring an audibility threshold implies covering a range of different sound levels, which can be time-consuming. Furthermore, the measurements strongly depend on the acoustic properties of the sounds. Consequently, the experiment must be repeated for any new condition of interest such as a new alarm signal, or different ambient noise type or level. In response to this problem, some models have been developed to predict masked thresholds [7, 8]. However, these are based on the explicit estimation of the masked thresholds, and therefore only efficient in well-controlled conditions.

The main motivation of our work is to propose a model capable of accurately estimating the audibility of acoustic alarms in noise. We intend this model to be applicable to a large variety of sound environments, including fluctuating noises and different types of warning signals. In that regard, we suggest that a deep learning approach, which is data-driven, would be suitable. A few studies have already begun to pave the way for connecting psychophysics and deep learning methods, for instance, by implementing a psychophysically inspired methodology for model evaluation [9]. More recently, in the context of handwritten document transcription, a loss function was reformulated to account for perceptual data [10]. The authors used it to train a CNN to perform character recognition and obtained consistent and repeatable performance improvement on standard datasets. In the field of audition, recent works have replicated human perceptual judgments with a high level of likeness in word recognition, genre recognition [11] and sound localization in natural environments [12].

In this paper, we propose to adapt methods used in ambient sound analysis [13, 14] to the specific topic of audibility assessment of auditory alarms. To that end, we introduce a new dataset consist-

ing of sound clips made up of auditory warning signals embedded in noise labelled through psychoacoustical experiments. Based on this dataset, we train a model to reproduce human judgment regarding the audibility of the alarms.

2. METHOD

Ambient sound analysis is a broad field of research that encompasses several sub-areas such as Acoustic Scene Classification (ASC) [15], Sound Event Dectection (SED) [16, 17], or audio tagging [18]. Although our approach does not fully fit into any of these categories, it does share certain aspects with some of them. First, similarly to what is done in ASC and audio tagging, we want the model to produce file-level estimations. Besides, the analysis focuses on the alarms, which are temporally localized sound events. This is pretty similar to SED or tagging. Despite this, we are not directly interested in the ability of the model to detect the alarm. Yet, if the model can predict that an alarm is audible, it gives an indication that the model did initially detect it. The opposite, however, is not true. It is quite possible that the model predicts that an alarm is not audible although it has detected the alarm itself.

To assess whether a particular alarm can be considered clearly audible in a given environment, we propose to frame the problem as a binary classification task. The proposed method uses 5.5-second audio clips containing auditory warning signals embedded in background noises as input. After extracting acoustical features from these signals, we used a CNN to produce a binary estimate of whether the alarm present in the clip is clearly audible or not. More details about the audio clip generation and the annotation process are provided in the next section.

3. PSYCHOACOUSTICAL DATA COLLECTION

The need for relatively large amounts of data with a sufficient variability is a well-known constraint associated with deep learning. To develop our model with psychoacoustical data, no ready-to-use dataset was available from external sources. Therefore, we had to collect one. To do this, our choices were guided by two major considerations. First, we had to find a way to collect the maximum amount of data at the lowest time cost. Second, we still wanted to keep the possibility of explaining all or part of our results within the psychoacoustical framework. We decided to divide the psychoacoustical experiments in three parts. The first part is following within-subject design, closer to psychoacoustical procedures. The second and third parts follow lighter procedures and lower data collection time cost. In this section, we describe the psychoacoustical experiments that have been carried out to collect the dataset. The dataset in this paper is a preliminary version used for a proof of concept. The final dataset is expected to be larger.

3.1. Stimuli and material

Stimuli were made of short alarm sounds (between 0.243 and 1.763 s long) embedded in background noises. The alarms were mostly synthetic signals, but some of them were clean recordings. Backgrounds were field recordings, taken to be industry-related (factory, roadworks, construction) or captured in noisy public spaces. Both alarms and noises were mono signals collected from different sources, mainly the Freesound database [19], BigSoundBank [20], and to a lesser extent, a published set of medical alarms [21] or self-recorded railway warning signals [22].

In the experiments, 5.5-second clips were generated, each containing a background noise and a single alarm with a random onset temporal location. The level of the noise varied between 60 dBA and 80 dBA. The SNRs were all taken between −30 dB and 15 dB. The stimuli were played at a sample rate of 44.1 kHz using an RME Babyface Pro sound card, and presented over Beyerdynamic DT 770 Pro headphones calibrated with Larson Davis AEC101 artificial ear and Model 824 sonometer.

3.2. Procedure and datasets

Twelve volunteers aged from 18 to 43 and free of reported hearing problems took part in this study. They came for multiple sessions of one or two hours each. All the participants were compensated for the time spent on the experiments. To evaluate the audibility, the subjects were presented with a clip made up of an alarm embedded in a background noise. At the end of the presentation, they had to answer the question "*Was the alarm clearly audible?*" by simply clicking *Yes* or *No*. Three different experiments were carried out and most of the subjects took part in each of the three experiments. Each experiment served to collect a separate subset designed for a specific purpose.

The first set consists of 6 audio clips, each declined in 2 different noise levels (60 dBA and 80 dBA) and 10 SNRs linearly spaced between −30 and 15 dB, making a total of 120 signals. In each of the 6 clips, the alarm signals and backgrounds are different from the other clips. All these stimuli have been annotated once by the 10 participants. As a result, it contains psychoacoustical data that are quite close to what would have been obtained through a standard procedure, yet it is often recommended to make more than one repetition per subject [1]. This set is the most controlled set and is selected as the evaluation set. The data collection process on this subset will eventually allow for further comparison with more standard psycho-acoustic experiments. The labels (0 or 1) have been obtained by setting a 0.5 threshold on the proportion of "*Yes*" across participant answers. This subset is referred to as *subA* in the remainder of the paper.

The second set contains 1800 audio clips, made with the same noises and alarms as in the first set, except that the 6 formerly used alarm-noise combinations were avoided. As a consequence, there was a total of 30 possible alarm-noise combinations. Six different SNRs ([−25, −10, −5, 0, 5, 12.5] in dB) and six noise levels ([60, 64, 68, 72, 76, 80] in dBA) were used and uniformly distributed among the clips. Each of the 1800 clips had a unique alarm onset location. Ten participants were involved in this experiment. Each of them listened to 180 clips. The distribution of the clips among subjects was done in such a way that each subject listened to the same number of clips per SNR and per noise level. No repetition was made across subjects, meaning each stimulus has been annotated just once. This subset is referred to as *subB*.

The data eventually collected in these experiments will be used to form a large training subset. However, the third set introduced in this paper is also composed of 1800 different clips. This is motivated by the desire to keep comparable size between this set and *subB*. There are 70 alarms and 52 background noises in this set, all different from the 6 used in the first and second sets. Two noise levels were used (60 dBA and 80 dBA). There were 46 different SNRs ranging from −30 to 15 dB with a step of 1 dB. Ten subjects contributed to the annotation. Some of the conditions have been randomly repeated among subjects, making a total of around 11500 annotation points. For clips with a single annotation point,

the subject's answer was kept as the final annotation. For clips with multiple annotation points, the labels were derived by setting a 0.5 on the *Yes*-rate. We refer to this subset as *subC*.

Subsets *subB* and *subC* were used separately for development, using 1440 training clips and 360 validation clips, which corresponds to a 80%/20% ratio. The training/validation split was performed randomly and kept fixed for all the experiments.

4. EXPERIMENTAL SETUP

4.1. Acoustic features

The signals were sampled at 44.1 kHz. The features we used are mel-spectrograms with 64 coefficients. They were extracted using a 1024-sample short-time Fourier transform (STFT), with 50% overlap and a Hamming window.

We did explore the idea of using more perceptually relevant representations such as cochleagrams [12] or spectro-temporal excitation patterns [23]. However, preliminary experiments with these features did not provide any significant performance improvement in terms of accuracy on the development sets. Therefore, in this paper we use only mel-spectrograms as input features.

For the experiments, the input representations were standardized to zero mean and unit variance along mel frequency bins. The standardization coefficients were computed over the whole training set.

4.2. Convolutional Neural Network

The architecture of the CNN used in this paper is inspired by models used in SED [14] and Bird Audio Detection [13]. The model is composed of 4 convolutional layers with [32, 64, 64, 128] filters per layer. Each filter has a 3-by-3 receptive field. Each convolutional layer is followed by ReLU activations and max pooling along the frequency axis ([1, 4], [1, 4], [1, 2] and [1, 2], respectively). The activation outputs from the last convolutional layer are stacked along frequency axis [13]. Preliminary experiments with recurrent layers did not lead to significant improvement. Therefore they are not used in this paper. Instead, we directly operate L_p aggregation over the time axis on the stacked representations. L_p aggregation with $p = 2$ was preferred over max pooling which is not differentiable and may lead to instability [24]. In addition, it has been shown to be more robust to variations in the relative duration of the alarm compared to the clip length [25]. The aggregation layer is followed by the classification layer that has one single neuron with sigmoid activation. The neuron is intended to produce an activation which is close to 1 when the alarm present in the clip is clearly audible, and close to 0 when the alarm is not clearly audible.

For training, back-propagation was performed using a binary cross-entropy loss function and Adam optimizer [26] with a learning rate of 0.0001. To reduce overfitting, dropout was applied on the outputs of all the convolutional layers with a rate of 0.25 and regularization was employed by fixing a 0.0001 weight decay in Adam. The model was trained for a maximum of 250 epochs and the epoch giving the best accuracy on validation set was kept.

4.3. Model evaluation

For the experiments, the model was evaluated from the area under the receiver operating characteristic curve (AUC) and the F1-score. We trained the models with 10 randomized initializations. The metrics were computed on the outputs obtained with these 10 models. We report the mean and 95% confidence intervals of the metrics.

5. RESULTS AND DISCUSSION

In this section, we report the two series of experiments that have been conducted on the model. The first series of experiments focuses on the model performance depending on the data used at training. The second series of experiments assesses the potential effects of the temporal context used in the model to predict the audibility.

5.1. Impact of the training data

Our first series of experiments investigates the performance of the models trained on *subB* and *subC*. As described in Section 3.2, the clips in *subA* and *subB* were made with the same alarms and background noises. For this reason, we expect the model to perform better when it is trained on *subB* than on *subC*.

At first, the model was trained on *subB* or *subC* data while *subB* validation data were used to select the model. Table 1 shows the AUC and F1-score on development and test sets. As we can observe, performance on the test set is better when *subB* is used for training. There are two potential causes for this. The difference in performance can be due to the fact that the alarm signals and backgrounds in *subC* are different from those in *subB* and in the test set or to the fact the task addressed in *subC* is more difficult than in *subB* (or both). We evaluated models trained on *subC* and validated on either *subB* or *subC* to verify this second hypothesis (see Table 2). The results show a significant difference in performance on development set depending on whether the model was validated on *subB* or *subC*. The high development score when *subC* is used for validation suggests that the model can be fitted to *subC* data, which indicates that the task addressed in *subC* is in fact not more difficult than in *subB*. However, the performance on test set shows that the model gives better results on test data when the alarms and background noises have been seen during training. This raises the question whether collecting more training data with a larger set of alarms and backgrounds can help to compensate for this performance gap. If not, it would induce the need to see test alarms or test backgrounds or both during training. In practice, such a scenario would not be realistic. Despite the difference in performance, it should be noted that when training on different alarms and backgrounds, the model performance does not actually collapse.

Subset		Development score	Test score
subB	AUC	94.4 ± 0.3	95.3 ± 0.7
	F1	91.9 ± 0.3	89.2 ± 1.6
subC	AUC	78.9 ± 1.0	87.9 ± 1.9
	F1	79.1 ± 1.2	79.3 ± 2.4

Table 1: AUC and F1-scores on development and evaluation sets with 95% confidence intervals. *subB* is used for validation.

5.2. Impact of the clip duration

The alarms that are present in the different clips have variable lengths. Since the longest alarm is less than 1.8 s long, all the 5.5-second clips contain both portions of noise alone and the whole section where the alarm occurs. It is still to be determined whether the model bases its predictions on the entire alarm or on a specific region of the alarm. This region could be the onset, for instance.

Validation set		Development score	Test score
subB	AUC	78.9 ± 1.0	87.9 ± 1.9
	F1	79.1 ± 1.2	79.3 ± 2.4
subC	AUC	94.4 ± 0.4	88.3 ± 1.8
	F1	87.2 ± 0.4	79.5 ± 2.0

Table 2: Performance of the model trained on *subC* with either *subB* or *subC* used for validation.

Moreover, it is not sure whether it also relies on the information present in the parts of the signal where there is no alarm. In this second series of experiments, we are interested in observing how the model uses the temporal context to produce estimations of the audibility of the alarms.

The model was trained on *subB* since it resulted to better performance in the previous experiment. We varied the duration of the clips used to train the model. Four different durations were experimented: 5.5, 1.0, 0.5, and 0.1 seconds. For this, each input representation was shortened to the desired length around the alarm position. Every time, the model was tested on *subA* using all four clip lengths. The results are reported in Table 3.

As a first observation, when tested with the same clip length as the one used for training, the model shows relatively good performance. This result is true whatever the clip length. However, the model is only able to perform well for all test clip durations when it is trained on 5.5-second clips. This result suggests that the model needs temporal context at training time but not necessarily to make predictions at inference time. Finally, when 5.5-second clips are used for training, the performance of the model weakens slightly as the duration of the test clips is reduced, though it is still quite high. It would therefore be reasonable to train on long clips if the model is then to make predictions over shorter time periods. These observations are based on the AUC. The-F1 score shows some unexplained effects such as a lower value observed when the model is trained on 5.5-second clips and tested on 1-second clips. The investigation of these effects may require a more detailed analysis.

Training \ Test		0.1	0.5	1.0	5.5
0.1	AUC	88.6 ± 1.4	53.2 ± 11.1	51.8 ± 11.9	50.1 ± 7.5
	F1	81.8 ± 1.6	36.0 ± 19.5	32.9 ± 20.6	32.2 ± 21.0
0.5	AUC	53.3 ± 13.2	92.1 ± 1.5	55.5 ± 14.9	50.1 ± 6.7
	F1	44.4 ± 17.7	85.0 ± 1.3	41.9 ± 18.7	33.8 ± 20.2
1.0	AUC	43.9 ± 15.2	53.5 ± 13.5	89.5 ± 1.8	47.7 ± 6.0
	F1	36.7 ± 15.5	45.5 ± 14.7	81.6 ± 1.6	40.6 ± 14.3
5.5	AUC	88.8 ± 6.0	92.3 ± 1.9	93.9 ± 1.6	95.0 ± 1.0
	F1	75.3 ± 6.5	80.0 ± 5.7	75.7 ± 4.2	87.0 ± 1.2

Table 3: Performance on test set depending on the clip length used for training and evaluation.

5.3. The model's output as a psychometric value

As previously mentioned, psychoacoustical experiments are usually conducted in a repeated measures design. For example, in order to evaluate the audibility of an alarm through a *Yes-No* task, a common approach consists in presenting same clip once or several times to every participant. With such procedure, we can measure the proportion of *Yes* responses over all trials. Then by varying a given attribute of the stimulus, it is possible to establish a relationship between this specific attribute and the subjects' responses. Such a relationship is called a psychometric function.

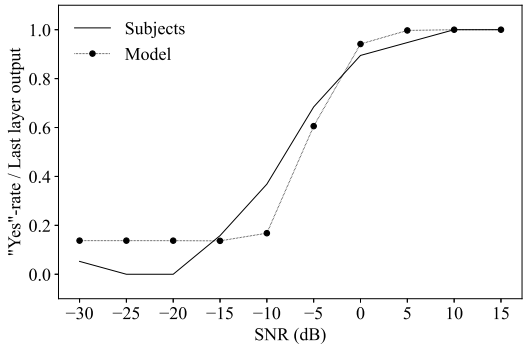


Figure 1: Psychometric function of a clip from *subA*. The rate of positive responses averaged across all participants as a function of the SNR is represented by the plain curve. The dotted curve with round markers shows the values taken by the last neuron.

For instance, consider the procedure described in Section 3.2 to collect data for *subA*. Different clips were generated from the 6 initial clips by varying the SNR and the noise level. By taking a given clip at a single ambient noise level, we can represent the evolution of the proportion of *Yes* responses as the SNR increases. This approach is quite different from what we do when we train a CNN to perform a binary classification task. Indeed, we use binary labels for training. This means that the model is trained to produce outputs as close to 1 as possible when the alarm in the clip is judged to be clearly audible and close to 0 when the alarm is not clearly audible. However, information such as an actual *Yes*-rate is totally absent from the data seen by the model. As a consequence, we do not necessarily expect a match between the output of the model and a psychometric function when varying the SNR of the alarm present in a clip. Yet, we did try to observe the output of the model when the inputs were the same clips of *subA* with different SNRs. The activation of the last neuron was found to roughly follow the evolution of what could be interpreted as a psychometric curve. This result opens up analytical perspectives for future studies. An example is shown in Figure 1.

6. CONCLUSION

In this paper, we proposed a proof of concept of a new approach to assess the audibility of acoustic alarms. We presented an experimental procedure that was specifically designed to collect a dataset with perceptual annotation. This dataset was used to develop a model that gave auspicious results on a binary classification task. Both the influence of the training data and the importance of the temporal context have been investigated. Our results showed that it is possible to predict the audibility of acoustic alarms in relative accordance with human perception, even if training was made on a dataset that was collected using a much lighter procedure than usual psychoacoustical tests. However, we are aware of the lack of a baseline to compare the results of the present work, and therefore plan to collect new perceptual data with different annotators whose "performance" will serve as a basis for comparison with the model. Lastly, the psychoacoustical experiments presented in this article are part of a broader experimental method that includes the numerical rating of the audibility and a detection task that have not been detailed here. The data collected on this occasion will be used in future developments.

7. REFERENCES

[1] I. O. for standardization (ISO), "Ergonomics — danger signals for public and work areas — auditory danger signals," 2008.

[2] J. Żera and A. Nagórski, "Preferred levels of auditory danger signals," *Int. J. Occup. Saf. Ergon.*, vol. 6:sup1, pp. 111–117, 2004.

[3] E. Zwicker and H. Fastl, *Psychoacoustics: Facts and Models*. Berlin: Springer, 1999.

[4] B. C. J. Moore, *An Introduction to the Psychology of Hearing*, 6th ed. Leiden, Netherlands: Brill, 2013.

[5] R. Patterson, "Auditory warning sounds in the work environment," *Phil. Trans. R. Soc. Lond.*, vol. B 327, pp. 485–492, 1990.

[6] M. R. Leek, "Adaptive procedures in psychophysical research," *Perception & Psychophysics*, vol. 63, pp. 1279–1292, 2001.

[7] Y. Zheng, C. Giguère, C. Laroche, C. Sabourin, A. Gagné, and M. Elyea, "A Psychoacoustical Model for Specifying the Level and Spectrum of Acoustic Warning Signals in the Workplace," *Journal of Occupational and Environmental Hygiene*, vol. 4, no. 2, pp. 87–98, Jan. 2007.

[8] B. R. Glasberg and B. C. J. Moore, "Development and Evaluation of a Model for Predicting the Audibility of Time-Varying Sounds in the Presence of Background Sounds," *J. Audio Eng. Soc*, vol. 53, no. 10, pp. 906–918, 2005.

[9] B. RichardWebster, S. E. Anthony, and W. J. Scheirer, "Psyphy: A psychophysics driven evaluation framework for visual recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 9, pp. 2280–2286, 2019.

[10] S. Grieggs, B. Shen, G. Rauch, P. Li, J. Ma, D. Chiang, B. Price, and W. Scheirer, "Measuring human perception to improve handwritten document transcription," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[11] A. J. Kell, D. L. Yamins, E. N. Shook, S. V. Norman-Haignere, and J. H. McDermott, "A task-optimized neural network replicates human auditory behavior, predicts brain responses, and reveals a cortical processing hierarchy," *Neuron*, vol. 98, no. 3, pp. 630–644.e16, 2018.

[12] A. Francl and J. McDermott, "Deep neural network models of sound localization reveal how perception is adapted to real-world environments," *Nat. Hum. Behav.*, vol. 6, pp. 11–133, 2022.

[13] E. Cakir, S. Adavanne, G. Parascandolo, K. Drossos, and T. Virtanen, "Convolutional recurrent neural networks for bird audio detection," in *2017 25th European Signal Processing Conference (EUSIPCO)*, 2017, pp. 1744–1748.

[14] N. Turpault and R. Serizel, "Training sound event detection in a heterogenous dataset," in *Proc. DCASE Workshop*, 2020.

[15] J. Abeßer, "A review of deep learning based methods for acoustic scene classification," *Appl. Sci.*, vol. 10, no. 6, 2020.

[16] E. Çakir and T. Virtanen, "End-to-end polyphonic sound event detection using convolutional recurrent neural networks with learned time-frequency representation input," in *2018 International Joint Conference on Neural Networks (IJCNN)*, 2018.

[17] X. Xia, R. Togneri, F. Sohel, Y. Zhao, and D. Huang, "A survey: Neural network-based deep learning for acoustic event detection," *Circuits Syst. Signal Process.*, vol. 38, pp. 3433–3453, 2019.

[18] E. Fonseca, M. Plakal, F. Font, D. P. W. Ellis, and X. Serra, "Audio tagging with noisy labels and minimal supervision," in *Proc. DCASE Workshop*, 2019.

[19] The Freesound project, https://www.freesound.org.

[20] BigSoundBank by Joseph SARDIN, https://www.bigsoundbank.com.

[21] J. Atyeo and P. M. Sanderson, "Comparison of the identification and ease of use of two alarm sound sets by critical and acute care nurses with little or no music training: a laboratory study," *Anaesthesia*, vol. 70, no. 7, pp. 818–827, 2015.

[22] J.-P. Arz, N. Grimault, and O. ElSawaf, "Experimental assessment of the effect of wearing hearing protectors on the audibility of railway warning signals for normal hearing and hearing impaired listeners," *International Journal of Occupational Safety and Ergonomics*, 2021. [Online]. Available: https://doi.org/10.1080/10803548.2021.1991681

[23] B. R. Glasberg and B. C. J. Moore, "A model of loudness applicable to time-varying sounds," *Journal of the Audio Engineering Society*, vol. 50, no. 5, pp. 331–342, 2002.

[24] B. McFee, J. Salamon, and J. P. Bello, "Adaptive pooling operators for weakly labeled sound event detection," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 26, no. 11, pp. 2180–2193, 2018.

[25] N. Turpault, R. Serizel, and E. Vincent, "Analysis of weak labels for sound event tagging," 2021, working paper or preprint. [Online]. Available: https://hal.inria.fr/hal-03203692

[26] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv:1412.6980*, 2014.

DETECTION AND IDENTIFICATION OF BEEHIVE PIPING AUDIO SIGNALS

Dominique Fourer and Agnieszka Orlowska

IBISC (EA 4526) - University of Evry / Paris-Saclay*
Evry-Courcouronnes, France
dominique.fourer@univ-evry.fr

ABSTRACT

Piping signals are particular sounds emitted by honey bees during the swarming season or sometimes when bees are exposed to specific factors during the life of the colony. Such sounds are of interest for beekeepers for predicting an imminent swarming of a beehive. The present study introduces a novel publicly available dataset made of several honey bee piping recordings allowing for the evaluation of future audio-based detection and recognition methods. First, we propose an analysis of the most relevant timbre features for discriminating between tooting and quacking sounds which are two distinct types of piping signals. Second, we comparatively assess several machine-learning-based methods designed for the detection and the identification of piping signals through a beehive-independent 3-fold cross-validation methodology.

Index Terms— bees piping signals, quacking, tooting, audio signal recognition, smart beekeeping

1. INTRODUCTION

Nowadays, smart beekeeping is gaining interest since it aims at developing innovative methods for enhancing the monitoring of beehives using AI techniques. To this end, the audio-based approach [1, 2] is promising since it allows to use low-cost sensors for monitoring a bee colony. Recent work pioneered the bee sound analysis problem through a machine learning approach to predict the different health states of a beehive. For example, the task of predicting the bee queen presence is investigated in [3, 4] and could help beekeepers to reduce the number of inspections which are stressful for a beehive. The prediction of colony swarming from audio signal is investigated [5, 6] and can be related to specific sounds emitted by the bees. Several studies analyze different piping sounds and show their interest for beekeepers [7, 5, 6]. Other studies explain that piping signals can also have other functions for synchronizing the colony activity [7, 8]. Such particular sounds can respectively be emitted by bee workers or by a queen and can easily be distinguished from classical background beehive sounds. A more recent study [9] proposes an acoustic analysis of piping signals which can be segregated into two classes with specific audio signatures: tooting and quacking. Both tooting and quacking signals can occur about 1 day before swarming and their occurrences can increase every 10 minutes during approximately 6 hours.

The present study pursues the piping sounds investigation with an analysis of the most relevant audio features using a machine learning-based methodology. Our contributions are manifold. First, we introduce a new publicly available audio piping dataset made of several recordings collected from various beekeepers which were manually segmented and annotated as tooting or quacking. Second, we present an acoustic analysis through timbre features to discriminate between the tooting and the quacking signals. Finally, we assess several methods for a supervised detection and classification of audio field recordings of beehive sounds. This paper is organized as follows. In Section 2, we explain the differences between piping signals and we introduce our new proposed dataset. In Section 3, we perform an acoustic analysis of piping sounds using timbre features. Section 4 presents our audio detection and classification results using several proposed methods. Finally, the paper is concluded by a discussion with future work directions in Section 5.

2. MATERIALS

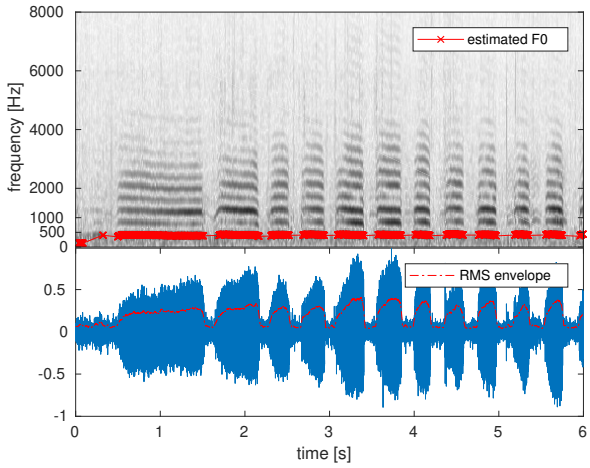
2.1. Tooting and Quacking

Piping sounds (cf. Fig. 1) are among the most noticeable signs of swarming. *Tooting* corresponds to the sound emitted by a virgin queen bee who announces her presence by releasing pheromones and by tooting. Tooting corresponds to a series of pulsed, high-pitched sounds produced by pressing her thorax and operating her wing-beating mechanism without spreading her wings [10]. Mature queens still confined within their queen cells answer the tooting with a distinct piping sound, called *Quacking*. A chorus of synchronized quacking follows each tooting, and those specific swarming sounds are broadcasting in the bee nest as vibrations of the combs and perceived by vibration detectors in the workers' tarsi [11]. Toots and quacks are made of different varying pulses: during the process of tooting, the queen produces a one-second-long pipe immediately followed by several bursts of less than half a second. The fundamental frequency increases with the age of queens, ranging from 200 to 550 Hz, and is usually observed around 400 Hz [12]. Quacks are made of several short pulses which are typically less than 0.2 seconds at a lower fundamental frequency around 350 Hz [13]. Piping sounds are not only emitted by queens but also by workers in queenless colonies: laying-workers and guarding-workers [14]. More recent studies show that workers could emit piping sounds to prepare a synchronized liftoff [7]. This prompts a conclusion that workers pipe in a variety of circumstances, while queens pipe only in the context of colony reproduction [15]. The queens' toots and quacks last several seconds and are broken up into syllables [12]. Piping sounds emitted by workers come from several sources and have a duration below one second. It often consists of a single pulse [14].

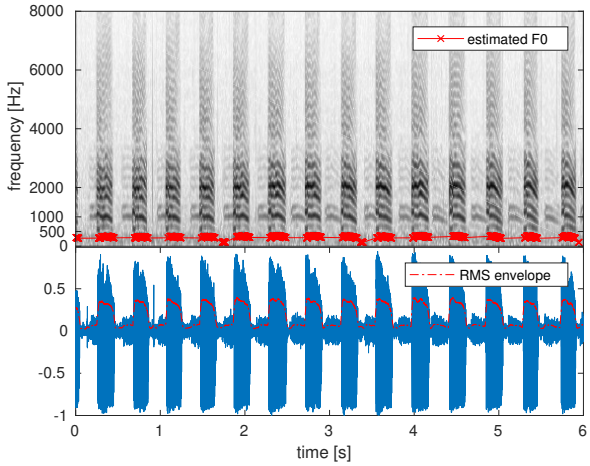
2.2. New Proposed Piping Dataset

We introduce a novel dataset of natural honey bee piping audio signals which was built by collecting 44 different recordings pub-

*This work is partly supported by the French ANR ASCETE project (ANR-19-CE48-0001).



(a) Tooting



(b) Quacking

Figure 1: Spectrograms with highlighted F_0 and waveforms with RMS envelope of two distinct piping signals.

lished on the YouTube platform by various beekeepers around the world.These audio recordings were obtained in field conditions using various non-professional microphones located close to the beehive when a piping signal is emitted. Each recording has a duration varying from 2 to 13 seconds and is annotated according to the beekeeper comment respectively as *Tooting* or *Quacking*. We extracted and segmented the audio from 14 distinct videos from which the signal is recorded without a loss of quality into WAVE files with a sampling frequency of $F_s = 22.05$ kHz and a sample precision of 16 bits. After manually removing the silent and spurious frames, the resulting dataset contains 36 tooting signals and 8 quacking signals which correspond to a duration of 145 seconds for tooting and 60 seconds for quacking (total 205 seconds). To avoid possible copyright issues, we only made publicly available the Short-Time Fourier Transform (STFT) matrices and the timbre descriptors computed using a matlab implementation of the timbre toolbox [16] from the post-processed signals used in our experiments. We propose a more detailed description of the dataset containing the links of the original Youtube videos with our matlab loader codes published on IEEE DataPort [17].

3. ACOUSTIC ANALYSIS

3.1. Signal analysis

We present in Fig. 1 the waveform of a tooting and of a quacking signal both extracted from our proposed dataset (Toot1 and Quack1) with almost the same duration of about 6 seconds. Colored in red, we plot the Root Mean Square (RMS) envelope computed for a window length of 23ms. We also display the spectrograms of the same signals where the fundamental frequency (F_0) estimated using the SWIPE method [18] is highlighted. From these observations, one can notice that tooting and quacking are both harmonic signals but with very different temporal and spectral structures. The tooting signal contains longer pulses with a higher F_0 (mean value of $\mu_T = 382.97$Hz with a standard deviation $\sigma_T = 61.45$ Hz) and a slightly lower number of pulses for the same observation duration. For the comparison, the quacking signal contains more pulses with a lower F_0 ($\mu_Q = 306.60$ Hz, $\sigma_Q = 23.98$ Hz). We also notice that the F_0 decreases at the end of each pulse for both tooting and quacking signals.

Table 1: Top-10 most relevant timbre descriptors selected using a mutual information criterion.

	Timbre feature	Relevance score
1	*ERB-gammmatone Spectral Centroid*	0.428
2	*ERB-gammatone Spectral Kurtosis*	0.419
3	*ERB-fft Spectral Kurtosis*	0.402
4	*ERB-gammatone Spectral Skewness*	0.373
5	*ERB- fft Spectral Skewness*	0.373
6	*ERB-fft Spectral Centroid*	0.371
7	*ERB-fft Spectral Spread*	0.334
8	*Zero-crossing rate*	0.321
9	*STFT Spectral Kurtosis*	0.314
10	*STFT Spectral Roll-Off*	0.311

3.2. Timbre Feature Selection

The timbre toolbox proposed by Peeters et al. [16] proposes a large set of hand-crafted audio features used in various audio recognition tasks. These features are expected to convey information about the perceived timbre of an arbitrary sound. They include temporal, spectral, harmonic and perceptual descriptors which are directly computed from the waveform and from the time-frequency representation of the analyzed signal. In this study, we investigate a total of 164 timbre features (cf. [19] Table. 2 for details) summarized by median and Inter Quartile Range (IQR) statistics related to the signal acoustic parameters. In Table 1, we present the top-10 most relevant features sorted by descending order of relevance according to the mutual information (MI) criterion [20] by considering the tooting/quacking classification problem. Our computation uses the scikit-learn MI python implementation which shows that perceptual-based Equivalent-Rectangular-Bandwith (ERB) spectral features appear to be the most relevant. Fig. 2a plots in 3 dimensions the whole dataset where each individual corresponds to a one-second-long frame where the axes correspond to the top-3 most relevant features. This figure shows that the components can almost be separated into two distinct clusters corresponding to tooting and quacking signals (plotted with different colors) using only 3 relevant features. In Fig. 2b, we plot a whole dataset projection using Principal Component Analysis (PCA) which is a dimension reduction method reducing the redundancy between the features while

preserving original data inertia. This second projection shows that the separation between tooting and quacking sounds is not trivial despite each cluster seem located in a different area. Finally, we perform a Linear Discriminant Analysis (LDA) [21] which can be viewed as a supervised PCA providing the optimal linear projection of the dataset which maximizes the Euclidean distance between individuals of different classes while minimizing the distance between individuals of the same class. Fig. 2c shows that there exists a linear combination of the original timbre features enabling to perfectly separate tooting and quacking sounds. This result paves the way of a supervised classification investigated in Section 4.

4. DETECTION AND CLASSIFICATION RESULTS

4.1. Experimental Setup

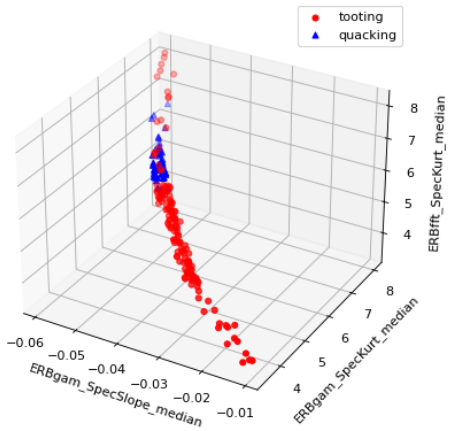
We focus on two distinct tasks which consist of the detection of piping signals and the discrimination between tooting and quacking piping signals. To this end, we consider three distinct experiments. **Experiment 1** focuses on the detection of piping signals from beehives recordings. We address this problem through a binary classification problem involving samples from our proposed dataset and beehive recordings from the OSBH dataset[1] made of several beehives sounds. **Experiment 2** focuses on the binary piping audio classification problem which consists in identifying respectively tooting and quacking signals where 145 recordings are labeled as *tooting* and 60 recordings as *quacking*. **Experiment 3** considers both the detection and the classification problem that is addressed through a 3-label supervised classification approach consisting in predicting if a signal is a *tooting*, a *quacking* or a *non-piping* signal. For each experiment, datasets are preprocessed by splitting signals into one-second-long chunks sampled at $F_s = 22.05$ kHz. Each signal is centered by subtracting the mean and the amplitude is normalized by dividing each sample by $\max(|x|)$. Our evaluation uses a 3-fold cross-validation methodology (2 training folds and 1 testing fold) where the recordings are beehive-independent to avoid overfitting and to assess over the whole dataset the generalizing capability of the trained models. Hence, all recordings from the same Youtube video are only present into a unique fold and cannot simultaneously appear in both the training and testing sets. In experiments 1 and 3 involving *non-piping* signals, we randomly add bee signals from the OSBH dataset to obtain the same number of *piping* and *non-piping* signals in each fold.

4.2. Methods

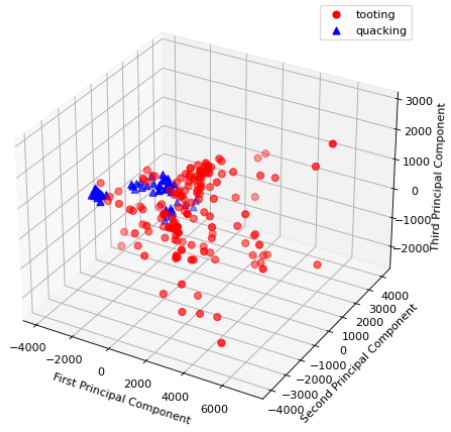
4.2.1. Classification

We comparatively assess four distinct supervised classification methods suitable for beehive audio signals. The **TTB+SVM** method uses the 164 timbre descriptors investigated in Section 3 combined with a support vector machines (SVM) classifier with a Gaussian radial basis function kernel [22]. The proposed **1D-CNN** method uses the modulus of the discrete Fourier transform of the signal as input of a 1D-convolutional neural network (CNN) with residual connections. This architecture (total: 7,684,226 trainable parameters) is made of 4 residual blocks with a different number of kernel filters (sequentially: 16, 32, 64, 128). Each residual block is made of 3 one-dimensional convolutional layers interspersed by the addition of the input followed by a Rectified Linear Unit (ReLU) activation and a max-pooling. Output of the last residual block is average-pooled and connected to 3 fully-connected (FC) layers including flatten and with ReLU and softmax activation for the final output. The **MFCC+CNN** and the **STFT+CNN** are based on the same 2D-CNN architecture (total: 404,770 trainable

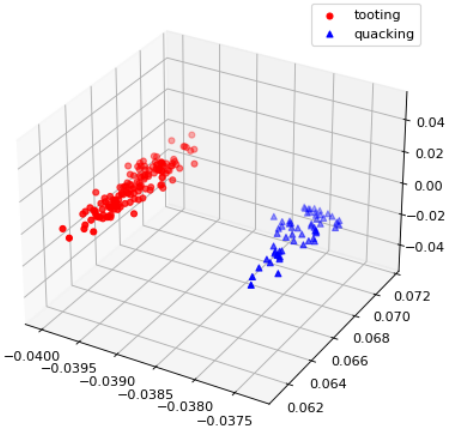
[1] https://zenodo.org/record/1321278



(a) Top-3 most relevant timbre features



(b) PCA



(c) LDA

Figure 2: Three-dimensional projections of our proposed piping dataset where each point corresponds to a one-second-long excerpt.

parameters) with 2 distinct inputs: Mel-Frequency Cepstral Coefficients (MFCC) and the spectrogram defined as the squared modulus of the short-time Fourier transform (STFT). The proposed 2D-CNN architecture is inspired from [2] and consists of 4 convolutional blocks containing 16 kernel filters of size 3×3, a 2×2 max-pooling layer and a 25% dropout layer. The output is connected to a 3 FC layers including 2 dropout layers of respectively 25% and 50% followed by a softmax activation function to compute the output predicted label. Convolutional and FC layers both use a LeakyReLU activation function defined as $LeakyRELU(x) = \max(\alpha x, x)$, with $\alpha = 0.1$.

4.2.2. Detection

For detecting piping in an arbitrary audio signal as proposed in **Experiment 1**, we also consider the 4 proposed classification methods using a binary *piping/non-piping* taxonomy. We also consider two additional methods based on the stochastic modeling of the estimated F_0 distribution respectively for piping and non-piping signals. This later approach is motivated by the harmonic property of piping signals described in Section 3. The **F0 Gaussian model** estimates the parameters $\theta = [\mu, \sigma^2]$ of a Gaussian probability distribution used to model respectively piping and non-piping signals. Thus, given the estimated F_0 denoted f_x of a signal, the decision to detect a piping signal is made when $p(f_x|\theta_{piping}) > p(f_x|\theta_{non-piping})$. The **F0 kernel model** is a variant of the **F0 Gaussian model** where $p(f_x|\theta_{piping})$ is estimated using the empirical distribution (i.e. histogram) of the estimated F_0 smoothed by a convolution product using a Gaussian kernel [23]. Our experiments used the SWIPE F_0 estimator [18] for which the median function is used to summarize a frame of signal with an arbitrary length.

4.3. Implementation details

The 17 first cepstral coefficients of the **MFCC+CNN** method are computed each 20 ms. The STFT is computed using a Hann analysis window with a 50% overlap and a FFT size of $M = 1025$ (i.e. 512 positive frequency bins). The input of the **1D-CNN** is set to half of the frequency sampling due to the Fourier transform Hermitian symmetry of a real signal (i.e. 11,025 real-valued coefficients). During the testing of each of the 3 folds, we use data augmentation (DA) [24] to artificially increase the number of training recordings by generating new samples from the original ones by the addition of a white Gaussian noise (SNR = 25dB) and by the application of temporal random circular shifts. The results reported in Tables 2, 3 and 4 correspond to the best ones obtained after several iterations (no significant improvement is shown by data augmentation). The training of our CNN methods is configured for a constant number of 25 epochs for the **1D-CNN** and 50 epochs for the **2D-CNN**, with a batch size of 16. The overall evaluation framework and the **TTB+SVM** method are implemented in matlab. The deep learning methods are implemented in Python using Keras with Tensorflow frameworks. Our codes are freely available online[2] for the sake of reproducible research.

4.4. Comparative results

According to Table 2, the best detection results in terms of accuracy for **Experiment 1** are obtained using the **TTB+SVM** and the **MFCC+CNN** method which both obtain 94%. The best piping classification (**Experiment 2**) results (cf. Table 3) are obtained using the **STFT+CNN** method with an overall accuracy of 95%, followed from far by the **MFCC+CNN** method which obtains an accuracy of 78%. Despite efforts, the two other techniques fail to identify quacking sounds and obtain poorer results with a quacking F-measure below 0.5. These poor quacking recognition results are confirmed in **Experiment 3** (cf. Table 4) where the best method remain **STFT+CNN** for which the results are poorer than in **Experiment 2**. This suggests the best pipeline which detects piping signals using **MFCC+CNN** or **TTB+SVM** before attempting to discriminate between tooting and quacking signals using **STFT+CNN**.

Table 2: Experiment 1: Piping signals detection comparative results.

Method	Feat. dimension	Label	Recall	Precision	F - score	Accuracy
F0 kern. model	1	Piping	0.68	0.96	0.79	0.84
		Non-piping	0.97	0.78	0.87	
F0 Gauss. model	1	Piping	0.69	0.99	0.81	0.85
		Non-piping	1	0.79	0.88	
TTB+SVM	164	Piping	**0.91**	0.96	**0.94**	0.94
		Non-piping	0.97	**0.93**	**0.95**	
1D-CNN	11,025	Piping	0.84	**1.00**	0.91	0.93
		Non-piping	**1.00**	0.88	0.93	
MFCC+CNN	17×47	Piping	0.87	**1.00**	0.93	**0.94**
		Non-piping	**1.00**	**0.90**	0.94	
STFT+CNN	512×42	Piping	0.86	0.96	0.91	0.92
		Non-piping	0.97	0.89	0.93	

Table 3: Experiment 2: Piping signals binary classification comparative results.

Method	Feat. dimension	Label	Recall	Precision	F - score	Accuracy
TTB+SVM	164	Tooting	0.78	0.85	0.71	0.66
		Quacking	0.24	0.18	0.38	
1D-CNN	11,025	Tooting	**0.97**	0.72	0.82	0.71
		Quacking	0.08	0.50	0.14	
MFCC+CNN	17×47	Tooting	0.93	0.79	0.86	0.78
		Quacking	0.42	0.71	0.53	
STFT+CNN	512×42	Tooting	0.94	**0.98**	**0.96**	**0.95**
		Quacking	**0.96**	**0.87**	**0.92**	

Table 4: Experiment 3: Simultaneously Detection and classification classification comparative results.

Method	Feat. dimension	Label	Recall	Precision	F - score	Accuracy
TTB+SVM	164	Tooting	0.88	0.78	0.83	0.82
		Quacking	0.03	0.12	0.05	
		Non-piping	0.99	0.89	0.94	
1D-CNN	11,025	Tooting	0.93	0.84	0.88	0.85
		Quacking	0.10	0.54	0.16	
		Non-piping	0.99	0.86	0.92	
MFCC+CNN	17×47	Tooting	0.88	0.81	0.84	0.84
		Quacking	0.18	0.45	0.26	
		Non-piping	**0.99**	**0.90**	**0.95**	
STFT+CNN	512×42	Tooting	**0.94**	**0.97**	**0.95**	**0.91**
		Quacking	**0.50**	**0.76**	**0.60**	
		Non-piping	**0.99**	0.89	0.94	

5. CONCLUSION

We introduced a new dataset made of beehive piping sounds designed for identifying tooting and quacking signals emitted by bees. The most relevant timbre features were presented and reveal a link with perceptual spectral features. Our numerical experiments involving several state-of-the-art approaches show that a time-frequency represention combined with a 2D-CNN is currently the most promising approach for addressing the tooting/quacking binary classification problem and can obtain an accuracy above 85%. Future work consists in evaluating new methods in more realistic application scenarios involving embedded systems.

[2] https://fourer.fr/dcase22

6. REFERENCES

[1] S. Cecchi, A. Terenzi, S. Orcioni, P. Riolo, S. Ruschioni, and N. Isidoro, "A preliminary study of sounds emitted by honey bees in a beehive," in *Audio Engineering Society Convention 144*, Milan, Italy, May 2018.

[2] S. Cecchi, A. Terenzi, S. Orcioni, and F. Piazza, "Analysis of the sound emitted by honey bees in a beehive," in *Audio Engineering Society Convention 147*, 2019.

[3] I. Nolasco and E. Benetos, "To bee or not to bee: Investigating machine learning approaches for beehive sound recognition," in *Proc. DCASE*, Nov. 2018.

[4] I. Nolasco, A. Terenzi, S. Cecchi, S. Orcioni, H. L. Bear, and E. Benetos, "Audio-based identification of beehive states," in *Proc. IEEE ICASSP*, 2019, pp. 8256–8260.

[5] A. Zgank, "Bee swarm activity acoustic classification for an iot-based farm service," *Sensors*, vol. 20, no. 1, p. 21, 2020.

[6] M.-T. Ramsey, M. Bencsik, M. I. Newton, M. Reyes, M. Pioz, D. Crauser, N. S. Delso, and Y. Le Conte, "The prediction of swarming in honeybee colonies using vibrational spectra," *Scientific reports*, vol. 10, no. 1, pp. 1–17, 2020.

[7] T. D. Seeley and J. Tautz, "Worker piping in honey bee swarms and its role in preparing for liftoff," *Journal of Comparative Physiology A*, vol. 187, no. 8, pp. 667–676, 2001.

[8] C. Thom, D. C. Gilley, and J. Tautz, "Worker piping in honey bees (apis mellifera): the behavior of piping nectar foragers," *Behavioral Ecology and Sociobiology*, vol. 53, no. 4, pp. 199–205, 2003.

[9] T. Yamamoto, M. Sugahara, R. Okada, and H. Ikeno, "Differences between queen piping temporal structures of two honeybee species, apis cerana and apis mellifera," *Apidologie*, vol. 52, no. 2, pp. 524–534, 2021.

[10] J. Simpson, "The mechanism of honey-bee queen piping," *Zeitschrift für vergleichende Physiologie*, vol. 48, no. 3, pp. 277–282, 1964.

[11] J. Simpson and S. M. Cherry, "Queen confinement, queen piping and swarming in apis mellifera colonies," *Animal Behaviour*, vol. 17, pp. 271–278, 1969.

[12] A. Michelsen, W. H. Kirchner, B. B. Andersen, and M. Lindauer, "The tooting and quacking vibration signals of honeybee queens: a quantitative analysis," *Journal of Comparative Physiology A*, vol. 158, no. 5, pp. 605–611, 1986.

[13] W. Kirchner, "Acoustical communication in honeybees," *Apidologie*, vol. 24, no. 3, pp. 297–307, 1993.

[14] T. Ohtani and T. Kamada, "'worker piping': The piping sounds produced by laying and guarding worker honeybees," *Journal of Apicultural Research*, vol. 19, no. 3, pp. 154–163, 1980.

[15] S. Pratt, S. Kühnholz, T. D. Seeley, and A. Weidenmüller, "Worker piping associated with foraging in undisturbed queenright colonies of honey bees," *Apidologie*, vol. 27, no. 1, pp. 13–20, 1996.

[16] G. Peeters, B. L. Giordano, P. Susini, N. Misdariis, and S. McAdams, "The timbre toolbox: Extracting audio descriptors from musical signals," *The Journal of the Acoustical Society of America*, vol. 130, no. 5, pp. 2902–2916, 2011.

[17] A. Orlowska and D. Fourer, "Identification of beehive piping audio signals," in *IEEE Dataport (doi:10.21227/53mq-g936)*, 2021. [Online]. Available: https://dx.doi.org/10.21227/53mq-g936

[18] A. Camacho and J. G. Harris, "A sawtooth waveform inspired pitch estimator for speech and music," *The Journal of the Acoustical Society of America*, vol. 124, no. 3, pp. 1638–1652, 2008.

[19] D. Fourer, J.-L. Rouas, P. Hanna, and M. Robine, "Automatic timbre classification of ethnomusicological audio recordings," in *Proc. ISMIR*, Taipei, Taiwan, Oct. 2014.

[20] N. Hoque, D. K. Bhattacharyya, and J. K. Kalita, "Mifs-nd: A mutual information-based feature selection method," *Expert Systems with Applications*, vol. 41, no. 14, pp. 6371–6385, 2014.

[21] T. W. Anderson, *An Introduction to Multivariate Statistical Analysis*. New York, USA: Wiley-Blackwell, 1958.

[22] A. J. Smola and B. Schölkopf, "A tutorial on support vector regression," *Statistics and computing*, vol. 14, no. 3, pp. 199–222, 2004.

[23] A. W. Bowman and A. Azzalini, *Applied smoothing techniques for data analysis: the kernel approach with S-Plus illustrations*. OUP Oxford, 1997, vol. 18.

[24] D. A. Van D. and X.-L. Meng, "The art of data augmentation," *Journal of Computational and Graphical Statistics*, vol. 10, no. 1, pp. 1–50, 2001.

SOUND EVENT LOCALIZATION AND DETECTION FOR REAL SPATIAL SOUND SCENES: EVENT-INDEPENDENT NETWORK AND DATA AUGMENTATION CHAINS

Jinbo Hu[1,2]*, Yin Cao*[3]*, Ming Wu*[1]*, Qiuqiang Kong*[4]*, Feiran Yang*[1]*, Mark D. Plumbley*[5]*, Jun Yang*[1,2]

[1]Key Laboratory of Noise and Vibration Research, Institute of Acoustics,
Chinese Academy of Sciences, Beijing, China, {hujinbo, mingwu, feiran, jyang}@mail.ioa.ac.cn
[2]University of Chinese Academy of Sciences, Beijing, China
[3]Xi'an Jiaotong Liverpool University, Suzhou, China, yin.k.cao@gmail.com
[4]ByteDance Shanghai, China, kongqiuqiang@bytedance.com
[5]Centre for Vision, Speech and Signal Processing (CVSSP), University of Surrey, UK
m.plumbley@surrey.ac.uk

ABSTRACT

Sound event localization and detection (SELD) is a joint task of sound event detection and direction-of-arrival estimation. In DCASE 2022 Task 3, types of data transform from computationally generated spatial recordings to recordings of real-sound scenes. Our system submitted to the DCASE 2022 Task 3 is based on our previous proposed Event-Independent Network V2 (EINV2) with a novel data augmentation method. Our method employs EINV2 with a track-wise output format, permutation-invariant training, and a soft parameter-sharing strategy, to detect different sound events of the same class but in different locations. The Conformer structure is used for extending EINV2 to learn local and global features. A data augmentation method, which contains several data augmentation chains composed of stochastic combinations of several different data augmentation operations, is utilized to generalize the model. To mitigate the lack of real-scene recordings in the development dataset and the presence of sound events being unbalanced, we exploit FSD50K, AudioSet, and TAU Spatial Room Impulse Response Database (TAU-SRIR DB) to generate simulated datasets for training. We present results on the validation set of Sony-TAu Realistic Spatial Soundscapes 2022 (STARSS22) in detail. Experimental results indicate that the ability to generalize to different environments and unbalanced performance among different classes are two main challenges. We evaluate our proposed method in Task 3 of the DCASE 2022 challenge and obtain the second rank in the teams ranking. Source code is released[1].

Index Terms— Sound event localization and detection, real spatial sound scenes, Event-Independent Network, data augmentation chains, simulated datasets

1. INTRODUCTION

Sound event localization and detection (SELD) consists of sound event detection (SED) and direction-of-arrival (DoA) estimation. SED aims to detect the presence and types of sound events, and DoA estimation predicts the spatial locations of different sound sources. SELD characterizes sound sources in a spatial-temporal manner. SELD plays an important role in a wide range of applications, such as robot auditory and surveillance of intelligent home.

SELD has received broad attention recently. Adavanne et al. [1] proposed a polyphonic SELD approach using an end-to-end network, SELDnet, which was utilized for a joint task of SED and regression-based DoA estimation. SELD was then introduced in Task 3 of the Detection and Classification of Acoustics Scenes and Events (DCASE) 2019 Challenge for the first time, which uses the TAU Spatial Sound Events 2019 dataset [2]. Most datasets of spatial sound events are computationally simulated and these recordings are generated by convolving randomly chosen sound event examples with a corresponding random real-life spatial room impulse response (SRIR) to spatially place them at a given position [2–4]. To bring each iteration of Task 3 of DCASE Challenge closer to real conditions, stronger reverberation, diversity of environment, dynamic scenes with both moving and static sound sources, ambient noise, sound events of the same type, and unknown directional interfering events out of the target classes were added into datasets to complicate the SELD task. In 2022, the challenge transforms from computationally simulated spatial recordings to real spatial sound scene recordings. The Sony-TAu Realistic Spatial Soundscapes 2022 (STARSS22) dataset is manually annotated and released to serve as the development and evaluation dataset of DCASE2022 Task 3 this year [5].

SELDnet is unable to detect sound events of the same type but with different locations [1], which is also called homogeneous overlap. An event-independent network (EIN) with a track-wise output format was proposed to detect the homogeneous overlap problem [6–8]. In EIN, there are several event-independent tracks, and each track can be of any event. The number of tracks needs to be pre-determined according to the maximum number of overlapping events. EINV2, an improved version of EIN, utilizes multi-head self-attention (MHSA) and a soft parameter-sharing strategy of multi-task learning to achieve better performance [7].

The training set often deviates from real-scene spatial and acoustical environments, and mismatched distribution of locations and sound types between the training set and test set is common. A novel data augmentation method is used to generalize the model [8, 9]. The data augmentation method contains several data augmentation chains. These data augmentation chains consist of some randomly sampled data augmentation operations. The augmentation method can increase the diversity of augmented features.

In this study, our system is based on our previous proposed EINV2 with data augmentation chains. EINV2 is extended by

[1]https://github.com/Jinbo-Hu/DCASE2022-TASK3

Conformer, which is a combination structure of self-attention and convolution. The data augmentation method is composed of several augmentation operations. These data augmentation operations are sampled and layered randomly to combine to several data augmentation chains [8]. External data is allowed in this challenge. We generate simulated data by randomly convolving chosen samples of sound events from AudioSet [10] and FSD50K [11] with measured SRIRs from TAU Spatial Room Impulse Responses Database[2] (TAU-SRIR DB). The experimental results show the proposed model with the novel data augmentation method, which was trained on our simulated data, outperforms the DCASE2022 challenge Task 3 baseline model which was trained on official synthetic SELD mixtures[3]. In addition, we present class-wise and room-wise metric scores of the validation set of STARSS22 in detail. The proposed system obtains the second rank in Task 3 of DCASE 2022 Challenge[4].

2. THE METHOD

2.1. Input features

In this method, log-mel spectrograms and intensity vectors (IV) in log-mel space are used for features of the SELD task. First order ambisonics (FOA) include four-channel signals, i.e., omnidirectional channel $\mathbf{w}$, and three directional channels $\mathbf{x}$, $\mathbf{y}$, and $\mathbf{z}$. Log-mel spectrograms are computed from the mel filter banks and the short-time Fourier transform spectrograms, and IVs are cross-correlation of log-mel spectrograms of $\mathbf{w}$ with $\mathbf{x}$, $\mathbf{y}$ and $\mathbf{z}$ [12]. These features are directly calculated online using a 1-D convolutional layer, which supports data augmentation on raw waveform.

2.2. Network Architecture

The track-wise output format was introduced in our previous works [6–8]. It can be defined as

$$\boldsymbol{Y}_{\text{Trackwise}} = \left\{ (y_{\text{SED}}, y_{\text{DoA}}) \mid y_{\text{SED}} \in \mathbb{O}_{\mathbf{S}}^{M \times K}, y_{\text{DoA}} \in \mathbb{R}^{M \times 3} \right\} \quad (1)$$

where M is the number of tracks, K is the number of sound-event types, $\mathbb{O}_{\mathbf{S}}^{M \times K}$ is one-hot encoding of K classes, and $\mathbf{S}$ is the set of sound events. Cartesian DoA estimation is used here.

The number of tracks is determined by the maximum polyphony. Each track can only detect a sound event with a corresponding direction of arrival. While a model with a track-wise output format is trained, sound events may be predicted in any track, instead of a fixed track. It may cause the track permutation problem that sound events predicted and their ground truth may not be aligned in a fixed track. Permutation-invariant training (PIT) is proposed to tackle the problem effectively. The PIT loss is defined as

$$\mathscr{L}_{PIT}(t) = \min_{\alpha \in \mathbf{P}(t)} \sum_{M} \left\{ \lambda \cdot \ell_{\alpha}^{\text{SED}}(t) + (1 - \lambda) \cdot \ell_{\alpha}^{\text{DoA}}(t) \right\} \quad (2)$$

where $\alpha \in \mathbf{P}(\mathbf{t})$ indicates one of the possible permutations and λ is a loss weight between SED and DoA. $\ell_{\alpha}^{\text{SED}}$ is binary cross entropy loss for the SED task, and $\ell_{\alpha}^{\text{DoA}}$ is mean square error for the DoA task. The lowest loss will be chosen by finding a possible permutation, and the back-propagation is then performed.

[2] https://doi.org/10.5281/zenodo.6408611
[3] https://doi.org/10.5281/zenodo.6406873
[4] https://dcase.community/challenge2022

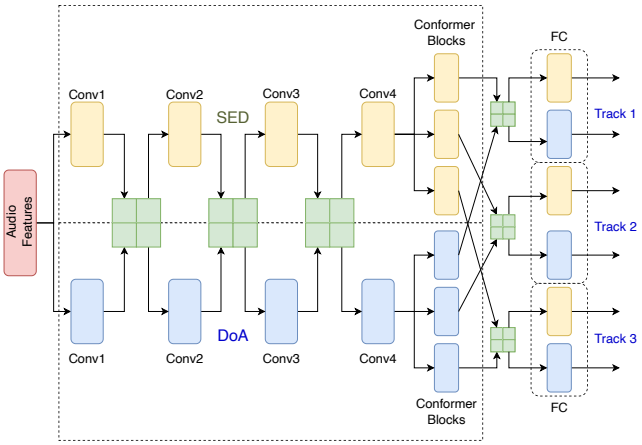


Figure 1: The architecture of the SELD network, which is a Conv-Conformer network. The upper half (yellow boxes) is the SED task. The lower half (blue boxes) is the DoA estimation task. The green boxes sandwiched between SED branch and DoA branch indicate soft connections between SED and DoA estimation.

From multi-task learning (MTL) perspective, joint SELD learning can be mutually beneficial. Hard parameter-sharing (PS) and soft PS are two typical methods to implement MTL. Hard PS means subtasks use the same feature layers, while soft PS means subtasks use their own feature layers with connections existing among those feature layers. In [7], experimental results show that soft PS using cross-stitch is more effective.

EINV2, which combines the track-wise output format, PIT, and soft PS, is utilized in our system. Three tracks are adopted to address up to three overlapped sound events. Multi-head self-attention (MHSA) blocks are replaced with Conformer blocks. Conformer consists of two feed-forward layers with residual connections sandwiching the MHSA and convolution modules, and hence has the ability to capture global and local patterns. [8, 13]. Our proposed network is shown in Fig. 1.

2.3. Data Augmentation Chains

The main characteristic of our data augmentation method is using some augmentation chains [8,9,14]. These augmentation chains are combined by some augmentation operations, which are randomly selected and linked in chain. We randomly sample $k = 3$ augmentation chains. Augmentation operations that are used here include Mixup [15], Cutout [16], SpecAugment [17], and frequency shifting [18]. Rotation of FOA signals [19] is an additional augmentation method, but excluded by data augmentation chains. The diagram of data augmentation chains is shown in Fig. 2.

Mixup utilize convex combinations of pairs of feature vectors and their labels to train the model. Mixup on both raw waveform and spectrograms is used here to improve the ability of detecting overlapping sound events. While random Cutout produces several rectangular masks on spectrograms, SpecAugment produces stripes masks on time and frequency dimension of spectrograms. Frequency shifting in the frequency domain is similar to pitch shift in the time domain, and it randomly shifts input features of all the channels up or down along the frequency dimension by several bands. We also use a spatial augmentation method, rotation of FOA signals. It rotates FOA format signals by channel swap to enrich DoA labels. This method does not lose physical relationships

Table 1: The SELD performance of our proposed system. The training set of STARSS22 is mixed into synthetic training set by default.

System	Datasets	Validation set				Evaluation (Blind test) set			
		ER_{20°	F_{20°	LE_{CD}	LR_{CD}	ER_{20°	F_{20°	LE_{CD}	LR_{CD}
Baseline FOA [5]	Official	0.71	21.0%	29.3°	46.0%	0.61	23.7%	22.9°	51.4%
EINV2 w/o dataAug chains	Official	0.75	32.3%	24.0°	56.1%	-	-	-	-
EINV2 w/ dataAug chains	Official	0.56	42.4%	19.3°	61.4%	-	-	-	-
System #1	A+B+C	0.50	48.4%	19.5°	65.7%	0.44	49.2%	16.6°	70.4%
System #2	A+B	0.50	51.0%	16.4°	65.9%	0.40	57.4%	15.1°	70.6%
System #3	A	0.53	48.1%	17.8°	62.6%	0.39	55.8%	16.2°	72.4%
System #4	B	0.53	45.4%	17.4°	62.5%	0.40	50.9%	15.9°	69.4%

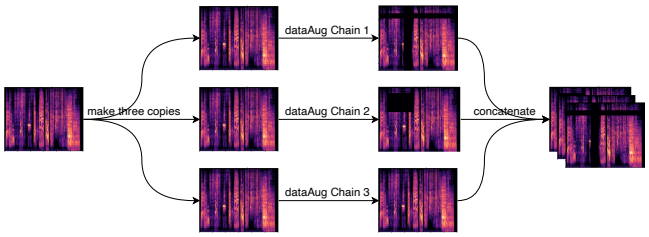


Figure 2: Diagram of data augmentation chains

between sound sources and observers. We use z-axis as the rotation axis to swap directional channel $\mathbf{x}$ and $\mathbf{y}$, which leads to 16 types of channel rotation.

2.4. Simulated Data

Manual annotations are expensive and the duration of STARSS22 (about 5 hours of the development set) is limited compared with the synthetic datasets (about 13 hours synthetic recordings in DCASE 2021) used in previous years, therefore, external datasets are used to improve the model performance. We generated simulated data using the generator code[5] provided by DCASE 2022.

Samples of sound events are mainly sourced from FSD50K dataset, based on affinity of the labels in that dataset to the target classes. The target class *background music* and the interference class *shuffling cards* are not in FSD50K dataset, therefore, we use AudioSet as a supplement. Spatial events were spatialized in 9 unique rooms, using collected SRIRs from the TAU-SRIR DB dataset. The ambient noise from the same room was additionally mixed at varying signal-to-noise ratios (SNR) ranging from 30 dB to 6 dB. The maximum polyphony of target classes is 3, excluding additional polyphony of interference classes.

We select sound event samples whose labels significantly corresponded to the target classes. Each sound event sample also has a different energy gain for mixing. By setting different ranges of gain and choosing different samples, we generate three datasets, A, B, and C. All of these synthetic datasets have 2700 1-minute clips.

3. EXPERIMENTS

3.1. Datasets

The STARSS22 dataset contains recordings of real scenes, and the density of sound event samples and the presence of each class varies greatly. The maximum number of the overlaps is 5, but those cases are very rare [5]. The overlap of 4 and 5 accounts for the proportion of 1.8% in total. Occurrences of up to 3 simultaneous events are fairly common, so we ignore the case scenarios that the number of overlapping events is more than 3. During the development stage, we train our proposed model on mixed datasets of synthetic recordings and the training set of STARSS22, and evaluate those systems using the validation set of STARSS22. During the evaluation stage, both synthetic recordings and all of the development set of STARSS22 are used for training.

[5] https://github.com/danielkrause/DCASE2022-data-generator

3.2. Hyper-parameters

Audio clips are segmented to have a fixed length of 5 seconds with no overlap for training and inference. Log-mel spectrograms and intensity vectors features, with 24 kHz sampling rate, a 1024-point Hanning window with a hop size of 400, and 128 mel bins, are extracted from these audio segments. AdamW optimizer is used. The learning rate is set to 0.0003 for the first 70 epochs and then decreased to 0.00003 for the following 20 epochs. The threshold for SED is set to 0.5 to binarize predictions. The loss weight λ is 0.5.

3.3. Evaluation Metrics

We use the official evaluation metrics to evaluate the SELD performance [20, 21]. The evaluation metrics use a joint metric of localization and detection: location-sensitive F-score ($F_{\leq T^\circ}$), error rate ($ER_{\leq T^\circ}$), and class-sensitive localization recall (LR_{CD}), localization error (LE_{CD}). T° means spatial threshold and is set to 20° in this challenge. $F_{\leq T^\circ}$ and $ER_{\leq T^\circ}$ consider true positives predicted under a spatial threshold T° from the ground truth. For LE_{CD} and LR_{CD}, the detected sound class has to be correct in order to count the corresponding localization predictions.

In the previous challenges, the evaluation metrics were micro-averaged, which gives equal weight to each individual decision and the performance is affected by the classes with more samples. In this challenge, macro-averaging of evaluation metrics is used. Macro-averaging gives equal weight to each class and emphasizes the system performance on the smaller classes [22].

3.4. Experimental Results

Table 1 summarizes the performance of our proposed systems. The official dataset means the synthetic mixtures for baseline training. The system baseline, EINV2 without dataAug chains, and EINV2 with dataAug chains all use the same dataset for training. EINV2

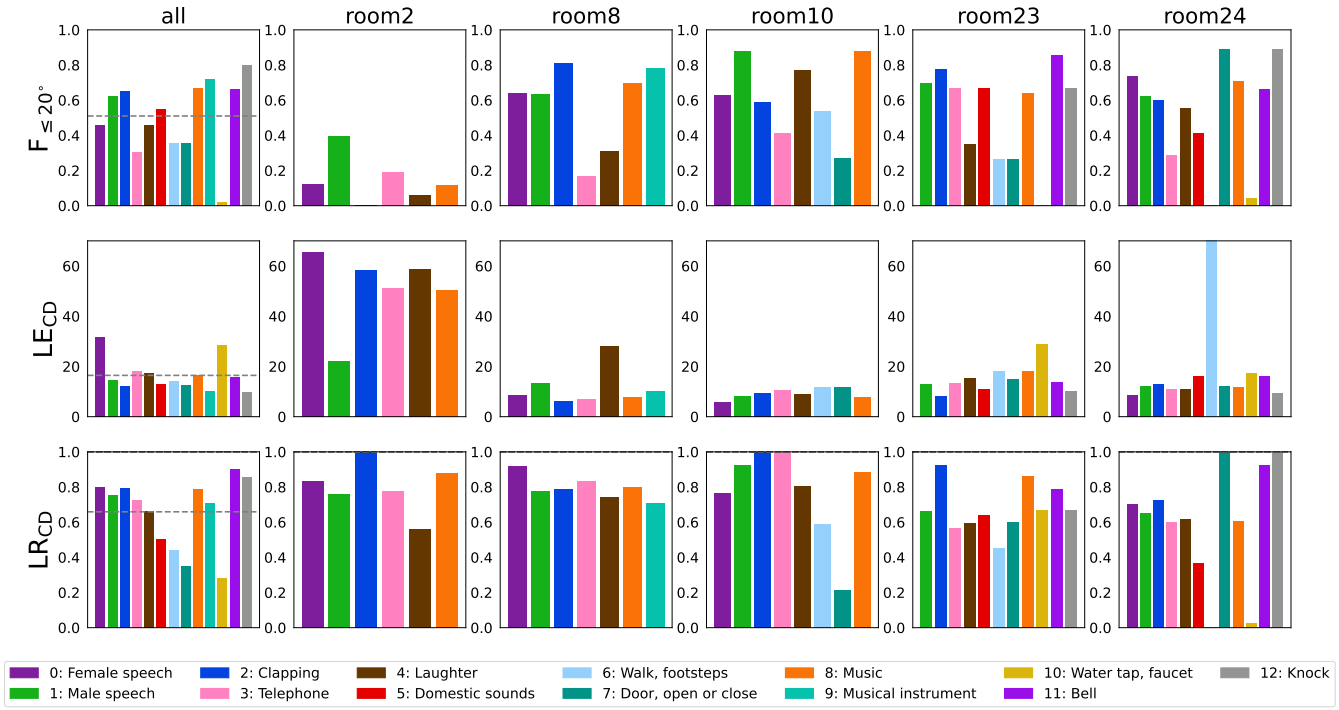


Figure 3: Metric scores of System #2 on validation set of STARSS22 in detail. The first column shows metric scores of the whole validation set. The following columns present metric scores of each room of validation set.

without data augmentation chains outperforms the baseline model, whereas EINV2 with data augmentation chains performs better.

All configurations of systems #1 - #4 are the same as system EINV2 with dataAug chains, except for the training set used. The results also demonstrate the effectiveness of our simulated data over the official dataset.

The first column of Fig. 3 shows class-wise metric scores of System #2 on the validation set of STARSS22. The class-wise performance on the whole validation set is highly skewed, with $F_{\leq 20^\circ}$ of *knock* class being 80.0%, whereas $F_{\leq 20^\circ}$ of *water tap and faucet* class being 2.2%. LE_{CD} of *female speech* class and *water tap and faucet* class is a lot higher than average. Other columns of Fig. 3 present class-wise performance for each room. Unbalanced class-wise performance among different rooms results in the skewed class-wise performance on the whole validation set.

The performance of the localization in room 2 is the worst among all the rooms, resulting in a directly significant increase of LE_{CD} of *female speech* class. It may be attributed to small room size of room 2 compared with other rooms. LR_{CD} of *walk, footsteps* (0.0%) class and *water tap and faucet* (2.4%) class in room 24 is very low. A possible reason is the low quality of synthetic training samples, because we ignore the natural temporal occurrences and spatial connections of some types of sounds happening in real scenes when simulating data [5]. For example, the target class *water tap and faucet* and the directional interference class *dishes, pots, and pans* often occur simultaneously in room 24, which leads to many observed false negatives of the class *water tap and faucet* in the system output. It is difficult to synthesis training samples that contains the temporal and spatial relationships of sound events in real scenes. These factors can lead to performance degradation.

4. CONCLUSION

We have presented an approach using an Event-Independent Network V2 (EINV2) with a novel data augmentation method for real-life sound event localization and detection. EINV2 is extended by conformer blocks. The novel data augmentation method contains several augmentation chains, which are stochastic combinations of data augmentation operations. For this challenge, we synthesized more training samples which are convolved using sound events from FSD50k and AudioSet with measured room impulse responses from TAU-SRIR DB. Our model with data augmentation chains performs better than the baseline model. Furthermore, experimental results show further improvement with our synthetic datasets. We also show results on the validation set of STARSS22 in detail. Our proposed method is evaluated in the evaluation set of STARSS22, and obtained the second best team in Task 3 of DCASE 2022 Challenge. The study of the generalization ability to different environments and the performance for unbalanced classes will be analyzed further in the future work.

5. ACKNOWLEDGEMENT

This work was partly supported by Frontier Exploration project independently deployed by Institute of Acoustics, Chinese Academy of Sciences (No. QYTS202009), UK Engineering and Physical Sciences Research Council (EPSRC) grant EP/T019751/1 “AI for Sound”. For the purpose of open access, the authors have applied a Creative Commons Attribution (CC BY) licence to any Author Accepted Manuscript version arising.

6. REFERENCES

[1] S. Adavanne, A. Politis, J. Nikunen, and T. Virtanen, "Sound event localization and detection of overlapping sources using convolutional recurrent neural networks," *IEEE J. Sel. Top. Signal Process.*, vol. 13, pp. 34–48, 2018.

[2] S. Adavanne, A. Politis, and T. Virtanen, "A multi-room reverberant dataset for sound event localization and detection," in *Proc. DCASE 2019 Workshop*, 2019, pp. 10–14.

[3] A. Politis, S. Adavanne, and T. Virtanen, "A dataset of reverberant spatial sound scenes with moving sources for sound event localization and detection," in *Proc. DCASE 2020 Workshop*, 2020, pp. 165–169.

[4] A. Politis, S. Adavanne, D. Krause, A. Deleforge, P. Srivastava, and T. Virtanen, "A dataset of dynamic reverberant sound scenes with directional interferers for sound event localization and detection," in *Proc. DCASE 2021 Workshop*, 2021, pp. 125–129.

[5] A. Politis, K. Shimada, P. Sudarsanam, S. Adavanne, D. Krause, Y. Koyama, N. Takahashi, S. Takahashi, Y. Mitsufuji, and T. Virtanen, "STARSS22: A dataset of spatial recordings of real scenes with spatiotemporal annotations of sound events," in *arXiv:2206.01948*, 2022.

[6] Y. Cao, T. Iqbal, Q. Kong, Y. Zhong, W. Wang, and M. D. Plumbley, "Event-independent network for polyphonic sound event localization and detection," in *Proc. DCASE 2020 Workshop*, 2020, pp. 11–15.

[7] Y. Cao, T. Iqbal, Q. Kong, F. An, W. Wang, and M. D. Plumbley, "An improved event-independent network for polyphonic sound event localization and detection," in *Proc. IEEE ICASSP 2021*, 2021, pp. 885–889.

[8] J. Hu, Y. Cao, M. Wu, Q. Kong, F. Yang, M. D. Plumbley, and J. Yang, "A track-wise ensemble event independent network for polyphonic sound event localization and detection," in *Proc. IEEE ICASSP 2022*, 2022, pp. 9196–9200.

[9] D. Hendrycks, N. Mu, E. D. Cubuk, B. Zoph, J. Gilmer, and B. Lakshminarayanan, "AugMix: A simple data processing method to improve robustness and uncertainty," in *Proc. ICLR 2020*, 2020.

[10] J. F. Gemmeke, D. P. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio Set: An ontology and human-labeled dataset for audio events," in *Proc. IEEE ICASSP 2017*, 2017, pp. 776–780.

[11] E. Fonseca, X. Favory, J. Pons, F. Font, and X. Serra, "FSD50K: an open dataset of human-labeled sound events," *IEEE/ACM Trans. on Audio, Speech, and Lang. Process.*, vol. 30, pp. 829–852, 2021.

[12] P.-A. Grumiaux, S. Kitić, L. Girin, and A. Guérin, "A survey of sound source localization with deep learning methods," *The Journal of the Acoustical Society of America*, vol. 152, no. 1, pp. 107–151, 2022.

[13] A. Gulati, J. Qin, C. Chiu, N. Parmar, Y. Zhang, J. Yu, W. Han, S. Wang, Z. Zhang, Y. Wu, and R. Pang, "Conformer: Convolution-augmented transformer for speech recognition," in *Proc. Interspeech 2020*, 2020, pp. 5036 – 5040.

[14] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *Proc. ICML 2020*, 2020, pp. 1597–1607.

[15] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," in *Proc. ICLR 2018*, 2018.

[16] Z. Zhong, L. Zheng, G. Kang, S. Li, and Y. Yang, "Random erasing data augmentation," in *Proc. of AAAI 2020*, vol. 34, no. 07, 2020, pp. 13 001–13 008.

[17] D. S. Park, W. Chan, Y. Zhang, C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "SpecAugment: A simple data augmentation method for automatic speech recognition," in *Proc. Interspeech 2019*, 2019, pp. 2613 – 2617.

[18] T. T. N. Nguyen, K. N. Watcharasupat, K. N. Nguyen, D. L. Jones, and W.-S. Gan, "SALSA: Spatial cue-augmented log-spectrogram features for polyphonic sound event localization and detection," *IEEE/ACM Trans. on Audio, Speech, and Lang. Process.*, vol. 30, pp. 1749–1762, 2022.

[19] L. Mazzon, Y. Koizumi, M. Yasuda, and N. Harada, "First order ambisonics domain spatial augmentation for DNN-based direction of arrival estimation," in *Proc. DCASE 2019 Workshop*, 2019, pp. 154–158.

[20] A. Politis, A. Mesaros, S. Adavanne, T. Heittola, and T. Virtanen, "Overview and evaluation of sound event localization and detection in DCASE 2019," *IEEE/ACM Trans. on Audio, Speech, and Lang. Process.*, vol. 29, pp. 684–698, 2020.

[21] A. Mesaros, S. Adavanne, A. Politis, T. Heittola, and T. Virtanen, "Joint measurement of localization and detection of sound events," in *Proc. IEEE WASPAA 2019*, 2019, pp. 333–337.

[22] A. Mesaros, T. Heittola, and T. Virtanen, "Metrics for polyphonic sound event detection," *Appl. Sci.*, vol. 6, no. 6, p. 162, 2016.

COLOC: CONDITIONED LOCALIZER AND CLASSIFIER FOR SOUND EVENT LOCALIZATION AND DETECTION

Sławomir Kapka, Jakub Tkaczuk*

Samsung R&D Institute Poland
Warsaw, Poland
s.kapka@samsung.com, j.tkaczuk@samsung.com

ABSTRACT

In this article we describe Conditioned Localizer and Classifier (CoLoC) which is a novel solution for Sound Event Localization and Detection (SELD). The solution constitutes of two stages: the localization is done first and is followed by classification conditioned by the output of the localizer. In order to resolve the problem of unknown number of sources we incorporate the idea borrowed from Sequential Set Generation (SSG). Models from both stages are SELDnet-like CRNNs, but with single outputs. Conducted reasoning shows that such two single output models are fit for SELD task. We show that our solution improves on the baseline system in most metrics on the STARSS22 Dataset.

Index Terms— DCASE2022 Challenge Task3, Sound Event Localization and Detection, CRNN, Ambisonics

1. INTRODUCTION

Sound Event Localization and Detection (SELD) is a complex task with many applications in robotics and surveillance. Since 2019, DCASE host annual Challenge in which Task3 is precisely SELD. This allows for gradual improvement of SELD systems over time.

The original SELDNet [1] has a drawback of being unable to detect multiple overlapping occurrences of events from the same class. We follow [2] and henceforth we call this problem *homogeneous overlap*. Thus, a new track-wise output format has been introduced in [2] incorporating Permutation Invariant Training (PIT) which precisely tackles the problem of homogeneous overlap. Since then, PIT was used in the improved version of DCASE2021 Top solution [3] as well as in the DCASE2022 Task3 Baseline System [4].

In this paper we propose a novel two-stage solution which incorporates class agnostic localizer based on Sequential Set Generation (SSG) and classifier conditioned on the output of the localizer. We will explain in the Section 2 how our solution solves the problem of homogeneous overlap without PIT. Our solution can be seen as an improved and refined version of our previous system from DCASE2019 [5]. In this solution the estimator of the number of active sources is included as a part of a conditional localizer, i.e. using SSG localizer we can retrieve the number of active sound sources in each time frame.

In the following Section 2 we describe the main components and the inference process of our method. In Section 3 we present a way how to train the components. Finally, in Section 4 we describe detailed results obtained on the STARSS22 Dataset [6] and compare our solution with the baseline system.

*Corresponding author.

2. PROPOSED METHOD

2.1. Motivation

From the perspective of statistical learning theory, the optimal solutions to the problem of Empirical Risk Minimalisation (ERM) are conditional probability $P(Y|X)$ and conditional expectation $E(Y|X)$ for classification with cross-entropy loss and for regression with L^2 loss respectively [7]. By abuse of language, we will call the general solution to the ERM problem simply as conditional probability and we will use notations $P(Y|X)$ and $E(Y|X)$ rather frivolously.

SELD is the problem of simultaneous localization and detection, so the optimal solution in each time frame may be modeled as joint probability:

$$P(\{c_i \wedge l_i\}_{i=1..k}|X),$$

where c_i and l_i denote the class and the location of a detected event e_i, and $k \leq N$, where N is the maximal number of overlapping events. The first problem rises from the fact that models do not output sets. The most popular workaround is to force a model to output a list of length N denoting the individual tracks with class and location information, where some tracks may be empty.

In our solution we first localize all audio events using the Sequential Set Generation method, which simultaneously allows us to estimate the number of active sound sources. More precisely, the localizer returns, in a sequential manner, directions of arrivals (DOAs) conditioned by DOAs which it already returned starting from the empty set $\emptyset$:

$$\begin{aligned} l_1 &= E(l|X, \emptyset) \\ l_2 &= E(l|X, \{l_1\}) \\ l_3 &= E(l|X, \{l_1, l_2\}) \\ &\dots \\ l_k &= E(l|X, \{l_i\}_{i=1..k-1}) \\ \tau &= E(l|X, \{l_i\}_{i=1..k}), \end{aligned}$$

where τ is a special token denoting that there is no more events. In our case τ is set to be an origin $\mathbf{0}$ from $\mathbb{R}^3$.

Based on the output from the localizer we then classify the event corresponding to this particular DOA. Thus informally

$$P(c_i \wedge l_i|X) = P(c_i|X, l_i) \cdot P(l_i|X),$$

which indicates that we could resolve the SELD task given an SSG localizer $E(l|X, \{l_i\}_i)$ and location-conditioned classifier $P(c|X, l)$.

The only edge case where the above solution may fail is when two or more events overlap spatially. However, in practice outputs from models have temporal context which should resolve this issue.

2.2. Stacked-Tracks

Let us consider a chunk of First Order Ambisonics (FOA) audio format in which there are at most N overlapping audio events. With such audio we associate meta information about the location and classes of the occurrences in each time frame. By location we mean xyz Cartesian coordinates on a unit sphere, and by class we mean one of the K predefined classes. We aim to construct an $N \times T \times 4$ tensor *stacked-tracks*, where N is the maximal number of overlapping events, T denotes the number of time bins and the last dimension contains information about locations and classes. The tensor contains all available meta information in a convenient form. To obtain stacked-tracks, we iterate sequentially over occurrences and stack them from bottom to top. If an event terminates in some track, then all events from the above tracks are stacked down. All remaining empty cells are filled with zeros in xyz coordinates and with a new class index K which is interpreted as the lack of any of predefined event, i.e. silence or unknown event. Figure 1 presents an example of how to obtain stacked-tracks. Since the tracks can be permuted before stacking, the stacked-tracks are not unique; we will exploit this during training later.

Tracks	Time Frames							
	0	1	2	3	4	5	6	7
T4		0.2 0.7 -0.2 3	0.2 0.8 -0.1 3					
T3			0.5 -0.7 0.5 7	0.5 -0.7 0.5 7	0.5 -0.7 0.5 7	0.6 -0.7 0.4 7	0.6 -0.7 0.4 7	
T2	-0.5 0.6 0.3 3	-0.4 0.7 0.3 3	-0.4 0.7 0.3 3	-0.4 0.8 0.3 3	-0.3 0.8 0.4 3			
T1							0.7 0.5 -0.5 11	0.7 0.5 -0.5 11
T0				-0.9 0.2 0.1 8	-0.9 0.2 0.1 8	-0.8 0.2 0.2 8		

↓↓↓ Stacking ↓↓↓

Tracks	Time Frames							
	0	1	2	3	4	5	6	7
ST2	0.0 0.0 0.0 13	0.0 0.0 0.0 13	0.2 0.8 -0.1 3	0.5 -0.7 0.5 7	0.5 -0.7 0.5 7	0.0 0.0 0.0 13	0.0 0.0 0.0 13	0.0 0.0 0.0 13
ST1	0.0 0.0 0.0 13	0.2 0.7 -0.2 3	0.5 -0.7 0.5 7	-0.4 0.8 0.3 3	-0.3 0.8 0.4 3	0.6 -0.7 0.4 7	0.6 -0.7 0.4 7	0.0 0.0 0.0 13
ST0	-0.5 0.6 0.3 3	-0.4 0.7 0.3 3	-0.4 0.7 0.3 3	-0.9 0.2 0.1 8	-0.9 0.2 0.1 8	-0.8 0.2 0.2 8	0.7 0.5 -0.5 11	0.7 0.5 -0.5 11

Figure 1: An illustration of obtaining stacked-tracks from regular tracks. In this example there are five tracks with maximal overlap of three events. In this example there are 13 predefined classes, thus cells with coordinates at origin get a class label with index 13.

2.3. Self-Conditioned SSG Localizer

The idea behind self-conditioned SSG localizer is to recursively localize all events from known classes without specifying class labels, i.e. it is class-agnostic. We will call this module simply as localizer. Our localizer consists of two trainable components:

- Localizer Encoder $L_{enc} : \mathbb{R}^3 \to \mathbb{R}^c$, where c is the hyperparameter denoting the number of new channels.
- Localizer Network $L_{net} : \mathbb{R}^{t \times F \times C} \to \mathbb{R}^{T \times 3}$, where $C = c + c_f$ given that c_f is the number of feature channels, where t, F are numbers of time and frequency bins respectively, and where T is the number of label time bins with meta resolution (in our case $t = 5 \cdot T$).

The core idea behind our self-conditioned SSG localizer is to recursively obtain all DOAs in all time frames. We start from the blank stacked-tracks and in each step we successively fill tracks in such a way that L_{enc} encodes previously detected DOAs which L_{net} should ignore. For that, we introduce below a modified version of the SSG method:

Step 0. During the first step we set xyz to zeros in each time frame (is is convenient to look at it as putting empty row below ST0 in the Figure 1). Hence, we end up with the vector of size $t \times 3$. Then we apply L_{enc} to it frame-wise to obtain tensor of size $T \times c$. Next, we repeat the tensor so many times to obtain vector which could be stacked with extracted features from audio channels. Thus, we obtain tensor of size $t \times F \times c$ which we concatenate with audio features tensor of size $t \times F \times c_f$ to obtain $t \times F \times C$ tensor. Finally, we feed it to L_{net} to get the tensor of size $T \times 3$ denoting the predicted DOAs in each frame. When there is no active sound sources in a frame we expect it to return origin (i.e. $xyz = \mathbf{0}$) similarly as it was done in [8]. On the other hand, when there are some events we expect the model to return xyz coordinates of any of the occurring events. In the end we simply threshold length of vectors to decide whether there has been any event. If the length is greater than 0.5, then we conclude that there is an event, otherwise we put zeros. In summary, we obtain information about DOAs in each time-frame which we put in a first row ST0 in the blank stacked-tracks.

Step 1. During the second iteration, the output from the first step is fed into the L_{enc}. If in the first step in some frames an event was detected, then we encode these predicted DOAs, $\mathbf{0}$ otherwise. In the same way as in the first step, the encoded tensor is repeated and stacked with features channels. Now, we expect the L_{net} model to output in each time-frame the DOA of second event if there is any and origin if there is none, or the already detected one is the only one. Analogously as in the first step, we threshold the length of output to decide if there is a new event. We stack the obtained results in stacked-track ST1 on top of ST0.

Step n. Lets say we already have $n-1$ stacked-tracks.. The aim of this step is to get the DOAs of events which haven't been already localized. For these frames where stacked-track no. $n-1$ denotes that there is less then $n-1$ events, we encode the origin via L_{enc}. Otherwise, for these frames where we acquired $n-1$ DOAs we encode them individually using L_{enc} and for each such frame we average $n-1$ obtained embeddings from $\mathbb{R}^c$. Note that thanks to average-pooling (in contrast to max-pooling as it was resolved in [9]) we potentially preserve some additional information about DOAs count. So, we encoded the set of all previous DOAs which L_{net} should ignore. Analogously as in previous steps, encoded embeddings are repeated and stack with extracted audio features. Next,

we predict DOAs of new events if there are some, threshold lengths, and finally obtain a new stacked-track.

One may ask in which order the localizer should return DOAs. We did not impose any restrictions on that and let the model learn its own internal hidden order. It will be evident later from the training process how it is done. In essence, in each frame the localizer returns DOAs by DOAs in its own fashion until all sound sources have been localized.

2.4. Location-Conditioned Classifier

Self-Conditioned SSG Localizer outputs information about DOAs in each time frame written in the stacked-frame format. Location-Conditioned Classifier simply takes each row from stacked-tracks and in each time-frame outputs probabilities of predicted classes conditioned by DOAs. If the localizer predicted that there is no event in a frame, the classifier is conditioned by the origin and it is expected to predict additional special class with index K, where K is the number of classes.

Our classifier similarly as localizer consists of two trainable components:

- Classifier Encoder $C_{enc} : \mathbb{R}^3 \to \mathbb{R}^c$, where c is the hyperparameter denoting the number of new channels (may be different than the one in localizer, but for simplicity we set it to be the same)
- Classifier Network $C_{net} : \mathbb{R}^{t\times F\times C} \to \mathbb{R}^{T\times(K+1)}$, where notation is the same as in the localizer above and where K denotes the number of classes.

3. TRAINING PROCESS

In our solution localizer and classifier are trained completely separately.

3.1. Self-Conditioned SSG Localizer

Let's say we have an audio chunk and associated meta with N stacked-tracks, where N denotes the maximal number of overlapping events. We select a random integer r from 0 to $N-2$ in a uniform way and split stacked-tracks into two parts:

- Conditioning part containing tracks from 0 to $r-1$,
- Target part containing tracks from r to $N-1$.

The aim of this splitting is to imitate the r'th iteration from the inference, by hiding stacked-tracks with indices $\geq r$.

We feed the Conditioning part into L_{enc} frame-wise in the following way: if there is some event in $r-1$'th stacked-track, then we encode all DOAs via L_{enc} and avarage pool the embeddings. Otherwise, we encode $\mathbf{0}$ via $L_{enc.}$ We then repeat obtained embeddings so many times to be able to stack them with audio features and we feed an acquired tensor through L_{net}. Thus, in each time-frame s we obtain new xyz coordinates which we denote by l^s_{pred}. We compare the predicted coordinates with the ones from the Target part. In each time frame we compute a $L_{1.5}$ distance between predicted DOAs and the ground truth DOAs from Target part and set the minimum value as our loss. If there are no more active sound sources in Target part we enforce the target to be a zero vector by minimizing $L_{1.5}$ norm of the predicted DOA. I.e.

$$Loss^{r,k}_{loc}(\boldsymbol{l}^s_{gt}, l^s_{pred}) = \begin{cases} \min_{i=r..k} \left\| l^s_{gt\,i} - l^s_{pred} \right\|_{1.5} & \text{if } k \geq r \\ \left\| l^s_{pred} \right\|_{1.5} & \text{if } k < r \end{cases},$$

where k is the index of the last nonzero DOAs from the Target part of stacked-tracks. During training, given a batch of size B of audio-meta pairs we select random r for each item in the batch, compute $Loss^{r,k}_{loc}$ loss (frame-wise and item-wise) and average it over all $k \leq N$ and $r \leq k$. I.e. the final loss between $[\boldsymbol{l}_{gt}]^t_b$ and $[l_{pred}]^t_b$ is

$$\frac{2}{N(N+1)+2} \sum_{k\leq N} \sum_{r\leq k} \frac{1}{|T_{r,k}|} \sum_{s\in T_{r,k}} Loss^{r,k}_{loc}(\boldsymbol{l}^s_{gt}, l^s_{pred}),$$

where $T_{r,k}$ stand for time-frames among whole batch and where k, r were sampled according to the rule described above.

One may ask why we decided to select L_p norm with $p = 1.5$. Since for $p = 2$ the optimal solution is the expected value, there is a possible risk of ignoring the conditioning L_{enc} and averaging xyz outputs in the case of multiple overlapping sound sources. Conversely, for $p = 1$ the optimal solution is the median which may be too "sharp" decision making. Thus $p = 1.5$.

3.2. Location-Conditioned Classifier

In the same manner as in the case of the localizer, let's assume that we have some audio-meta pair from a chunk of the sound signal and let N denote the maximal number of overlapping events. We select random number r from 0 to $N-1$ and we select r'th row from stacked-meta containing DOA information. We aim to output classes associated with these DOAs. In each time-frame we encode the DOAs from the selected row via C_{enc}. Then we repeat the encoded tensor and stack it with audio features. We forward it through C_{net} to obtain $K+1$ scores in each time-frame s associated with K classes and one score denoting unknown class or lack of any event. We then simply compare predicted probability scores p^s_{pred} with ground truth classes c^s_{gt} from the selected stacked-track. Since there is great imbalance of classes due to the fact that class associated with unknown event is over-represented, we used focal loss [10] with $\gamma = 1$ instead of regular cross-entropy. I.e.

$$Loss_{cls}(c^s_{gt}, p^s_{pred}) = -(1 - p^s_{c,pred}) \log(p^s_{c,pred}).$$

For a batch of size B and T time-frames per item we simply average everything to obtain the final loss for a back-propagation.

3.3. Angle perturbation

One drawback of our approach is the error propagation. Since localizer and classifier are conditioned by the output of the localizer, the error in DOA predictions downgrade the quality of outputs in the next steps. To partly resolve that issue we decided to perturb angles fed to L_{enc} and C_{enc} during the training as well during the inference. We decided to randomly perturb azimuth and elevation by 5 degrees in each time-frame from meta.

3.4. Stack-track permutation

In order to increase the number of training samples we could permute ordinary tracks before stacking into stacked-tracks. However, in our case it was more convenient to operate on stacked-tracks. Thus, for each stacked-track tensor we permute the tracks and stack them back.

In our solution we constrained to $N = 3$, i.e. we allow up to 3 overlapping audio events. We simply ignore the rest.

3.5. Model Architecture

For L_{net} and C_{net} we used the SELDnet-like architecture introduced in [1]. More precisely, for localizer and classifier we took

the baseline model [6] and changed the number of filters in convolutional blocks from 64 to 128. After the last time-distributed dense layer we put at the end another time-distributed dense layer with 3 outputs with tanh activation for localizer and with 13 outputs with softmax activation for classifier.

We set L_{enc} and C_{enc} to be single dense layers with 3 inputs and $c = 5$ outputs with tanh activation.

As for the models complexity, L_{net} and C_{net} have slightly above 2.3 million parameters each while L_{enc} and C_{enc} have just 20 parameters each.

3.6. Complexity

During the training of the localizer, we need to compute up to $\frac{N\cdot(N+1)}{2}+1$ components to the loss given that the maximal number of overlapping sound sources is N. For the classifier, the complexity scales linearly with N. This contrasts with PIT, where in principle we need to compute $N!$ components to the loss. During the inference as described in Section 2, our method requires up to N steps for both localizer and classifier.

3.7. Features and augmentations

We used 24kHz FOA format for our nets. We extracted complex spectrograms from each four FOA channels using Short Time Fourier Transform (STFT) with $n_{fft} = 1024$ and Hanning window and hop length of 960 and 480 respectively. From each obtained spectrogram we acquire the log-power spectrogram and the phase spectrogram. For the last three channels we used intensity vectors as it was done in [11]. In summary, from each FOA audio signal we acquire 11 audio features of size $t \times 513$, where t is the number of time bins from the STFT. In our case we randomly selected 5s audio chunks from the recordings which constitutes of 250 time bins.

For data augmentation we used volume perturbation by selecting a random number between 0.5 and 1.5 and multiplying all audio channels by that number. We also used FOA domain spatial augmentation [12] to augment every fourth audio-meta pair.

We trained localizer and classifier using Adam optimizer [13] with default parameters except learning rate which we set to be 0.0005. We trained both models for half a million batches of size 96.

4. EVALUATION

4.1. Metrics

The DCASE2022 Challenge Task3 organizers provided two types of datasets for the development stage [6]:

- Synth: 1200 one-minute synthesized mixtures from collected SRIRs and selected sound events from FSD50K [14]
- STARSS22: 292 minutes of real recordings simulating real life scenarios gathered in 11 rooms in Tokyo and Tempere.

STARSS22 is further split into train and test folds. In this section we will discuss results on the test fold. During the training, for every batch we sampled half of the recordings from Synth dataset and half from the train split from STARSS22, utilizing all the data. We noticed that if we do not use synthetic dataset the scores drop drastically.

For localizer we first report the average DOA error in angles. In Table 1 we show average DOA errors in multiple cases differentiating between number of active sound sources and number of DOAs in conditioning. As excepted, the more DOAs in conditioning, the larger the error. However, what is interesting is the fact that the more sound sources, the more accurate the model is in detecting the first few DOAs.

Table 1: Dependence of the localizer's average DOA error (in degrees) on the number of active sources (noas) and the number of DOAs in conditioning (#cond) on STARSS22 test split.

noas \ #cond	0	1	2
1	23		
2	20	33	
3	14	19	51

For the classifier we report conditional accuracy (CAcc) in which we count the class c with maximal probability $P(c|X, l)$ correct if that class corresponds to the conditioning DOA l. Otherwise we treat it as an incorrect prediction. On the STARSS22 test split we achieved CAcc of 68%. We also kept track of the number of frames where the classifier misses known classes and of frames where the classifier predicted a known class where there is none, but for those cases the classifier achieve almost perfect scores.

Finally, we report our results on official DCASE2022 Task3 metrics [6], namely: the localization-dependent error rate ER_{20°, F1-score F_{20°, the localization error LE_{CD} and the localization recall LR_{CD}. We compare the baseline system with two versions of our solution. In the first one we will predict up to 3 DOAs and corresponding classes, and in the second one we terminate inference on 2 events. We denote these solutions as `max_ov3` and `max_ov2` respectively. Note that we used the very same models in both solutions, only the inference changes. We summarise the results in Table 2. In most metrics our solution outperforms the baseline system. The only one when our solution is lacking is the error rate. Furthermore, the error rate is worse when we try to infer more events. We speculate that this is due to the fact that the localizer is very inaccurate in later steps (see Table 1), which may generate many false positives.

Table 2: Official metrics; the **boldface** denotes the best scores.

	ER_{20°	F_{20°	LE_{CD}	LR_{CD}
`Baseline`	**0.71**	21%	29.3°	46%
`max_ov3`	0.85	32%	24.7°	**51**%
`max_ov2`	0.76	**33**%	**24.6°**	49%

5. CONCLUSION

In this paper we presented alternative solution to the SELD problem. Our solution uses our custom SSG method to determine DOAs one by one and as a result to determine number of sound sources in each time frame. The localizer is followed by the location-conditioned classifier. The performance of our method is comparible to the DCASE2022 Task3 baseline system which uses PIT. Thus, we imagine that in the future SSG and PIT may be combined together to obtain the best of both worlds.

6. REFERENCES

[1] S. Adavanne, A. Politis, J. Nikunen, and T. Virtanen, "Sound event localization and detection of overlapping sources using convolutional recurrent neural networks," *IEEE Journal of Selected Topics in Signal Processing*, vol. 13, no. 1, pp. 34–48, 2018.

[2] Y. CAO, T. IQBAL, Q. Kong, Y. ZHONG, W. WANG, and M. D. PLUMBLEY, "Event-independent network for polyphonic sound event localization and detection," in *Detection and Classification of Acoustic Scenes and Events 2020 Workshop (DCASE 2020)*. University of Surrey.

[3] K. Shimada, Y. Koyama, S. Takahashi, N. Takahashi, E. Tsunoo, and Y. Mitsufuji, "Multi-accdoa: Localizing and detecting overlapping sounds from the same class with auxiliary duplicating permutation invariant training," in *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022, pp. 316–320.

[4] https://dcase.community/challenge2022.

[5] S. Kapka and M. Lewandowski, "Sound source detection, localization and classification using consecutive ensemble of crnn models," in *Acoustic Scenes and Events 2019 Workshop (DCASE2019)*, 2019, p. 119.

[6] A. Politis, K. Shimada, P. Sudarsanam, S. Adavanne, D. Krause, Y. Koyama, N. Takahashi, S. Takahashi, Y. Mitsufuji, and T. Virtanen, "Starss22: A dataset of spatial recordings of real scenes with spatiotemporal annotations of sound events," 2022. [Online]. Available: https://arxiv.org/abs/2206.01948

[7] V. Vapnik, *The nature of statistical learning theory*. Springer science & business media, 1999.

[8] K. Shimada, Y. Koyama, N. Takahashi, S. Takahashi, and Y. Mitsufuji, "Accdoa: Activity-coupled cartesian direction of arrival representation for sound event localization and detection," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 915–919.

[9] M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Poczos, R. R. Salakhutdinov, and A. J. Smola, "Deep sets," *Advances in neural information processing systems*, vol. 30, 2017.

[10] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2980–2988.

[11] Y. Cao, T. Iqbal, Q. Kong, M. Galindo, W. Wang, and M. Plumbley, "Two-stage sound event localization and detection using intensity vector and generalized cross-correlation," DCASE2019 Challenge, Tech. Rep., June 2019.

[12] L. Mazzon, M. Yasuda, Y. Koizumi, and N. Harada, "Sound event localization and detection using foa domain spatial augmentation," in *Proc. of the 4th Workshop on Detection and Classification of Acoustic Scenes and Events (DCASE)*, 2019.

[13] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014. [Online]. Available: https://arxiv.org/abs/1412.6980

[14] E. Fonseca, X. Favory, J. Pons, F. Font, and X. Serra, "Fsd50k: an open dataset of human-labeled sound events," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 30, pp. 829–852, 2021.

MODEL TRAINING THAT PRIORITIZES RARE OVERLAPPED LABELS FOR POLYPHONIC SOUND EVENT DETECTION

Rie Koga[1]*, Sawa Takamuku*[1]*, Keisuke Imoto*[2]*, Naotake Natori*[1]

[1] AISIN CORPORATION, 1-1-20 Aomi, Koto-ku, Tokyo, Japan,
{rie.yamada, sawa.takamuku, naotake.natori}@aisin.co.jp
[2] Doshisha University, 1-3 Tatara Miyakodani, Kyotanabe, Kyoto, Japan,
keisuke.imoto@ieee.org

ABSTRACT

In this study, we propose a model training method for polyphonic sound event detection (polyphonic SED) that prioritizes rare event label frames during multiple overlapping sound events. Multi-label classification typically utilized in polyphonic SED often fails to recognize such events. To overcome this problem, the proposed method is designed to represent event overlaps of rare labels easily without a complicated network structure. During model training, we periodically apply either binary cross-entropy loss (BCE) for multi-label classification or softmax cross-entropy loss (Softmax-CE) for multi-class classification. When multi-class classification is performed using Softmax-CE, the labels of the overlapping frames are reconstructed from the target labels to include the rarest ones and exclude the frequent ones. The model was evaluated on strongly labeled AudioSet data, from which only human voice segments were extracted. The proposed method achieves an improvement of 0.23 percentage points over the baseline, which only used the BCE, in terms of the mean average precision. In particular, the proposed method outperforms the baseline with respect to rare labels, with an average precision of 1.18 percentage points. The experimental results also demonstrate the effectiveness of the proposed method for both overlap of sound events and rare labels.

***Index Terms*—** Polyphonic sound event detection, multi-label classification, multi-class classification

1. INTRODUCTION

Due to the advancements in deep learning, sound event detection (SED), which is a technique used for estimating the type and interval (onset and offset times) of sound events present in an acoustic signal, has recently attracted attention. Additionally, shared mobility services have become ubiquitous in many cities worldwide. For safety, they require surveillance of both the drivers and passengers inside the vehicles [1, 2]. In an in-vehicle surveillance system, various sound events must be detected to understand what is occurring inside the vehicle. Therefore, this study focused on human voice SED for an in-vehicle surveillance system based on human voice signals.

Some DCASE competitions [3, 4] have previously dealt with the sound of human speech or crying babies, where target sounds are often overlapped, whereas real-world data often suffer from extreme imbalances between classes as well as overlapping sound events. For example, as shown in Fig. 1, in the strongly labeled AudioSet [5] dataset annotated using real-world data, despite focusing on the top seven classes with the highest number of event frames among the 14 "human voice" classes, the number of frames among the classes is imbalanced, with a ratio of approximately 100 to 1 between the most common and the rarest.

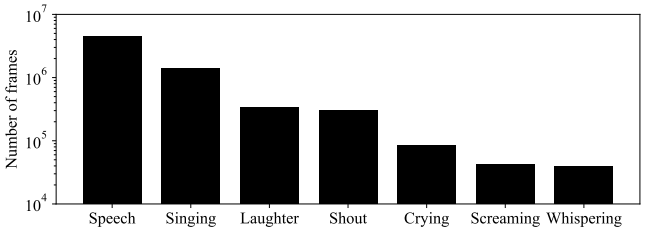


Figure 1: Number of frames in training data.

Many studies have treated SED as a multi-label classification problem for handling overlapping events that often cause detection errors [6, 7, 8]. [9] represented event overlaps by linking a bivariate probability distribution based on time and frequency with class-wise hidden Markov models. In [10], a non-negative matrix decomposition-based method that jointly trained a dictionary and a multinomial logistic regression classifier was used to manage the overlap of sounds. In [11], an event independent network for SED and localization was developed with a track-wise output. In polyphonic sound event detection (polyphonic SED) with deep learning, [12] performed multi-class classification by considering all possible overlapping event combinations as classes. However, the model architecture requires significant modification to manage a multi-class multi-tasking problem. Therefore, the conventional method [12] cannot be applied to the current polyphonic SED system without any modification of the network architecture.

Binary cross-entropy (BCE) loss is often employed as the loss function of multi-label classification in polyphonic SED. However, SED using BCE often falls into imbalance between sound event classes when training an SED model. Therefore, when applied to real-world data, the accuracy of rare class event detection decreases. Specifically, accurately detecting anomalous or rare sounds such as "Screaming" is more important than detecting common sounds such as "Speech," as shown in Fig. 1. Several loss functions that are effective for imbalanced data have been proposed in polyphonic SED. [13] proposed asymmetric focal loss and focal batch Tversky loss; however, these mainly address the imbalance problem between negative and positive samples. [14] proposed time-balanced focal loss, which is highly dependent on the dataset because the class weights used in the loss function are adjusted as hyperparameters.

Therefore, without modifying the original model architecture

or preparing the class weights, we propose a method that periodically uses multi-label classification based on BCE and multi-class classification to prioritize rare classes as target labels when sound events overlap.

The contributions of this study can be summarized as follows:

- We propose a new model training method for detecting overlapped and rare sound events. The proposed method combines multi-class classification, in which rare classes are preferentially learned as target labels, along with multi-label classification. We then confirm the efficacy of this method.
- We reconstructed a strongly labeled AudioSet using seven sound event classes with "Human voice" at the upper level. We conducted a baseline evaluation for an SED task covering multiple types of human vocalization with these classes.

2. DATASET

Based on AudioSet's strong labels [5], we created a new dataset comprising 10 seconds of audio taken from the soundtrack of a YouTube video, with approximately 67,000 clips for training and 18,000 clips for evaluation. The strongly labeled AudioSet ontology is a hierarchy of 356 sound event classes. The sound classes selected for this study were the following seven event classes within the "Human voice" class: "Speech," "Singing," "Laughter," "Shout," "Crying, sobbing (Crying)," "Screaming," and "Whispering." In cases where the selected classes have subclasses, the subclasses are merged into the superclass. For example, subclasses "Baby cry, infant cry" and "Whimper" are merged into a superclass "Crying." Sound clips with other sound events in "Human voice," such as "Humming" or "Yawn," were not used in the dataset because there were few events in each class.

When sound clips contain other sound events from the category non-"Human voice," such as "Music" or "Hands," the clips were still used. However, these sound events were only background noise, that is, they were not used as target labels. After extracting the dataset to contain the selected sound class for each audio clip, the dataset contained 50,650 sound clips for training and 8,747 sound clips for evaluation. Note that in this study, rare labels ("Screaming" and "Whispering") were defined as appearing with approximately 1% of the frequency of the most frequent label.

3. PROPOSED METHOD

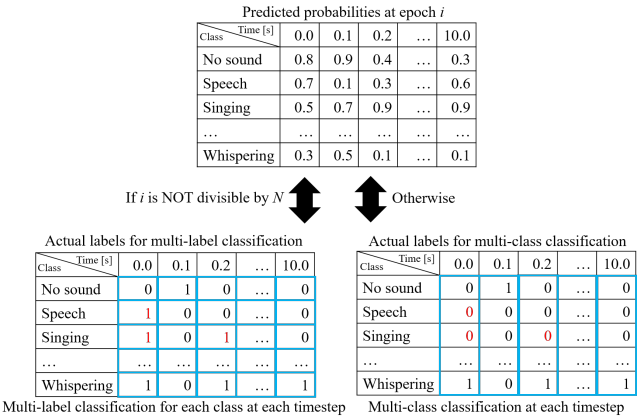
In this section, we first describe the loss functions used in this study for the multi-label and multi-class classification tasks. Next, we discuss a new model training method combining those loss functions. Finally, we describe a method of label selection for multi-class classification using polyphonic SED.

3.1. Loss function

Generally, a sigmoid activation function-based BCE is employed in training polyphonic SED models.

$$f(s)_{i,j} = \frac{1}{1+e^{-s_{i,j}}} \quad (1)$$

$$\mathcal{L}_{\mathrm{BCE}} = -\sum_{i}^{C}\sum_{j}^{T}\{y_{i,j}\log(f(s)_{i,j}) + (1-y_{i,j})\log(1-f(s)_{i,j})\}, \quad (2)$$



Class \ Time [s]	0.0	0.1	0.2	…	10.0
No sound	0.8	0.9	0.4	…	0.3
Speech	0.7	0.1	0.3	…	0.6
Singing	0.5	0.7	0.9	…	0.9
…	…	…	…	…	…
Whispering	0.3	0.5	0.1	…	0.1

Class \ Time [s]	0.0	0.1	0.2	…	10.0
No sound	0	1	0	…	0
Speech	1	0	0	…	0
Singing	1	0	1	…	0
…	…	…	…	…	…
Whispering	1	0	1	…	1

Class \ Time [s]	0.0	0.1	0.2	…	10.0
No sound	0	1	0	…	0
Speech	0	0	0	…	0
Singing	0	0	0	…	0
…	…	…	…	…	…
Whispering	1	0	1	…	1

Figure 2: Overview of the proposed model training method.

where f is the sigmoid function, $s_{i,j}$ is the ith class's jth time frame logit, $y_{i,j}$ is the ith class's jth time frame's target label, C is the total number of classes, and T is the total number of time frames.

Conversely, a softmax activation function-based cross-entropy (Softmax-CE) loss for a multi-class classification is employed in monophonic SEDs, to choose one event from multiple sound event classes.

$$g(s)_{i,j} = \frac{e^{s_{i,j}}}{\sum_{k}^{C} e^{s_{k,j}}} \quad (3)$$

$$\mathcal{L}_{\mathrm{Softmax-CE}} = -\sum_{i}^{C}\sum_{j}^{T} y_{i,j}\log(g(s)_{i,j}), \quad (4)$$

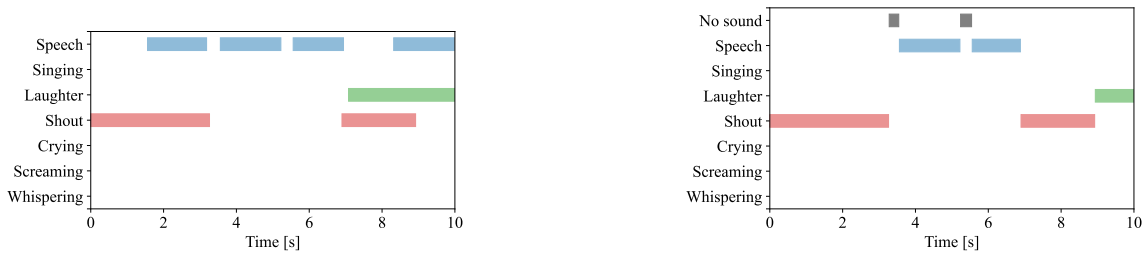
where g is the softmax function.

3.2. Training process

In this study, we applied alternately either BCE or Softmax-CE within a defined period, as shown as Fig. 2. Specifically, Softmax-CE was applied while using multi-class classification every N epoch. When the model was trained by Softmax-CE, it was trained by BCE and saved at the next epoch. Then, the model for evaluation was only used at the epoch with the minimum validation loss. For example, when multi-class classification was performed at every third epoch, i.e., at the 3, 6, 9, ..., 3*i epochs, the validation loss was monitored at the 4, 7, 10, ..., (3*i+1) epochs, with i being a positive integer.

3.3. Label selection for multi-class classification

Two problems are often encountered when performing multi-class classification for polyphonic SED because multi-class classification always requires one target label during loss computation. The first problem is determining which label to allocate when multiple sound events occur simultaneously. The second problem is determining which label to allocate when none of the target sound events occur. To solve these problems, we propose a new method of label selection using the sparsity of sound events. Specifically, we prioritize the rarest label for multiple events and define a new class label for no events. The details are discussed below.



(a) multi-label classification.

(b) multi-class classification.

Figure 3: Overview of the target labels. (a) multi-label classification and (b) multi-class classification.

Table 1: Number of frames for each sound event.

Event class	The number of frames	
	Multi-label	Multi-class
Speech	9,083,336	8,627,958
Singing	2,781,533	2,690,509
Laughter	683,036	666,186
Shout	612,058	607,208
Crying	174,120	173,731
Screaming	85,578	85,578
Whispering	80,194	80,194

Multiple events This section describes the method to allocate sound event labels when multiple overlapping sound events occur concurrently during a single clip, as shown in Fig. 3. In Fig. 3, multiple sound events overlap as follows.

- 1.6 - 3.2 seconds : Speech and **Shout**
- 7.1 - 8.3 seconds : Laughter and **Shout**
- 8.3 - 8.9 seconds : Speech, Laughter, and **Shout**
- 8.9 - 10.0 seconds : Speech and **Laughter**

We adopted the rarest label for each time frame in the clip. In the example in Fig. 3, when multi-class classification was performed, the labels in bold were used. The number of frames for each label in the training data when changing from the labels used in multi-label classification to those used in multi-class classification is shown in Table 1. Frequent labels such as "Speech" and "Singing" show a large decrease in the number of frames when compared with the number of rare labels.

No events In multi-label classification, BCE originally includes calculation of inactive frames. However, in multi-class classification, even when none of the seven target labels exist in a frame, one class must be set as the target label. Therefore, a new class "No sound" was created and allocated to time frames containing no target class. Fig. 3(b) shows the "No sound" class with thick black lines between 3.3 - 3.5 seconds and 5.2 - 5.5 seconds.

4. EXPERIMENT

4.1. Experimental setups

The AudioSet sound clips were downloaded from YouTube. These sounds were mostly monaural. The left and right sides of the stereophonic sounds were averaged to produce monaural sounds. The sounds were resampled to 44.1 kHz, as previously configured [7]. The sounds were then converted into a logmel scale of $F = 64$ filters calculated every 40 milliseconds with a hop size of 20 milliseconds.

Table 2: Model architecture. The kernel sizes of the convolutional and pooling layer are denoted as "Conv (kernel size)" and "Max Pooling (kernel size)," respectively. The number of attention heads is denoted as "Transformer Encoder (number of attention heads)."

Conv3	RB
Log-mel spectrogram 500 frames × 64 mel bins	
Conv (3 × 3) BN, ReLU, Dr Max Pooling (8 × 1)	Conv (3 × 3) BN, ReLU ResBlock
Conv (3 × 3) BN, ReLU, Dr Max Pooling (4 × 1)	ResBlock
Conv (3 × 3) BN, ReLU, Dr Max Pooling (2 × 1)	ResBlock
(Transformer Encoder (32)) × 2	
FC, Sigmoid	

Inspired by [13], a convolutional neural network (CNN)-transformer-based network was used as the model architecture. This architecture performs better than the CNN-biGRU-based network, which is widely used in SED [7, 15, 16]. The model architecture is shown in Table 2. The system has two types of CNN backbones: one with three Convolution layers (Conv3) and the other with three ResBlocks (RB). The parameters of the convolutional layers in RB are the same as those of Conv3.

The models were trained using the RAdam optimizer [17] with a learning rate of 0.001. Early stopping was implemented after 50 epochs if no improvement on validation loss was noted.

As evaluation metrics, we used the mean average precision (mAP), the micro average precision (micro-AP), and the frame-based macro- and micro-Fscores with a threshold for prediction of 0.5. Even when the proposed method was deployed with eight event classes including "No sound," we evaluated them using only the original seven classes. In this study, we used eight classes when performing both the multi-label and multi-class classification using the proposed method.

Table 3: Average SED performance for two backbones.

Method	Conv3				RB			
	mAP	macro-Fscore	micro-AP	micro-Fscore	mAP	macro-Fscore	micro-AP	micro-Fscore
baseline	51.96%	38.16%	83.39%	73.83%	58.04%	50.06%	85.48%	77.11%
AFL	50.92%	36.92%	82.90%	73.19%	59.05%	52.15%	85.66%	77.56%
e0	51.60%	39.15%	83.25%	72.72%	58.22%	49.71%	85.35%	76.82%
e1	51.51%	37.03%	79.59%	50.88%	56.28%	32.97%	80.39%	47.36%
e2 (proposed)	**52.19%**	**40.41%**	**83.55%**	**74.65%**	**60.32%**	**53.40%**	**86.65%**	78.52%
e3 (proposed)	51.63%	38.20%	83.11%	73.49%	59.51%	50.96%	86.47%	78.38%
e4 (proposed)	51.54%	38.59%	83.27%	73.48%	59.89%	52.70%	86.59%	**78.58%**
e5 (proposed)	51.28%	38.25%	83.14%	73.49%	59.66%	51.35%	86.36%	78.28%

Table 4: Average SED performance of the rare-event labels for the two backbones.

	Method	Conv3		RB	
		AP	Fscore	AP	Fscore
Screaming	baseline	17.29%	3.57%	27.04%	**17.03%**
	AFL	16.46%	2.92%	28.08%	15.14%
	e2 (proposed)	**18.73%**	**4.01%**	**28.37%**	13.43%
Whispering	baseline	55.86%	42.91%	64.68%	61.64%
	AFL	55.03%	40.55%	65.67%	60.81%
	e2 (proposed)	**56.79%**	**45.19%**	**67.34%**	**65.64%**
Avg.	baseline	36.58%	23.24%	45.86%	39.34%
	AFL	35.74%	21.73%	46.87%	37.98%
	e2 (proposed)	**37.76%**	**24.60%**	**47.85%**	**39.54%**

4.2. Experimental results

Table 3 shows the results for the baseline using only multi-label classification and the proposed method that performed multi-label classification while using multi-class classification every N epochs. Each result is the average of five iterations. The value of N in eN represents frequency of switching to Softmax-CE. Here, e0 was trained using multi-task learning, where multi-label classification and multi-class classification were performed simultaneously every epoch. Conversely, e1 was trained using Softmax-CE every epoch and evaluated as a multi-label classification. The baseline method was performed on the original seven-class multi-label classification without the "No sound" class, and the proposed method was performed on the eight-class multi-label classification and multi-class classification including the "No sound" class. When e2, i.e., multi-label classification and multi-class classification, was used independently of the backbone for every other epoch, it demonstrated the best performance on several metrics. The proposed method improved mAP by 0.23 percentage points and 2.28 percentage points for Conv3 and RB, respectively. This result demonstrates that the proposed method improves the performance of rare events by using Softmax-CE and retains the performance of frequent events by using BCE. Meanwhile, the simultaneous use of Softmax-CE and BCE in e0 could have prevented the influence of Softmax-CE. With an increase in N, the performance of several metrics gradually decreases, and at e5, the performance is comparable to the baseline values. When Softmax-CE is used less frequently, it becomes less effective.

Table 4 shows the results when focusing on rare labels ("Screaming" and "Whispering"). Comparisons were made with asymmetric focal loss (AFL), which has an effect on the imbalanced data [13]. There is a significant difference between "Screaming" and "Whispering." "Screaming," which is similar to "Shout" and "Singing," is more likely to occur in noisy environments, whereas "Whispering" is a special type of sound event where other sound events are unlikely to occur concurrently. As with the overall performance, e2 performed highest on many measures, but the Fscore for "Screaming" dropped significantly. "Screaming" had a lower Fscore, which was based on the threshold, because underfitting caused by a rare label reduced the predicted probability of "Screaming." However, because e2 performed the highest, when comparing under the same conditions, the intrinsic prediction performance of e2 is superior and is effective for rare labels. Unlike the original experimental dataset used for evaluating AFL, the number of data classes (seven) was limited due to an imbalance of approximately 1 to 100 in a class. This may have contributed to the e2 performance being superior to that of AFL.

5. CONCLUSION

We proposed a method of polyphonic SED by periodically using either multi-label or multi-class classification. Based on the sparsity of sound events, multi-class classification was used to strongly train rare sound event labels, in which the rarest sound event was selected as the label representing each frame. The proposed method was evaluated on a human voice dataset extracted from the strongly labeled AudioSet data. Our approach was found to be most effective when the two loss functions were alternately applied. For the imbalanced data, regarding both the overall metrics and for rare labels, this method significantly outperformed the conventional methods.

6. REFERENCES

[1] G. McKenzie, "Urban mobility in the sharing economy: A spatiotemporal comparison of shared mobility services," *Computers, Environment and Urban Systems*, vol. 79, p. 101418, 2020.

[2] A. Mishra, S. Lee, D. Kim, and S. Kim, "In-cabin monitoring system for autonomous vehicles," *Sensors*, vol. 22, no. 12, 2022.

[3] A. Mesaros, A. Diment, B. Elizalde, T. Heittola, E. Vincent, B. Raj, and T. Virtanen, "Sound event detection in the dcase 2017 challenge," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 27, no. 6, pp. 992–1006, 2019.

[4] A. Politis, S. Adavanne, and T. Virtanen, "A dataset of reverberant spatial sound scenes with moving sources for sound event localization and detection," in *Proc. 5th Workshop on Detection and Classification of Acoustic Scenes and Events (DCASE)*, 2020, pp. 165–169.

[5] S. Hershey, D. P. W. Ellis, E. Fonseca, A. Jansen, C. Liu, R. Channing Moore, and M. Plakal, "The benefit of temporally-strong labels in audio event classification," in *Proc. 46th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 366–370.

[6] T. N. T. Nguyen, K. N. Watcharasupat, Z. J. Lee, N. K. Nguyen, D. L. Jones, and W. S. Gan, "What makes sound event localization and detection difficult? insights from error analysis," in *Proc. 6th Workshop on Detection and Classification of Acoustic Scenes and Events (DCASE)*, 2021, pp. 120–124.

[7] E. Çakır, G. Parascandolo, T. Heittola, H. Huttunen, and T. Virtanen, "Convolutional recurrent neural networks for polyphonic sound event detection," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 6, pp. 1291–1303, 2017.

[8] H. Phan, O. Y. Chén, P. Koch, L. Pham, I. McLoughlin, A. Mertins, and M. D. Vos, "Unifying isolated and overlapping audio event detection with multi-label multi-task convolutional recurrent neural networks," in *Proc. 44th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 51–55.

[9] E. Benetos, G. Lafay, M. Lagrange, and M. D. Plumbley, "Detection of overlapping acoustic events using a temporally-constrained probabilistic model," in *Proc. 41th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016, pp. 6450–6454.

[10] V. Bisot, S. Essid, and G. Richard, "Overlapping sound event detection with supervised nonnegative matrix factorization," in *Proc. 42th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017, pp. 31–35.

[11] Y. Cao, T. Iqbal, Q. Kong, F. An, W. Wang, and M. D. Plumbley, "An improved event-independent network for polyphonic sound event localization and detection," in *Proc. 46th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 885–889.

[12] H. Phan, T. N. T. Nguyen, P. Koch, and A. Mertins, "Polyphonic audio event detection: Multi-label or multi-class multi-task classification problem?" in *Proc. 47th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022, pp. 8877–8881.

[13] K. Imoto, S. Mishima, Y. Arai, and R. Kondo, "Impact of sound duration and inactive frames on sound event detection performance," in *Proc. 46th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 860–864.

[14] S. Park and M. Elhilali, "Time-balanced focal loss for audio event detection," in *Proc. 47th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022, pp. 311–315.

[15] R. Serizel, N. Turpault, H. Eghbal-Zadeh, and A. P. Shah, "Large-scale weakly labeled semi-supervised sound event detection in domestic environments," in *Proc. 3rd Workshop on Detection and Classification of Acoustic Scenes and Events (DCASE)*, 2018, pp. 19–23.

[16] D. de Benito-Gorrón, D. Ramos, and D. T. Toledano, "An analysis of sound event detection under acoustic degradation using multi-resolution systems," *Applied Sciences*, vol. 11, no. 23, 2021.

[17] L. Liu, H. Jiang, P. He, W. Chen, X. Liu, J. Gao, and J. Han, "On the variance of the adaptive learning rate and beyond," in *Proc. 8th International Conference for Learning Representations (ICLR)*, 2020.

ANALYZING THE EFFECT OF EQUAL-ANGLE SPATIAL DISCRETIZATION ON SOUND EVENT LOCALIZATION AND DETECTION

Saksham Singh Kushwaha[1], *Iran R. Roman*[2*], *Juan Pablo Bello*[2,3]

[1] Courant Institute of Mathematical Sciences, New York University, NY, USA
[2] Music and Audio Research Lab, New York University, NY, USA
[3] Center for Urban Science and Progress, New York University, NY, USA

ABSTRACT

Sound event localization and detection (SELD) models detect and localize sound events in space and time. Datasets for SELD often discretize spatial sound events along the polar coordinates of azimuth (integers from -180º to 180º) and elevation (integers from -90º to 90º). This discretization, known as equal-angle, results in more dense points at the poles (±90º elevation) than at the equator (0º elevation). We first analyzed the effect of equal-angle discretization on the 2022 DCASE SELD baseline model. Since the STARSS 2022 dataset that accompanies the model shows unbalanced sampling of spatial sound events along the elevation axis, we created a synthetic dataset. Our dataset has spatial sound events uniformly distributed along the elevation axis. We created two versions: one with targets spatially discretized using equal-angle, and another one with a uniform spatial discretization (both versions had the same audio). The model trained with equal-angle showed a greater angular localization error for targets around the equator compared to the poles, while the model trained with uniform spatial discretization showed a uniform localization error along the elevation axis. To train the model with the STARSS2022 dataset and reduce the effect of its equal-angle-discretized targets, we modified the model's loss function to penalize localization errors above an angular distance threshold around each target. Using this loss we fine-tuned a model trained with the original loss, and also trained the same model from scratch. Results showed improved localization metrics in both models compared to baseline, while retaining classification metrics. Our results show that equal-angle discretization yields models with non-uniform localization errors for targets along the elevation axis. Finally, our proposed loss function penalizes the SELD model's angular localization errors, regardless of which spatial discretization was used to annotate the dataset targets.

Index Terms— sound event localization and detection, spatial sampling, activity-coupled Cartesian direction of arrival, DCASE

1. INTRODUCTION

Sound event localization and detection (SELD) consists of localizing sound events in space and time while also assigning them to a class label [1]. SELD can be applied for environmental sound classification [2], simultaneous localization and mapping for navigation without visual input or with occluded targets [3, 4], tracking of sound sources of interest [5], audio surveillance [6], and acoustic imaging [7]. As a result, there has been an increased interest in SELD modeling, and research communities have organized challenges to centralize efforts and advancements [8, 9, 10].

*corresponding author email: roman@nyu.edu

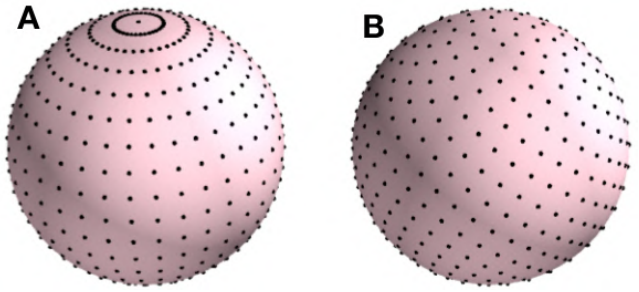


Figure 1: Two types of spatial discretization of points on a sphere. (A) equal-angle and (B) Fibonacci. (A) results in denser points at the poles than at the equator, while (B) does not.

The Detection and Classification of Acoustic Scenes and Events (DCASE) community introduced its annual SELD challenge in 2019 [10]. The DCASE SELD challenge provides participants with multichannel audio recordings of categorical sound events (i.e. speech, footsteps, running water, etc.) and their spatiotemporal trajectories on the azimuth and elevation axes. Participants must develop methods to detect, localize, and classify each event. The DCASE SELD challenge also provides a baseline model, which reflects key incremental advancements from the community. For example, a significant advancement has been in the training loss function, which went from separately measuring sound event detection (SED) and direction-of-arrival (DOA) [1] to jointly carrying out these using a mean-squared-error (MSE) regressor that accounts for overlapping sound events categories [11].

SELD datasets often have spatial targets that are discretized in a sphere along the elevation ($\theta = [-90, 90] \in \mathbb{Z}$) and azimuth ($\phi = [-180, 180) \in \mathbb{Z}$) axes in units of degrees [1, 12, 13, 14]. This sampling of points in space, known as equal-angle, is easy to interpret because it yields uniform-looking grids on a 2D projection (ϕ vs θ). However, equal-angle points on the sphere shows a larger density of points at the poles (±90° elevation) than at the equator (0° elevation) [15]. Furthermore, equal-angle sampling results in larger quantization errors around the equator.

As far as we know, current SELD research has not studied how a non-uniform density of points along the elevation axis impacts model performance. We hypothesize that training SELD models with equal-angle discretized data results in non-uniform localization performance along the elevation axis. This paper empirically analyzes the impact of equal-angle spatial discretization on SELD model performance and proposes a practical way to mitigate it.

2. EQUAL-ANGLE DISCRETIZATION IS IRREGULAR

A sphere can be discretized into N points using a sampling function $s(\theta, \phi)$, resulting in the set of vectors $\{x_1, x_2, ...x_N\} \subset \mathbb{S}^2$. This set represent a lattice of directions around the sphere [16]. For the specific case of equal-angle spatial discretization, $N = N_E \times N_A$, where N_E is the number of points uniformly sampled along the elevation axis $\theta \in [-\frac{\pi}{2}, \frac{\pi}{2}]$ and N_A is the number points along the azimuth axis $\phi \in [0, 2\pi)$ (see supplement S1[1] for a detailed mathematical description of the equal-angle sampling function and its non-uniform density along the elevation axis).

Figure 1A shows how equal-angle spatial discretization, although regular along the axes of azimuth and elevation, results in a lattice with non-uniform distances between points, particularly noticeable along the elevation axis (i.e. poles versus equator).

It is worth noting that humans listen most events close to the equator (i.e. other speakers or ecologically-relevant sound sources on the azimuth). This introduces another sampling bias in realistic SELD datasets. Thus, equal-angle discretization can yield less-than-ideal resolution where most relevant sound sources exist.

3. SELD WITH EQUAL-ANGLE SPATIAL TARGETS

We want to empirically test if equal-angle discretization affects SELD model performance. We hypothesize that the model's localization error will be a function of target elevation. More specifically, we predict that the irregularities of equal-angle discretization will result in larger errors around the equator than at the poles.

3.1. Synthetic dataset with equal-angle spatial discretization

We wanted to do this analysis with the DCASE STARSS2022 dataset [17] (real-world events belonging to thirteen categories, spatially discretized using equal-angle). However, its distribution of events on the elevation axis is not uniform (see supplement S2). Therefore, it will be hard to determine the effect of equal-angle discretization on model performance (since the data shows non-uniform distribution of targets). Moreover, our analysis is not focused on classification, so a single sound event category would be enough. We decided to create a synthetic dataset that controls for uniform target localization along the elevation axis using a single sound category. In contrast to our dataset, DCASE STARSS2022 is more complex, so we expect SELD models to easily learn our synthetic training split and generalize to the test split.

Our synthetic dataset has no moving or overlapping events, and repeats a single alarm sound (5 seconds duration) from FSD50k [18]. We used impulse responses (IRs) from two rooms (No. 3 and No. 4) in the TAU-SRIR database [19] that we convolve with the alarm sound. The rooms were selected because they contain IRs from sources localized at elevations spanning the integers

$$\theta \in [-33..32] \mid \theta \neq -25, -24, -3, -2, 3, 20, 21 \qquad (1)$$

in units of degrees (the missing integers are elevations not present in the two rooms). To generate the data we used the DCASE2022-data-generator that accompanies the TAU-SRIR database [19].

We synthesized two data folds: training and testing with 1600 and 900 tracks, respectively. Each track was a four-channel signal ($fs = 24kHz$) with a duration of 1 minute (sequences of alarm sounds and silence). While both training and test sets had elevation values spanning the range described in Eq. 1, no IRs overlapped across sets. This ensured that the absolute location of simulated sound sources was different between training and test sets. Figure 2 shows the distribution of sound event locations in our dataset.

[1]the supplement and code are available at `https://github.com/sakshamsingh1/dcase_seld_spatial_sampling_analysis`

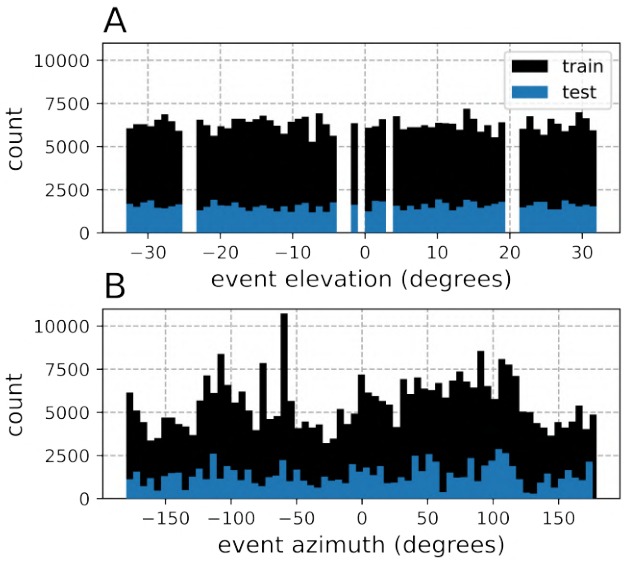


Figure 2: Density of sound events locations in our synthetic dataset along the elevation (A) and azimuth axes (B). Data synthesis controlled for uniform density of sound events along the elevation axis.

3.2. SELD model localization on equal-angle targets

We used our dataset to train the 2022 DCASE SELD baseline model [1], which is a convolutional recurrent neural network that maps multichannel audio features (generalized cross-correlation with phase transform) into sound event locations and classes (see section 6.1 for a description of the model's output format). The trained model detected all test set sound events and showed an average angular localization error of 1.81°. Figure 3A shows a scatter plot with the localization error for each test set prediction as a function of target elevation. To gain intuition about how the model's localization varies as a function of elevation, we fit a line and a second-order polynomial (i.e. parabola) to this plot. The coefficient that multiplies the polynomial's second-order term determines the curvature of the parabola. If its value is close to zero, this indicates that the parabola resembles a line. In contrast, a more negative (positive) coefficient indicates that the parabola is more curved, and we can interpret it as the model's error decreasing (increasing) as a function of elevation away from the equator.

The line was $\hat{y}_l = 0.89 + (3.2 \times 10^{-3})x$, while the polynomial was $\hat{y}_p = 1.03 + (2.9 \times 10^{-3})x - (0.4 \times 10^{-3})x^2$, also shown in Figure 3A. The polynomial's second order coefficient reveals an upside-down parabolic relationship between the model's angular localization error and target elevation. The Pearson's correlation coefficient for the linear regression was $r = 0.01$, and for the polynomial regression was $r = 0.09$. This indicates that the parabola better explains the model's error than the line. These results show that training a SELD model with equal-angle data results in larger localization errors around the equator.

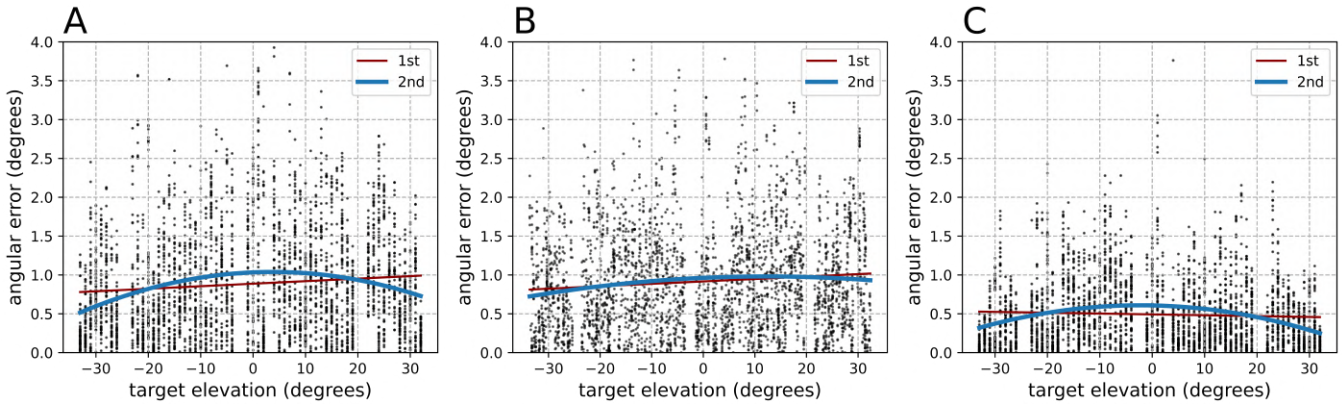


Figure 3: Scatter plots of SELD model angular localization error on our synthetic test data as a function of target elevation. Each dot is a sound event. (A) Trained with equal-angle points. (B) Trained with Fibonacci points. (C) Trained with equal-angle points and fine-tuned using our proposed loss function (see section 6). The red and blue lines are linear and second-order polynomial regressions, respectively.

4. UNIFORM DISCRETIZATION OF THE SPHERE

Alternatives to equal-angle discretization that uniformly sample points on the sphere exist [20, 21]. Perfectly-uniform discretization is limited by using the five Platonic solids, whose vertices can be used as points on the sphere [15]. The Fibonacci lattice [16] is another possible method that results in neighboring points separated by a roughly equal angular distance and is obtained by sampling points along a spiral that links the two poles (see [16] for the Fibonacci lattice formula). Figure 1B shows Fibonacci discretization.

5. SELD WITH UNIFORM SPATIAL TARGETS

We want to analyze SELD model localization when trained with uniformly discretized spatial targets. We hypothesize that this will result in uniform localization error on the elevation axis.

5.1. Synthetic dataset with uniform spatial sampling

Our synthetic dataset generated in section 3 spatially discretized sound events using equal-angle. To generate a uniformly discretized version of our dataset, we took our dataset's equal-angle annotations and transformed them into Fibonacci discretization by converting each equal-angle target into the nearest Fibonacci point. Our Fibonacci discretization had N=32768 points, which is the power of two that yields an angular distance between neighboring points around 1° [16] (similar to the distance between integers in Eq. 1). The audio tracks were exactly the same across equal-angle and Fibonacci versions of the dataset.

5.2. SELD model localization on Fibonacci targets

The 2022 DCASE SELD model trained with Fibonacci targets also detected all sound events, showing an average angular localization error of 1.86° on the test set (Figure 3B shows this model's scatter plot). We also fit a line to this plot, which was $\hat{y}_l = 0.92 + (3.1 \times 10^{-3})x$, and a second-order polynomial, which was $\hat{y}_p = 0.96 + (3.0 \times 10^{-3})x - (0.1 \times 10^{-3})x^2$. Compared to the polynomial for the model trained with equal-angle data, this polynomial's second order term reveals a less pronounced parabola, which is also visible in Figure 3B. The Pearson's correlation coefficient for the linear regression was $r = 0.09$, and for the polynomial regression was $r = 0.11$. This indicates that, compared to equal-angle, the parabola and the line more similarly explain the model's error as a function of target elevation after training with Fibonacci targets. In other words, the model trained with Fibonacci data shows localization errors that are uniform as a function of target elevation.

Our results clearly illustrate how SELD model performance is affected by equal-angle and Fibonacci discretization. However, we recognize that the Pearson and parabolic coefficients we observed show clear trends but are relatively weak indicators. Future work could support our observations using more robust statistical testing.

6. PROPOSED SOLUTION

Our empirical analysis with synthetic data revealed that equal-angle discretization can result in a SELD model with larger localization errors at the equator than at the poles. Substituting the equal-angle discretization with a uniform one (like a Fibonacci lattice) would be a simple solution. In fact, resampling the DCASE STARSS2022 dataset using a Fibonacci lattice and training the model from scratch did result in improved metrics on the test set compared to baseline (see Table 6.1). However, since equal-angle discretization is prevalent in SELD datasets, engineered SELD learning methods that reduce its impact without the need to spatially resample the data are needed. Here we propose a training loss function that, in addition to computing the mean-squared error (MSE) between targets and model predictions, penalizes the model's angular localization error uniformly for all points on the sphere.

6.1. The threshold angular error ADPIT (TAEADPIT) loss

The 2022 DCASE SELD model is trained with the auxiliary duplicating permutation invariant training (ADPIT) loss [11], which uses the multi-class activity-coupled cartesian direction of arrival (multi-ACCDOA) target format $\boldsymbol{P} \in \mathbb{R}^{3\times N\times C\times T}$, where 3 is the dimensionality of 3D cartesian coordinates, N is the maximum number of simultaneous sound events the model is trained to detect, C is the number of classes and T is the number of time frames. A vector $\boldsymbol{P}_{nct} \in \mathbb{R}^3$ has a magnitude of 1, i.e. $||\boldsymbol{P}_{nct}||_2 = 1$ and represents the location of a sound event for a specific track n, a specific

class c and a specific time-frame t. Such a vector may also represent the absence of a sound event if it has a magnitude of 0. A related term, $a \in \mathbb{R}^{N\times C\times T}$, indicates the activities over tracks, classes, and time, and $a_{nct} \in \{0, 1\}$. It is worth noting that the target contains duplicated sound events along the N dimension when the number of simultaneous sound events for each class is less than the maximum N. A complete description of the ADPIT loss can be found in its original publication [11].

The multi-ACCDOA format is permuted for each time-frame and class, and all permutation are compared against the model's output $\hat{\boldsymbol{P}} \in \mathbb{R}^{3\times N\times C\times T}$ using MSE, yielding the ADPIT loss

$$\mathcal{L}^{ADPIT} = \frac{1}{CT}\sum_{c}^{C}\sum_{t}^{T}\min_{\alpha\in\text{Perm}(ct)} l_{\alpha,ct}^{ACCDOA}, \tag{2}$$

$$l_{\alpha,ct}^{ACCDOA} = \frac{1}{N}\sum_{n}^{N}\text{MSE}(\boldsymbol{P}_{\alpha,nct}^{*}, \hat{\boldsymbol{P}}_{nct}), \tag{3}$$

where $\boldsymbol{P}_{\alpha,nct}^{*}$ indicates a permutation of the multi-ACCDOA format, and only the permutation that resulted in the minimum $l_{\alpha,ct}^{ACCDOA}$ term is used to average over classes and time-frames.

Due to the nature of the multi-ACCDOA format, the ADPIT loss function operates over Cartesian coordinates. We propose adding a term that penalizes the model's angular localization error on the sphere where the data is spatially discretized:

$$\begin{aligned} l_{\alpha,ct}^{ALE} &= \max(\boldsymbol{a}_{\alpha,nct}\text{ALE}_{\alpha,nct}, H) \\ \text{ALE}_{\alpha,nct} &= \angle(p(\boldsymbol{P}_{\alpha,nct}), p(\hat{\boldsymbol{P}})_{nct}), \end{aligned} \tag{4}$$

where $p(x)$ is a function that converts from cartesian to polar coordinates denotes, $\angle(a, b)$ is the angular distance between inputs a and b, and H is a threshold. ALE stands for angular localization error. Note that the $\boldsymbol{a}_{\alpha,nct}$ term masks ALE so that only the model's angular localization error related to active targets counts toward the loss. Adding the ADPIT loss gives:

$$\mathcal{L}^{TAEADPIT} = \frac{1}{CT}\sum_{c}^{C}\sum_{t}^{T}\min_{\alpha\in\text{Perm}(ct)} l_{\alpha,ct}^{ACCDOATAE}, \tag{5}$$

$$l_{\alpha,ct}^{ACCDOATAE} = \frac{1}{N}\sum_{n}^{N}\text{MSE}(\boldsymbol{P}_{\alpha,nct}^{*}, \hat{\boldsymbol{P}}_{nct}) + \beta(l_{\alpha,ct}^{ALE} - H), \tag{6}$$

where β is a scale factor on the new term. We call this new loss "thresholded angular error ADPIT" (TAEADPIT) loss. The new term ALE is a regularizer that uniformly penalizes angular localization errors, independent of how targets are spatially discretized.

7. EMPIRICAL EVALUATION OF THE TAEADPIT LOSS

We conducted experiments to assess the TAEADPIT loss. First, we used it to fine-tune the model trained with the equal-angle version of our synthetic dataset. Figure 3C shows the model's angular localization error as a function of elevation. We also fit a line to this plot, which was $\hat{y}_l = 0.49 - (1.1\times10^{-3})x$, and a second-order polynomial, which was $\hat{y}_q = 0.61 - (1.4\times10^{-3})x - (0.31\times10^{-3})x^2$. The parabola's second order coefficient has a value of -0.31×10^{-3}, which is closer to zero compared to the one found before fine-tuning (-0.42×10^{-3}). This indicates that fine-tuning with the TAEADPIT loss flattened the parabola. In other words, the model fine-tuned with the TAEADPIT loss shows more uniform localization errors as a function of target elevation than it did before being fine-tuned.

Loss	ER_{20^o}	F_{20^o}	LE_{CD}	LR_{CD}	SELD
ADPIT-base	0.69	0.24	30.43	0.43	0.55
TAEDPIT-tune	0.71	0.23	28.86	0.47	0.54
TAEDPIT	0.71	0.20	26.42	0.41	0.56
ADPIT-Fib	0.68	0.22	26.11	0.46	0.54

Table 1: Comparison of SELD model performance when training with ADPIT loss versus training with the proposed TAEADPIT loss. ADPIT-base: the baseline 2022 DCASE SELD model. TAEADPIT-tune: fine-tuning the baseline model with TAEADPIT. TAEADPIT: baseline model trained from scratch with TAEADPIT. ADPIT-Fib: baseline model trained with the data spatially resampled to the Fibonacci lattice and the ADPIT loss. The metrics are the DCASE SELD challenge metrics with class-depending macro-averaging are used (see [10]).

We also wanted to assess the TAEADPIT loss using real-world data. First, we ensured that we could replicate the baseline DCASE SELD 2022 model metrics using the four-channel microphone version of the DCASE STARSS2022 dataset [14] and supplemental synthetic data [22] (see Table 1). Next, we ran a couple of experiments to assess whether the TAEDPIT loss could benefit this model's performance. In all experiments, $\beta = 1 \times 10^{-3}$ in Eq. 5 (empirically-found). First, we used the TAEADPIT loss to fine-tune it. Then, we trained it from scratch using the TAEADPIT loss. Table 6.1 shows the results on the DCASE SELD metrics, indicating that using the TAEADPIT loss to fine-tune or train the SELD model from scratch can improve its localization error while retaining or only marginally impacting the classification metrics.

Our results show that the TAEADPIT loss can be used to train a SELD model using equal-angle data and improve localization metrics, and that it does so by reducing the larger localization error around the equator produced by equal-angle discretization.

8. CONCLUSION

In this paper, we studied the irregularities of equal-angle spatial discretization, which results in a larger density of points at the poles than at the equator. No previous studies have shown whether a SELD model's performance is affected by training with equal-angle discretized targets. We have empirically shown that equal-angle data affects SELD model localization on the elevation axis, causing larger localization errors around the equator than at the poles.

We also studied whether discretizing targets using a uniform Fibonacci lattice resulted in the same effect. We found that training a SELD model with Fibonacci data results in more uniform localization errors along the elevation axis compared to equal-angle. We also proposed a loss function to mitigate the effect of equal-angle by adding a thresholded angular localization error term to the ADPIT loss. Empirical results using our proposed loss when training a SELD model with equal-angle showed improved localization metrics compared to when using the ADPIT loss.

Next, we would like to assess whether a thresholded angular localization error in the training loss benefits other SELD models, and whether the benefit depends on audio format (i.e. FOA, HOA, stereo) and/or localization target format (i.e. Cartesian, 3D polar).

9. ACKNOWLEDGMENTS

This work is supported by the National Science Foundation grant no. IIS-1955357. The authors thank the funding source and their grant collaborators. We would also like to thank Charalampos Avraam for his comments to improve Supplement Section S1.

10. REFERENCES

[1] S. Adavanne, A. Politis, J. Nikunen, and T. Virtanen, "Sound event localization and detection of overlapping sources using convolutional recurrent neural networks," *IEEE Journal of Selected Topics in Signal Processing*, vol. 13, no. 1, pp. 34–48, 2018.

[2] D. Barchiesi, D. Giannoulis, D. Stowell, and M. D. Plumbley, "Acoustic scene classification: Classifying environments from the sounds they produce," *IEEE Signal Processing Magazine*, vol. 32, no. 3, pp. 16–34, 2015.

[3] C. Evers and P. A. Naylor, "Acoustic slam," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 26, no. 9, pp. 1484–1498, 2018.

[4] C. Chen, U. Jain, C. Schissler, S. V. A. Gari, Z. Al-Halah, V. K. Ithapu, P. Robinson, and K. Grauman, "Soundspaces: Audio-visual navigation in 3d environments," in *European Conference on Computer Vision*. Springer, 2020, pp. 17–36.

[5] W. Mack, U. Bharadwaj, S. Chakrabarty, and E. A. Habets, "Signal-aware broadband doa estimation using attention mechanisms," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 4930–4934.

[6] G. Valenzise, L. Gerosa, M. Tagliasacchi, F. Antonacci, and A. Sarti, "Scream and gunshot detection and localization for audio-surveillance systems," in *2007 IEEE Conference on Advanced Video and Signal Based Surveillance*. IEEE, 2007, pp. 21–26.

[7] M. Simeoni, S. Kashani, P. Hurley, and M. Vetterli, "Deepwave: a recurrent neural-network for real-time acoustic imaging," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[8] R. Stiefelhagen, K. Bernardin, R. Bowers, J. Garofolo, D. Mostefa, and P. Soundararajan, "The clear 2006 evaluation," in *International evaluation workshop on classification of events, activities and relationships*. Springer, 2006, pp. 1–44.

[9] C. Evers, H. W. Löllmann, H. Mellmann, A. Schmidt, H. Barfuss, P. A. Naylor, and W. Kellermann, "The locata challenge: Acoustic source localization and tracking," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 1620–1643, 2020.

[10] A. Politis, A. Mesaros, S. Adavanne, T. Heittola, and T. Virtanen, "Overview and evaluation of sound event localization and detection in dcase 2019," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 684–698, 2020.

[11] K. Shimada, Y. Koyama, S. Takahashi, N. Takahashi, E. Tsunoo, and Y. Mitsufuji, "Multi-accdoa: Localizing and detecting overlapping sounds from the same class with auxiliary duplicating permutation invariant training," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Singapore, Singapore, May 2022.

[12] A. Politis, S. Adavanne, and T. Virtanen, "A dataset of reverberant spatial sound scenes with moving sources for sound event localization and detection," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2020 Workshop (DCASE2020)*, Tokyo, Japan, November 2020, pp. 165–169. [Online]. Available: https://dcase.community/workshop2020/proceedings

[13] A. Politis, S. Adavanne, D. Krause, A. Deleforge, P. Srivastava, and T. Virtanen, "A dataset of dynamic reverberant sound scenes with directional interferers for sound event localization and detection," in *Proceedings of the 6th Detection and Classification of Acoustic Scenes and Events 2021 Workshop (DCASE2021)*, Barcelona, Spain, November 2021, pp. 125–129. [Online]. Available: https://dcase.community/workshop2021/proceedings

[14] A. Politis, K. Shimada, P. Sudarsanam, S. Adavanne, D. Krause, Y. Koyama, N. Takahashi, S. Takahashi, Y. Mitsufuji, and T. Virtanen, "Starss22: A dataset of spatial recordings of real scenes with spatiotemporal annotations of sound events," *arXiv preprint arXiv:2206.01948*, 2022.

[15] B. Rafaely, *Fundamentals of spherical array processing*. Springer, 2015, vol. 8.

[16] B. Keinert, M. Innmann, M. Sänger, and M. Stamminger, "Spherical fibonacci mapping," *ACM Transactions on Graphics (TOG)*, vol. 34, no. 6, pp. 1–7, 2015.

[17] A. Politis, Y. Mitsufuji, P. Sudarsanam, K. Shimada, S. Adavanne, Y. Koyama, D. Krause, N. Takahashi, S. Takahashi, and T. Virtanen, "STARSS22: Sony-TAu Realistic Spatial Soundscapes 2022 dataset," May 2022. [Online]. Available: https://doi.org/10.5281/zenodo.6600531

[18] E. Fonseca, X. Favory, J. Pons, F. Font, and X. Serra, "Fsd50k: an open dataset of human-labeled sound events," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 30, pp. 829–852, 2021.

[19] A. Politis, S. Adavanne, and T. Virtanen, "TAU Spatial Room Impulse Response Database (TAU- SRIR DB)," Apr. 2022. [Online]. Available: https://doi.org/10.5281/zenodo.6408611

[20] L. Perotin, R. Serizel, E. Vincent, and A. Guérin, "Crnn-based multiple doa estimation using acoustic intensity features for ambisonics recordings," *IEEE Journal of Selected Topics in Signal Processing*, vol. 13, no. 1, pp. 22–33, 2019.

[21] P.-A. Grumiaux, S. Kitić, L. Girin, and A. Guérin, "Improved feature extraction for crnn-based multiple sound source localization," in *2021 29th European Signal Processing Conference (EUSIPCO)*. IEEE, 2021, pp. 231–235.

[22] A. Politis, "[DCASE2022 Task 3] Synthetic SELD mixtures for baseline training," Apr. 2022. [Online]. Available: https://doi.org/10.5281/zenodo.6406873

IS MY AUTOMATIC AUDIO CAPTIONING SYSTEM SO BAD? SPIDEr-max: A METRIC TO CONSIDER SEVERAL CAPTION CANDIDATES

Etienne Labbé, Thomas Pellegrini, Julien Pinquier

IRIT, Université Paul Sabatier, CNRS, Toulouse, France
{etienne.labbe, thomas.pellegrini, julien.pinquier}@irit.fr

ABSTRACT

Automatic Audio Captioning (AAC) is the task that aims to describe an audio signal using natural language. AAC systems take as input an audio signal and output a free-form text sentence, called a caption. Evaluating such systems is not trivial, since there are many ways to express the same idea. For this reason, several complementary metrics, such as BLEU, CIDEr, SPICE and SPIDEr, are used to compare a single automatic caption to one or several captions of reference, produced by a human annotator. Nevertheless, an automatic system can produce several caption candidates, either using some randomness in the sentence generation process, or by considering the various competing hypothesized captions during decoding with beam-search, for instance. If we consider an end-user of an AAC system, presenting several captions instead of a single one seems relevant to provide some diversity, similarly to information retrieval systems. In this work, we explore the possibility to consider several predicted captions in the evaluation process instead of one. For this purpose, we propose SPIDEr-max, a metric that takes the maximum SPIDEr value among the scores of several caption candidates. To advocate for our metric, we report experiments on Clotho v2.1 and AudioCaps, with a transformed-based system. On AudioCaps for example, this system reached a SPIDEr-max value (with 5 candidates) close to the SPIDEr human score of reference.

Index Terms— audio captioning, evaluation metric, beam search, multiple candidates

1. INTRODUCTION

Automated Audio Captioning (AAC) is the task, in which a system takes an audio signal as input and provides a short description of its content using natural language. AAC could be useful for hearing-impaired people, in machine-to-machine interaction, surveillance and information retrieval in general. In the last few years, the research community has developed a keen interest in AAC, in particular thanks to the Detection and Classification of Acoustic Scenes and Events (DCASE) Challenges and Workshops [1], which have provided datasets and benchmarks for this task.

Most AAC systems use deep neural networks with a sequence-to-sequence encoder-decoder architecture, to build a semantic audio representation and generate a valid sentence as output [1]. They rely on models pretrained on large-scale datasets, to solve the data scarcity issue in AAC [2, 3, 4].

In this work, we are interested in the evaluation of AAC systems. AAC evaluation borrows metrics from machine translation and image captioning, and consists of comparing a candidate caption to one or several manually produced captions of reference. Since evaluating text generated automatically is a difficult problem, several metrics are used in combination. We investigate in particular the SPIDEr metric [5], a short name used to designate the average of two metrics called Consensus-based Image Description Evaluation (CIDEr) [6] and Semantic Propositional Image Caption Evaluation (SPICE) [7]. SPIDEr is used, for instance, in the DCASE yearly challenges to rank the participant AAC systems [2].

In this paper, we report experiments using the AAC system we developed to participate in the DCASE 2022 AAC task. Like most AAC systems, we use a beam search decoder that allows to generate several candidate captions. The most likely one is used to compute the SPIDEr score of our system. A strong limitation of SPIDEr, in our opinion, is that only one caption candidate is considered for evaluation. As we shall illustrate in this paper, two correct captions that differ by a single word may have very different SPIDEr scores, if one of the words happens to be in the caption(s) of reference. To overcome this issue, we propose a metric that we call SPIDEr-max, which takes into account multiple candidates for a single audio recording.

2. METRICS

In the literature, most AAC systems are evaluated using the CIDEr, SPICE or SPIDEr metrics. These metrics come from the field of image captioning and evaluate a single candidate caption against a reference set.

2.1. CIDEr

CIDEr [6] is a metric based on the TF-IDF (term frequency-inverse document frequency) scores of each n-gram of the candidate and reference sentences. TF-IDF is used to give a higher weight to infrequent n-grams and lower weight to frequent n-grams.

The CIDEr metric calculation starts by stemming all the words and compute all the n-grams of size 1 to N across all candidates and references. The frequency of each n-gram in references are used to compute TF-IDF of all captions. This means that the score of each candidate does not only depend on its corresponding references, but also on all the other references of the corpus being evaluated. Then, the TF-IDF scores are vectorized and used to compute cosine similarity between the candidate and each reference. The similarities are rescaled by a factor of 10 and averaged across the references to get the final score of the candidate. All the scores are averaged again to get the global score on a dataset.

The CIDEr-D metric is a more robust version of CIDEr supposed to be closer to human judgement. It removes the stemming operation to take into account the tense and plural of words, adds

[1] http://dcase.community/

[2] http://dcase.community/challenge2022/

a penalty factor, and limits the maximum occurrence of candidate n-grams to penalize longer repetitive sentences. The penalty is multiplied by a similarity measure based on the length of the candidate c and the reference r:

$$\text{Penalty}(c, r) = \exp\Big(-\frac{(|c| - |r|)^2}{2\sigma^2}\Big) \quad (1)$$

Some AAC papers do not specify which version of CIDEr they use, but in this paper we report CIDEr-D scores as used in the DCASE challenge. We use the default settings of CIDEr-D with the maximum n-gram size N set to 4, and the hyperparameter σ used for the penalty set to 6.

2.2. SPICE

SPICE [7] attempts to extract the semantic content of a sentence. Sentences are used as input to a Probabilistic Context-Free Grammar dependency parser[8], with several additional rules to build a dependency tree where each node is a word and each edge is a syntactic dependency. Custom rules are used to compute another graph, a "semantic scene graph", comprised of three types of nodes: objects, attributes and relations. Attributes are linked to a single object, and relations connect objects between them. The reference graphs are merged into one to be compared with a candidate graph. Then, the scene graphs are converted into lists of word tuples. An object is a tuple with the object name, an attribute is a tuple of two words with the object and attribute names, and a relation is a tuple of three words containing the two objects connected and the relation names. Finally, the list is binarized for the candidate and the references, and used to compute an F-Score.

The M-SPICE metric [9] is a variant of SPICE, which takes multiple candidates for a single audio. This metric was introduced to evaluate the diversity of the words used in multiple candidates generated by stochastic decoding methods. The only difference is that the semantic graph of each candidate is merged into one, exactly as for the reference list. The other steps remain the same.

2.3. SPIDEr

SPIDEr [5] is a metric originally used as a cost function to optimize a model on SPICE and CIDEr-D at the same time. SPIDEr is the average of CIDEr-D and SPICE, and is supposed to have the benefits of both previous metrics. Since CIDEr-D gives a score between 0 and 10 and SPICE between 0 and 1, the SPIDEr score is between 0 and 5.5, which is quite uncommon for a metric. SPIDEr is usually the metric used in AAC papers to compare models, even if other machine-translation metrics like BLEU [10], ROUGE-L [11], and METEOR [12] scores are also reported.

3. SYSTEM DESCRIPTION

3.1. Datasets

The Clotho v2.1 dataset [13] is an audio captioning dataset containing 6974 audio files of approximately 43.6 hours from Freesound between 15 and 30 seconds. Each audio is described by 5 captions annotated by humans. The dataset is divided in 3 different splits: development, validation and evaluation, which corresponds to development-training, development-validation and development-testing, the conventional names used in the DCASE Challenge. In this paper, we use these names. The training subset contains 217362 words with a caption length between 8 and 20 words.

AudioCaps [14] is another audio captioning dataset containing 49838 training files of approximately 136.6 hours from AudioSet [15], a large audio tagging dataset with audio extracted from YouTube. AudioCaps contains only 1 caption per audio in the training subset and 5 captions for the validation and testing subsets. Since YouTube removes videos uploaded by users for various reasons, our version of AudioCaps contains only 46230 over 49838 files in training subset, 464 over 495 in validation subset and 912 over 975 files in testing subset. Our training subset contains 402482 words with a caption length between 1 and 52 words.

To extract audio features, we resample audio signals to 32 kHz and compute log-Mel spectrograms with a window size of 32 ms, a hop size of 10 ms and 64 Mel bands. All captions are put in lowercase and punctuation characters are removed. We used the spaCy tokenizer [16] to split sentences into words, resulting in a vocabulary of 4370 tokens for Clotho and 4724 words for AudioCaps.

3.2. Model architecture

We adopt a standard encoder-decoder structure used in most AAC systems, with a pre-trained encoder to extract audio features and a transformer decoder to generate our captions. The encoder is the CNN10 model, a convolutional network from the Pretrained Audio Neural Networks study (PANN) [2]. We used the weights available on Zenodo[3] to initialize the model at the beginning of the training. An affine layer was added to project 512-dimensional to 256-dimensional embeddings. We kept the time axis of the audio embedding used as input for the decoder.

The decoder is a standard transformer decoder [17]. It takes the audio embeddings as inputs and all the previous words predicted. The word embeddings are randomly initialized and learned during training. We use teacher forcing with cross-entropy to train the model. During the testing phase, captions are generated using beam search, and we select the best candidate using the probability of the sentence P given by the model. The combination of our encoder and decoder is simply named "CNN10-Transformer".

3.3. Experimental setup

We trained models for 50 epochs, on both datasets separately. To optimize our networks, we used Adam [18], with a learning rate set to 5.10^{-4} at the first epoch, a 10^{-6} weight decay, a 0.9 β_1 and 0.999 β_2, and ϵ set to 10^{-8}. We used a cosine learning rate scheduler with the following rule:

$$\text{lr}_k = \frac{1}{2}\left(1 + \cos\left(\frac{k\pi}{K}\right)\right)\text{lr}_0 \quad (2)$$

with k being the current epoch index, and K the total number of epochs.

The transformer decoder uses an embedding dimension d_{model} of 256, four attention heads h, six stacked standard decoder layers, and a global dropout P_{drop} set to 0.2. The last affine layer projects the 256-dimensional embeddings to an output of the vocabulary size of the dataset. We used label smoothing to reduce overfitting, set to 0.1 for AudioCaps and 0.2 for Clotho. In order to avoid gradient explosion, we clip gradients by a maximal L2-norm value set to 10 and 1 for AudioCaps and Clotho, respectively. During testing, beam size is set to 8 for Clotho and 2 for AudioCaps. The final encoder-decoder model results in 16M trainable

[3] https://zenodo.org/record/3987831

parameters. We also used SpecAugment [19] as audio data augmentation with two bands dropped on the time axis with a maximal size of 64 bins and one band dropped on the frequency axis with a maximum size of two bins. Our implementation uses PyTorch [20], PyTorch-Lightning [21] and our aac-datasets [4] package to download and manage audio captioning datasets.

4. SPIDEr RESULTS

Results on Clotho and AudioCaps of our model CNN10-Transformer are shown in Table 1. Standard deviations of our model are very small (0.001 and 0.004 for Clotho and AudioCaps, respectively). Cross-reference scores are computed by using one of the reference as a candidate and the four others as references five times.

Table 1: SPIDEr scores on Clotho v2.1 and AudioCaps with state-of-the-art results and cross-reference scores.

System	Clotho	AudioCaps
Best	0.320 [22]	0.465 [4]
Human	N/A	0.565 [14]
Cross-Referencing	0.573	0.564
CNN10-Transformer (ours)	0.247	0.401

Our model performs much better on AudioCaps than Clotho, with a SPIDEr score of 0.401 and 0.247, respectively. It is also closer to the cross-reference and human scores in AudioCaps. This is probably due to the fact that the CNN10 encoder has been pre-trained on AudioSet, which is a superset of AudioCaps. In addition, the captions in AudioCaps are simpler than those in Clotho, with shorter sentences and a relatively smaller vocabulary. The current best score on AudioCaps is also much closer to the cross-reference top score (0.100 difference) than the one on Clotho (0.253 difference).

5. SPIDEr LIMITATIONS

5.1. The SPIDEr score varies greatly between beam search candidates

Tables 2 and 3 show examples of candidates and captions of reference, one from Clotho, one from AudioCaps. The probability P given by the model is also indicated. It used to select the best candidate among the beam search hypotheses. We also reported the SPIDEr score associated to each candidate.

In the Clotho example, the most likely caption candidate is also the one with the highest SPIDEr score, based on the fact that the rather rare word "tin" was found by the automatic system. Thus, in this example, the differences observed between the various hypotheses seem justified. On the contrary, in the second example, from AudioCaps, the most likely automatic caption is different from the one with the highest SPIDEr score.

The agreement accuracy between the best candidate according either to the likelihood and to the SPIDEr score is only of 26.5% on Clotho, and 22.6% on AudioCaps. The correlation coefficient on all the likelihoods and the SPIDEr scores is 0.224 on Clotho and 0.259 on AudioCaps. This shows that the maximum candidate likelihood P does not select the best caption according to the SPIDEr score.

[4] https://pypi.org/project/aac-datasets

Table 2: Captions for the Clotho development-testing file named "rain.wav".

Candidates	P	SPIDEr
heavy rain is falling on a roof	0.361	0.562
heavy rain is falling on a **tin** roof	**0.408**	**0.930**
a heavy rain is falling on a roof	0.369	0.594
a heavy rain is falling on the ground	0.351	0.335
a heavy rain is falling on the roof	0.340	0.594
References		
heavy rain falls loudly onto a structure with a thin roof		
heavy rainfall falling onto a thin structure with a thin roof		
it is raining hard and the rain hits a tin roof		
rain that is pouring down very hard outside		
the hard rain is noisy as it hits a tin roof		

Table 3: Captions for an AudioCaps testing file (id: 'jid4t-FzUn0').

Candidates	P	SPIDEr
a woman speaks and a sheep bleats	0.475	0.190
a woman speaks and a **goat** bleats	0.464	**1.259**
a man speaks and a sheep bleats	0.464	0.344
an adult male speaks and a sheep bleats	0.450	0.231
an adult male is speaking and a sheep bleats	**0.491**	0.189
References		
a man speaking and laughing followed by a goat bleat		
a man is speaking in high tone while a goat is bleating one time		
a man speaks followed by a goat bleat		
a person speaks and a goat bleats		
a man is talking and snickering followed by a goat bleating		

5.2. Can we choose a better candidate automatically?

Selecting automatically the best candidate among the beam search hypotheses is a difficult problem: most candidates are very similar and usually describe the same events with different words. Figure 1 shows the histogram of the beam hypothesis indices that give the maximum SPIDEr score possible, for each candidate list when using a beam size of five. It reveals that no beam index seems better than another *a priori*. The same conclusion can be drawn with Clotho. We tried to automatically select the best candidate using several features: vocabulary size, sentence length, and even with a shallow neural network trained to rank the sentences, but all these approaches failed to significantly improve the global SPIDEr score.

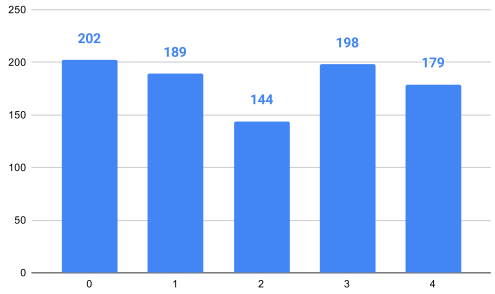


Figure 1: SPIDEr best beam indexes on AudioCaps testing subset

To overcome these limitations, we propose to consider all the candidates produced by the model and select the best SPIDEr score between them with a new metric.

6. SPIDEr-max

6.1. Definition

We propose SPIDEr-max, defined by the following equation:

$$\text{SPIDEr-max}(C, R) = \max_i \text{SPIDEr}(C_i, R) \quad (3)$$

where C is a list of N caption candidates and R a list of references.

It consists of retaining the largest SPIDEr score among the scores calculated for a set of caption candidates, to avoid having to choose a single hypothesis. The SPIDEr-max values are between 0 and 5.5, like the SPIDEr score. The source code in PyTorch will be made available on GitHub[5] upon paper acceptance.

6.2. Results

The score of SPIDEr-max highly depends on how many candidates we use, so we report results with various beam sizes in figures 2 and 3 for AudioCaps and Clotho, respectively. We varied the beam size from 1 to 10. If we imagine a human end-user, proposing at most 5 candidates captions would be reasonable, in our opinion.

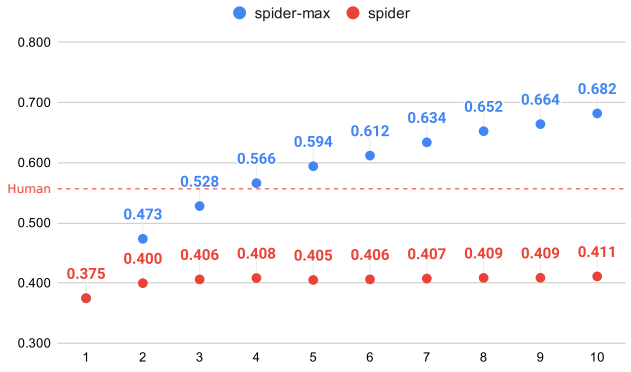


Figure 2: SPIDEr and SPIDEr-max scores with different beam sizes, calculated on the AudioCaps testing subset with CNN10-Transformer.

On AudioCaps, the SPIDEr-max score increases rapidly above the score of our model from 0.401 to 0.473 with only a beam size of two. The scores continue to rise above the human SPIDEr score (0.565), meaning that our model is already producing human-like captions, but fail to select them, if we take the maximum likelihood criterion.

On Clotho, the scores also increase with a higher beam size, but they do not reach the cross-reference score on the first beam sizes. This is probably due to the references of Clotho, which show more diversity in terms of vocabulary and n-grams than AudioCaps.

We also tried to compute the SPIDEr-max score for a beam size equal to 100, which gave 0.953 on AudioCaps and 0.535 on Clotho, but we decided to focus on a few candidates, as it would be more realistic in a real scenario, where, for instance, automatic captions are proposed to an end-user.

[5] https://github.com/Labbeti/spider-max

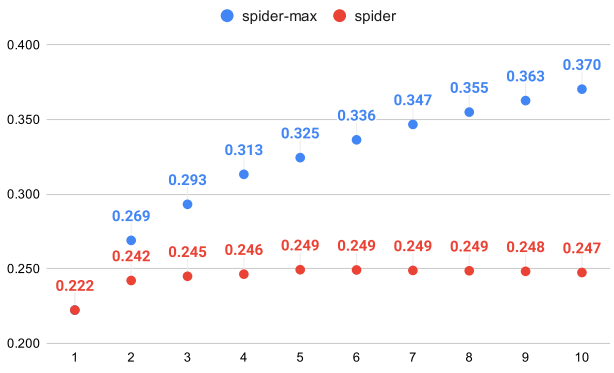


Figure 3: SPIDEr and SPIDEr-max scores with different beam sizes, calculated on the Clotho development-testing subset with CNN10-Transformer.

6.3. Why such a boost in SPIDEr-max?

As we saw in the previous section, SPIDEr-max increases rapidly and even outreaches the human SPIDEr score on AudioCaps. We also noticed that predicting a correct infrequent n-gram seems to drastically improves the score of a candidate, probably due to the CIDEr-D metric based on the TF-IDF of the n-grams. To see if there is a relation between TF-IDF and the SPIDEr and SPIDEr-max scores, we computed the difference between them with the best candidate given by the model and the best one given by the SPIDEr score for various beam sizes.

The correlation value between this variation of TF-IDF and SPIDEr scores is almost one for AudioCaps and Clotho. It suggests that the candidates selected by SPIDEr-max have a much higher TF-IDF than those selected by the model probabilities, which appears to significantly increase the CIDEr-D score and, thus, also the SPIDEr-max score.

7. CONCLUSION

In this paper, we showed that the SPIDEr score is very sensitive to the words used in the caption candidates, so we proposed a new metric, SPIDEr-max, that takes into account multiple candidates for each audio recording. The scores of SPIDEr-max compared to human scores of SPIDEr show that our model already produces human-like caption candidates, but selecting the caption with the highest SPIDEr score is not trivial. As future work, we are interested to study other metrics that do not use TF-IDF, such as model-based metrics like BERTScore [23] or FENSE [24]. We also look forward to testing SPIDEr-max with new models to see if our findings are repeated across architectures and training methods.

8. ACKNOWLEDGMENT

This work was partially supported by the Agence Nationale de la Recherche LUDAU (Lightly-supervised and Unsupervised Discovery of Audio Units using Deep Learning) project (ANR-18-CE23-0005-01). Experiments presented in this paper were carried out using the OSIRIM platform that is administered by IRIT and supported by CNRS, the Region Midi-Pyrénées, the French Government, ERDF (see http://osirim.irit.fr/).

9. REFERENCES

[1] X. Xu, M. Wu, and K. Yu, "A comprehensive survey of automated audio captioning," *arXiv preprint arXiv:2205.05357*, 2022.

[2] Q. Kong, Y. Cao, T. Iqbal, Y. Wang, W. Wang, and M. D. Plumbley, "PANNs: Large-Scale Pretrained Audio Neural Networks for Audio Pattern Recognition," *arXiv:1912.10211 [cs, eess]*, Aug. 2020, arXiv: 1912.10211.

[3] X. Mei, Q. Huang, X. Liu, G. Chen, J. Wu, Y. Wu, J. Zhao, S. Li, T. Ko, H. L. Tang, X. Shao, M. D. Plumbley, and W. Wang, "An encoder-decoder based audio captioning system with transfer and reinforcement learning for DCASE challenge 2021 task 6," DCASE2021 Challenge, Tech. Rep., July 2021.

[4] F. Gontier, R. Serizel, and C. Cerisara, "Automated audio captioning by fine-tuning bart with audioset tags," in *DCASE 2021 - 6th Workshop on Detection and Classification of Acoustic Scenes and Events*, Virtual, Spain, Nov. 2021.

[5] S. Liu, Z. Zhu, N. Ye, S. Guadarrama, and K. Murphy, "Improved Image Captioning via Policy Gradient optimization of SPIDEr," *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 873–881, Oct. 2017, arXiv: 1612.00370.

[6] R. Vedantam, C. L. Zitnick, and D. Parikh, "CIDEr: Consensus-based Image Description Evaluation," *arXiv:1411.5726 [cs]*, June 2015, arXiv: 1411.5726.

[7] P. Anderson, B. Fernando, M. Johnson, and S. Gould, "SPICE: Semantic Propositional Image Caption Evaluation," *arXiv:1607.08822 [cs]*, July 2016, arXiv: 1607.08822.

[8] D. Klein and C. D. Manning, "Accurate unlexicalized parsing," in *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - ACL '03*, vol. 1. Sapporo, Japan: Association for Computational Linguistics, 2003, pp. 423–430.

[9] W.-N. Hsu, D. Harwath, C. Song, and J. Glass, "Text-Free Image-to-Speech Synthesis Using Learned Segmental Units," Dec. 2020, arXiv:2012.15454 [cs, eess].

[10] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "BLEU: a method for automatic evaluation of machine translation," in *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics - ACL '02*. Philadelphia, Pennsylvania: Association for Computational Linguistics, 2001, p. 311.

[11] C.-Y. Lin, "ROUGE: A package for automatic evaluation of summaries," in *Text Summarization Branches Out*. Barcelona, Spain: Association for Computational Linguistics, July 2004, pp. 74–81.

[12] M. Denkowski and A. Lavie, "Meteor Universal: Language Specific Translation Evaluation for Any Target Language," in *Proceedings of the Ninth Workshop on Statistical Machine Translation*. Baltimore, Maryland, USA: Association for Computational Linguistics, 2014, pp. 376–380.

[13] K. Drossos, S. Lipping, and T. Virtanen, "Clotho: An Audio Captioning Dataset," *arXiv:1910.09387 [cs, eess]*, Oct. 2019, arXiv: 1910.09387.

[14] C. D. Kim, B. Kim, H. Lee, and G. Kim, "AudioCaps: Generating captions for audios in the wild," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, June 2019, pp. 119–132.

[15] J. F. Gemmeke, D. P. W. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio Set: An ontology and human-labeled dataset for audio events," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. New Orleans, LA: IEEE, Mar. 2017, pp. 776–780.

[16] I. Montani, M. Honnibal, M. Honnibal, S. V. Landeghem, A. Boyd, H. Peters, P. O. McCann, M. Samsonov, J. Geovedi, J. O'Regan, G. Orosz, D. Altinok, S. L. Kristiansen, Roman, E. Bot, L. Fiedler, G. Howard, W. Phatthiyaphaibun, Y. Tamura, S. Bozek, murat, M. Amery, B. Böing, P. K. Tippa, L. U. Vogelsang, B. Vanroy, R. Balakrishnan, V. Mazaev, and GregDubbin, "explosion/spaCy: v3.2.1: doc_cleaner component, new Matcher attributes, bug fixes and more," Dec. 2021.

[17] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *CoRR*, vol. abs/1706.03762, 2017.

[18] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *arXiv:1412.6980 [cs]*, Jan. 2017, arXiv: 1412.6980.

[19] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition," in *Proc. Interspeech 2019*, 2019, pp. 2613–2617.

[20] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *proc. NeurIPS*, 2019, pp. 8026–8037.

[21] W. Falcon and .al, "Pytorch lightning," *GitHub. Note: https://github.com/PyTorchLightning/pytorch-lightning*, vol. 3, 2019.

[22] X. Xu, Z. Xie, M. Wu, and K. Yu, "The SJTU system for DCASE2022 challenge task 6: Audio captioning with audio-text retrieval pre-training," DCASE2022 Challenge, Tech. Rep., July 2022.

[23] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi, "BERTScore: Evaluating Text Generation with BERT," *arXiv:1904.09675 [cs]*, Feb. 2020, arXiv: 1904.09675.

[24] Z. Zhou, Z. Zhang, X. Xu, Z. Xie, M. Wu, and K. Q. Zhu, "Can Audio Captions Be Evaluated with Image Caption Metrics?" Jan. 2022, number: arXiv:2110.04684 arXiv:2110.04684 [cs, eess].

MULTI-SCALE ARCHITECTURE AND DEVICE-AWARE DATA-RANDOM-DROP BASED FINE-TUNING METHOD FOR ACOUSTIC SCENE CLASSIFICATION

Joo-Hyun Lee[*]*, Jeong-Hwan Choi*[*]*, Pil Moo Byun*[*]*, and Joon-Hyuk Chang*

Department of Electronic Engineering, Hanyang University, Seoul, Republic of Korea
{jhyun42, brent1104, fordream0309, jchang}@hanyang.ac.kr

ABSTRACT

We propose a low-complexity acoustic scene classification (ASC) model structure suitable for short-segmented audio and fine-tuning methods for generalization to multiple recording devices. Based on the state-of-the-art architecture of the ASC, broadcasting-ResNet (BC-ResNet), we introduce BC-Res2Net that uses hierarchical residual-like connections within the frequency- and temporal-wise convolutions to extract multiscale features while using fewer parameters. We also incorporate the attention and aggregation method proposed in short-utterance speaker verification with BC-Res2Net to achieve high performance. In addition, we train the model with a novel fine-tuning method using a device-aware data-random-drop to avoid optimization for only a few devices. When the amount of data differed for each device in the training dataset, the proposed method gradually dropped the data of the primary device from the mini-batch. The experimental results on the TAU Urban Acoustic Scenes 2022 Mobile development dataset demonstrated the effectiveness of multi-scale modeling in short audio. Furthermore, the proposed training strategy significantly reduced the multi-class cross-entropy loss for various devices.

Index Terms— Acoustic scene classification, multi-scale, data imbalance, fine-tuning, short-segmented audio

1. INTRODUCTION

Remarkable progress in acoustic scene classification (ASC) has been accomplished with the development of deep learning, and several studies have recently been conducted to implant deep neural networks (DNNs) into low-resource devices that are suitable for practical applications [1, 2]. Notably, the Detection and Classification of Acoustic Scenes and Events (DCASE) Challenge have been held with various audio tasks, including ASC, and contributes to advance computational environmental audio analysis techniques. In DCASE Task 1, ASC systems should satisfy constraints such as data imbalance depending on the recording device, low complexity limitations, and short audio input while ensuring high classification accuracy [3, 4].

Owing to the limitation on the number of parameters, convolutional neural networks (CNNs) are preferred over DNNs for ASC models. To further enhance the feature extraction capability of CNNs, the ResNet [5] structure and depth-wise separable CNNs (DW-CNNs) [6] were adopted in [7] and [8], respectively. Moreover, the modified MobileNet [9], EfficientNet [10], and broadcasting-ResNet (BC-ResNet) [11] structures, which were designed to consider computational power, exhibited excellent performance [3, 12]. To improve the generalization of the model to the multiple devices, ResNorm was proposed with BC-ResNet, which normalized the frequency bands with the residual path [12].

Several studies introduced the model architecture in the speaker verification field, which focused on the frequency of speech and channels in CNN to improve the verification performance in short-duration speech [13–15]. Liu *et al*. [14] conducted a multiscale frequency-channel attention (MFA) framework with frequency-channel attention and Res2Net structure [16], which learned multiscale features to emphasize the significant frequency and channel components. Jung *et al*. [15] proposed a feature pyramid module (FPM) that upsamples in a top-down pathway to effectively aggregate various-resolution feature maps.

This paper introduces two strategies for generalized ASC for various devices under low complexity and short input time conditions: improving the model structure, and training with a novel fine-tuning method. Inspired by Res2Net, we propose BC-Res2Net that are accomplished by modifying the BC-ResNet structure with multiscale modeling to increase the receptive field size of each CNN. We integrate the BC-Res2Net with MFA and FPM, which effectively extract and aggregate features from short speech signals, to construct the ASC model. Subsequently, we perform device-aware data-random-drop-based fine-tuning that drops data of the selected device in batch-level processing for the pretrained model to obtain consistent performance in various recording devices. We choose the device that recorded the most in the training dataset. We do not drop the data at the beginning of the fine-tuning but gradually increased the drop rate to a given parameter. In addition, we add regularization with a cross-entropy loss to avoid overfitting devices that are not selected.

2. ASC MODEL ARCHITECUTRE

2.1. BC-Res2Net structure

Broadcasted residual learning [11] residually connects two-dimensional (2D) and one-dimensional (1D) feature maps with the input. These different sized feature maps are extracted by frequency-wise 2D and temporal-wise 1D DW-CNNs, respectively, and they contain frequency and frequency-aware temporal features. To correct the size mismatch, the output of the 1D DW-CNN, a 1D feature map is broadcast along the frequency axis. This residual connection was combined with basic structures such as ResNet and Transformer [17] to obtain state-of-the-art results in various audio and speech fields [11, 12, 18] where it is paramount to effectively capture frequency-time characteristics.

Res2Net, however, computes more efficiently than the conventional convolution-based structure because it comprises several CNNs connected in a hierarchical residual-like manner. In particular, the input feature map is sliced precisely with the same channel

[*]Equal contributions.

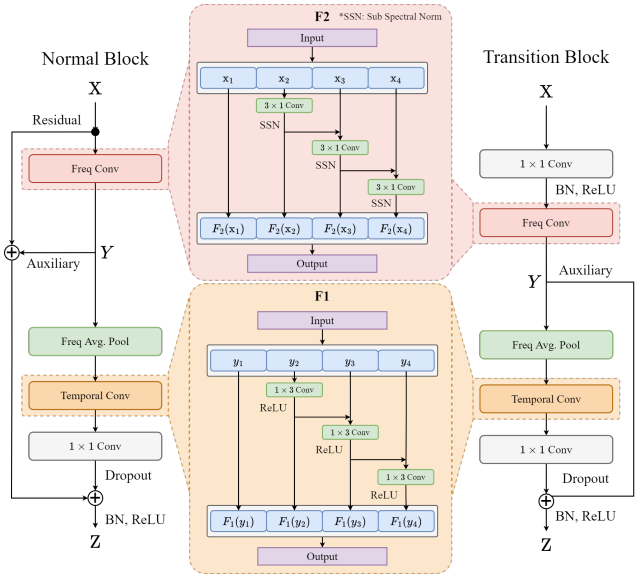


Figure 1: BC-Res2Net structure.

size of CNNs. Each partitioned feature subset has a corresponding CNN that is fed while being sequentially added to the output of the previous CNN. The CNN outputs have various receptive field sizes owing to the multiscale operation. To extract the frequency and frequency-aware temporal features in a multiscale manner, we propose the BC-Res2Net that converts the frequency- and temporal-wise convolution of the BC-ResNet into a Res2Net structure. Figure 1 shows the network blocks of the BC-Res2Net comprising $\mathbf{F}_2$, $\mathbf{F}_1$, frequency average pooling, and point-wise 1D CNN, where $\mathbf{F}_2$ and $\mathbf{F}_1$ denote the Res2Net style 2D and 1D convolutions including nonlinear functions, respectively. Because the transition block is used when the given input and output feature map sizes are assigned differently, we describe the proposed structure based on the normal blocks.

The input feature map $\mathbf{X} \in \mathbb{R}^{C\times F\times T}$ is sliced into S subfeature maps along with the channel axis, and the feature map subset is denoted by $\{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_s\}$. Except for the first subfeature map $\mathbf{x}_1$, each $\mathbf{x}_i \in \mathbb{R}^{C/S\times F\times T}$ has a corresponding frequency-wise 2D DW-CNN with a kernel size of 3×1 and subspectral normalization [19], denoted by $f_{2C,i}(\cdot)$ and $f_{SSN,i}(\cdot)$, respectively. The feature subset $\mathbf{x}_i$ is sequentially fed into $f_{2C,i}(\cdot)$ and $f_{SSN,i}(\cdot)$, after adding the output of $f_{SSN,i-1}(\cdot)$. The overall process for $\mathbf{F}_2$ can be formulated as follows:

$$\mathbf{F}_2(\mathbf{x}_i) = \begin{cases} \mathbf{x}_i, & \text{if } i = 1 \\ f_{SSN,i}(f_{2C,i}(\mathbf{x}_i)), & \text{if } i = 2 \\ f_{SSN,i}(f_{2C,i}(\mathbf{x}_i + \mathbf{F}_2(\mathbf{x}_{i-1}))), & \text{otherwise} \end{cases} \quad (1)$$

Frequency average pooling is applied after concatenating the set of outputs $\{\mathbf{F}_2(\mathbf{x}_1), \mathbf{F}_2(\mathbf{x}_2), ..., \mathbf{F}_2(\mathbf{x}_s)\}$ along the channel axis. The obtained feature map $Y \in \mathbb{R}^{C\times 1\times T}$ is then fed into $\mathbf{F}_1$ to extract the temporal characteristics. $\mathbf{F}_1$ can be expressed as follows:

$$\mathbf{F}_1(y_i) = \begin{cases} y_i, & \text{if } i = 1 \\ f_{ReLU,i}(f_{1C,i}(y_i)), & \text{if } i = 2 \\ f_{ReLU,i}(f_{1C,i}(y_i + \mathbf{F}_1(y_{i-1}))), & \text{otherwise} \end{cases} \quad (2)$$

where $y_i \in \mathbb{R}^{C/S\times 1\times T}$ denotes the subfeature map of Y that is sliced into S along with the channel axis, and $f_{1C,i}$, and $f_{ReLU,i}(\cdot)$ denote a corresponding temporal-wise 1D DW-CNN with a kernel size of 3 and the ReLU activation, respectively. The following operations are performed sequentially: concatenating outputs of $\mathbf{F}_1$ into one, point-wise convolution, channel-wise dropout, and expanding the feature map size $\mathbb{R}^{C\times 1\times T}$ to $\mathbb{R}^{C\times F\times T}$ along with the frequency axis. Finally, batch normalization with ReLU activation is applied after combining the output with two auxiliary residuals that the input identity and result of $\mathbf{F}_1$. Note that the BC-Res2Net operates with fewer CNN parameters than the BC-ResNet because the first subfeature map sliced from $\mathbf{F}_2$ and $\mathbf{F}_1$ does not proceed with convolution. The transition block differs from the normal block in two ways: auxiliary point-wise 2D CNN is applied before $\mathbf{F}_2$ to change the input channel size, and there is no residual connection for identity due to the size difference between the input and output.

2.2. ASC model for MFA and FPM

Short audio or speech makes feature extraction difficult because of insufficient temporal information. Several studies have focused on adding or enhancing the DNN structure in speaker verification fields to overcome performance degradation under short-utterance situations. In this study, we combined MFA and FPM, which improved the feature maps using the attention mechanism and aggregated the features from multiple resolutions, respectively, with the BC-Res2Net structure to introduce the ASC model for short audio.

Table 1: Architectures of proposed BC-Res2Net-based ASC model. T ,F, and C denote the number of time sequences, frequency bins, and CNN channel respectively. Input feature size is $1 \times F \times T$.

Output size	Stage	Operator
$2C \times F/2 \times T/2$	Stem	Conv2D $[5 \times 5]$, stride 2 BatchNorm + MFA
$C \times F/2 \times T/2$	Stage 1	BC-Res2Net $\times$ 2 ResNorm + MFA
$1.5C \times F/4 \times T/4$	Stage 2	Max-pool $[2 \times 2]$ BC-Res2Net $\times$ 2 ResNorm + MFA
$2C \times F/8 \times T/8$	Stage 3	Max-pool $[2 \times 2]$ BC-Res2Net $\times$ 2 ResNorm + MFA
$2.5C \times F/8 \times T/8$	Stage 4	BC-Res2Net $\times$ 3 ResNorm + MFA
$4C \times 1 \times 1$	Aggregation	FPM
# Classes $\times 1 \times 1$	Classifier	Linear

The overall architecture is presented in Table 1. Assigning the importance of the frequency channel components for all the outputs is necessary because the output of each stage has a different resolution and receptive field. Therefore, we apply MFA to Stem and every stage after ResNorm. Figure 2 shows the FPM aggregating the MFA output of each stage in a bottom-up pathway, where the base CNN channel of the proposed model is set to 40. The last three feature maps are upsampled to the size of $\mathbb{R}^{C\times F/2\times T/2}$, which is the same as the output of Stage 1. We considered the average of four feature maps across the frequency-temporal dimension and concatenated them into a single feature map with a size of $\mathbb{R}^{4C\times 1\times 1}$. The upsampling method of the FPM is converted into the pixel shuffle method [20] for greater efficiency compared to the transposed convolution method proposed in [15]. The output of FPM is reshaped through a linear layer according to the number of classes.

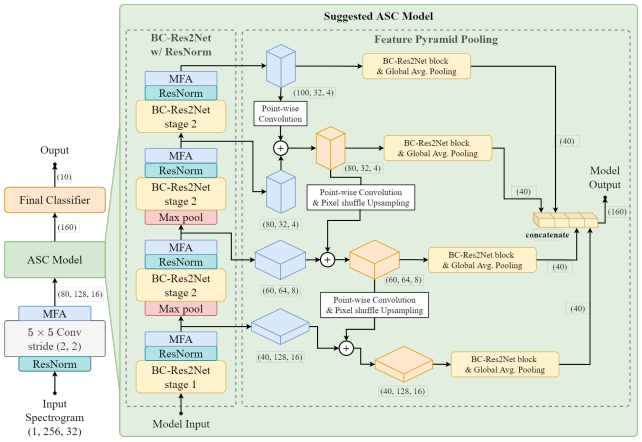


Figure 2: Proposed BC-Res2Net-based ASC model architecture.

3. FINE-TUNING METHOD USING DEVICE-AWARE DATA-RANDOM-DROP

When we train the ASC model with a significantly different amount of data for each device, the model undertakes greater effort to characterize a few specific devices occupying a large percentage of the dataset than to represent the general features of the overall device. To alleviate this issue, we load a pre-learning model that neglected the device types and fine-tuned the model by removing data from specific devices. Before fine-tuning, the last linear layer that classified the scenes is initialized and retrained to become more device-agnostic. Next, we fine-tune the model using the suggested device-aware data-random-drop method. It selects the device with the most data from the dataset and manages the mini-batch for every step by randomly removing data from it. However, an excessive drop of specific devices from the training process can lead to poor ASC performance owing to a lack of data diversity Therefore, inspired by curriculum learning [21], we design a method in which the data-drop rate is initialized with zero and then increased step-wise. The drop rate gradually increases with the shape of the sigmoid function from zero to the given parameter. Furthermore, we add regularization to minimize the square weight difference between the fine-tuning and pretrained model parameters to prevent excessive loss of information from the selected device as given by:

$$L = L_C + \lambda \sum_i (\theta_i - \widetilde{\theta}_{P,i})^2, \tag{3}$$

where L, L_C, and λ denote the total loss, classification loss, and scaling factor of regularization, respectively. θ and $\widetilde{\theta}_P$ denote the fine-tuned and pretrained model parameters, respectively, except for the classifier layer. The pretrained model parameters are stored in advance.

4. EXPERIMENTAL SETUP

4.1. Datasets and preprocessing

The TAU Urban Acoustic Scenes 2022 Mobile Development dataset [4] had the same format as the 2020 development dataset [3], same sample rate of 44.1 kHz and 24 bits. However, this 2022 dataset segments were significantly shorter (1 s) compared to the last 2020 development set (10 s). In addition, the number of segments grew tenfold as the 2020 dataset split the 2022 dataset by 1 s.

Table 2: Ablation study of the BC-Res2Net evaluated on the TAU Urban Acoustic Scenes 2022 Mobile development dataset. (Acc. indicates the top-1 test accuracy(%).)

Systems	# Params	MACs	Log Loss	Acc.
BC-ResNet-40	88.1K	17.21M	1.327	57.1
BC-Res2Net-40	85.8K	15.89M	1.235	59.1
w/ MFA	123.6K	17.45M	1.198	59.3
w/ FPM	93.6K	17.06M	1.212	59.5
w/ MFA & FPM	126.6K	26.76M	**1.167**	**60.8**

Table 3: Log loss and top-1 test accuracy (%) comparison for different duration of test audio on the TAU Urban Acoustic Scenes 2020 Mobile development dataset. (Dur. indicates durations.)

Dur.	BC-ResNet-40	BC-Res2Net-40	BC-Res2Net-40 w/ MFA & FPM
1 s	1.327 / 57.1	1.235 / 59.1	**1.167 / 60.8**
2 s	1.285 / 57.8	1.190 / 60.3	**1.146 / 61.6**
5 s	1.301 / 56.7	1.185 / 59.7	**1.172 / 60.5**
10 s	1.315 / 56.3	1.195 /58.7	**1.192 / 59.5**

The audio segments were ten types of acoustic scenes from ten cities, recorded from three real devices (A, B, and C) and six simulated devices (S1–S6). According to the train-split method of [4], development dataset 2022 was separated into training and test subsets comprising 139,970 and 29,680 segments, respectively. In the training subset, the data for Device A accounted for 73% of the total. In the test split, the data from all the devices were evenly distributed. The test split contained data recorded with devices S4-S6, which were excluded in the training data split. The evaluation dataset was provided without labels for submitting the results.

We used the log Mel spectrum as the input feature for our system. The input features were prepared through three steps: downsampling from 44.1 kHz to 16.0 kHz, log Mel spectrum feature extraction, and data augmentation. The log Mel spectrograms were 256-dimensional, extracted with 2048 samples of the Hanning window, and 512 sample shifts. The input feature size obtained using the preprocessing method mentioned was [1, 256, 32]. The time-rolling method was used for time-domain augmentation. The input audio was randomly rolled along the time axis, ranging from -0.5-0.5 s, with out-of-range parts shifted to the other side. Specaugment [22], except time wrapping, was also employed with two frequency and temporal masks each. Mask parameters of 40 and 4 were used for the frequency and temporal masks, respectively. Each time-rolling and Specaugment mask was applied with a probability of 0.8. We also applied Mixup [23] with $\alpha = 0.3$ to the acoustic feature space.

4.2. Implementation details

We trained the BC-Res2Net-based ASC model with pretraining and fine-tuning phases. In the pretraining phase, the AdamW optimizer [24] with a weight decay of 0.05 was used over 300 epochs, and the mini-batch size was set to 512. Warmup [25] was applied, where the learning rate linearly increased from 1e-8 to 0.01 over the first ten epochs and decayed to zero with a cosine annealing scheduler [26]. We applied a device-aware data-random-drop, treating the selected Device A as an excluded recording device in the fine-tuning phase. The mixup was disabled to correct the mismatch between the training and test conditions. The scaling factor of regularization was 0.4, and the AdamW optimizer with a weight decay of 1e-8 and fixed

Table 4: Device-wise top-1 test accuracy (%) and overall log loss comparison of proposed fine-tuning method according to maximum drop rate on the TAU Urban Acoustic Scenes 2022 Mobile development dataset. (Acc. indicates the top-1 test accuracy (%).)

Systems	Fine-tuning (Drop rate)	Seen device						Unseen device			Average Log loss / Acc.
		A	B	C	S1	S2	S3	S4	S5	S6	
BC-Res2Net-40 w/ MFA & FPM	✗	72.0	64.5	68.2	62.7	59.5	64.2	54.4	56.4	45.3	1.167 / 60.8
	✓ (0.00)	72.8	**68.1**	69.7	63.1	62.0	66.2	53.4	56.0	47.6	1.083 / 62.1
	✓ (0.50)	73.0	67.8	70.1	62.9	61.6	**66.5**	55.2	56.8	48.4	1.085 / 62.5
	✓ (0.90)	**73.2**	68.0	69.4	**63.7**	61.6	66.2	55.2	57.3	**49.0**	**1.076 / 62.6**
	✓ (0.99)	72.7	67.2	**70.3**	63.0	**62.2**	66.0	**55.9**	**57.4**	48.8	1.081 / 62.6

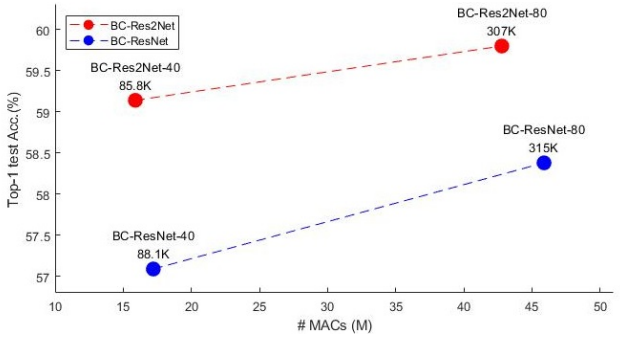


Figure 3: Top-1 test accuracy comparison of the BC-Res2Net and the BC-ResNet according to model parameters and MACs.

learning rate of 1e-5 was used. For the model structure, both $\mathbf{F}_2$ and $\mathbf{F}_1$ of the BC-Res2Net were sliced into four subchannels. Sub-spectral normalization [19] with four sub-bands and ResNorm with 0.1 (hyperparameter of the identity shortcut path,) were applied to the BC-Res2Net.

5. RESULT

We evaluated the result in terms of the top-1 test accuracy and multiclass cross-entropy (log loss). We also reported the number of model parameters and multiply-accumulate operations (MACs) were used to observe computational complexity. Figure 3 shows the comparison of the BC-ResNet and BC-Res2Net when the base CNN channel size increases from 40 to 80. For all the CNN channel sizes, the BC-Res2Net achieves higher accuracy than the BC-ResNet while having small MACs and model parameters. Table 2 presents the effects of the proposed structural modifications on the ASC model. The BC-Res2Net-40 requires 2.6% fewer parameters and 7.7% fewer MACs than the BC-ResNet-40 but performs better in terms of log loss and accuracy. When MFA and FPM were applied to the BC-Res2Net, the accuracy improved by 0.2% and 0.3%, respectively; when both were applied, the accuracy improved by 1.7%. Table 3 shows the results for the short audio conditions. We evaluated the cropped test data with the given duration within each 10 s audio of the 2020 data. The BC-Res2Net performed better than the BC-ResNet for all the test lengths; in particular, the model that added MFA and FPM to the BC-Res2Net obtained better results at shorter durations of 1 s and 2 s. These results show that the BC-Res2Net extracts the information required to classify the scenes more effectively than the BC-ResNet, and using MFA and FPM additionally assists in classifying sound in short-segmented audio. Table 4 presents the effect of the fine-tuning method based on the maximum drop rate. Compared with the pretrained model, the overall accuracy and log loss improved by 0.084 and 1.3%, respectively; when the fine-tuning was applied without data-random-drop, and better performance was achieved when the drop was applied to the selected Device A. Maximum drop rate of 0.9 exhibited the best average log loss and accuracy, and achieved significant improvements of 0.091 and 1.8%, respectively, compared to the case when fine-tuning was not applied. In particular, on the seen device, the performance of multiple devices including Device A was improved evenly, and the performance improvement was observed even in the unseen device, showing that the proposed fine-tuning method benefits generalization of the device.

6. RELATIONSHIPS WITH TECHNICAL REPORT

In a technical report [27], quantization-aware training (QAT) [28] was additionally introduced to satisfy the quantization conditions of INT8. The log loss and accuracy of the QAT-applied BC-Res2Net-40-based ASC model were degraded to be 1.193 and 60.3%, respectively, compared with the results without quantization. We submitted the outputs of the two systems with the proposed fine-tuning in QAT environment. To investigate the effect of the proposed fine-tuning according to the regularization scaling factor, we submitted systems results trained with the fine-tuning method with a drop rate of 0.9 and the regularization factors with 0.04 and 0.4. Each result achieved log losses of 1.072 (Acc. 62.2%) and 1.065 (Acc. 62.6%), respectively, for the test set of the development dataset. For the rest of the two trials, we additionally applied knowledge distillation [29] introduced in [7] to improve the performance. The best result was assigned the teacher model size equal to the student model and the scaling factor of regularization to 0.4 and achieved a log loss of 0.835 and accuracy of 70.1% from the development dataset. Finally, this result achieved a log loss of 1.147 and accuracy of 60.8% in the challenge evaluation and placed second in the competition.

7. CONCLUSION

We proposed the BC-Res2Net by modifying the BC-ResNet in a multiscale manner. Moreover, we improved ASC performance under short audio evaluation conditions by using the MFA and the FPM method, which finds important components among frequency and channel components and effectively aggregates feature maps of different resolutions. Finally, we suggested the device-aware data-random-drop method-based fine-tuning method to promote optimization for multiple devices.

8. ACKNOWLEDGMENT

This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No. 2021-0-00456, Development of Ultra-high Speech Quality Technology for Remote Multi-speaker Conference System)

9. REFERENCES

[1] M. Valenti *et al*., "DCASE 2016 acoustic scene classification using convolutional neural networks," in *Proc. Workshop Detection Classif. Acoust. Scenes Events*, 2016, p. 95–99.

[2] S. Chu, S. Narayanan, and C.-C. J. Kuo, "Environmental sound recognition with time–frequency audio features," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 17, no. 6, pp. 1142–1158, 2009.

[3] I. Martín-Morató, A. Ancilotto, T. Heittola, A. Mesaros, and T. Virtanen, "Low-complexity acoustic scene classification for multi-device audio: analysis of DCASE 2021 challenge systems," *arXiv:2105.13734*, 2021.

[4] I. Martín-Morató, F. Paissan, A. Ancilotto, T. Heittola, A. Mesaros, E. Farella, A. Brutti, and T. Virtanen, "Low-complexity acoustic scene classification in DCASE 2022 challenge," *arXiv:2206.03835*, 2022.

[5] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *arXiv:1512.03385*, 2015.

[6] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient convolutional neural networks for mobile vision applications," *arXiv:1704.04861*, 2017.

[7] B. Lehner and K. Koutini, "Acoustic scene classification with reject option based on ResNets," *DCASE 2019 Challenge, Tech. Rep.*, 2019.

[8] T. Heittola, A. Mesaros, and T. Virtanen, "Acoustic scene classification in DCASE 2020 challenge: generalization across devices and low complexity solutions," in *Proc. Detection and Classification of Acoustic Scenes and Events* (DCASE) *Workshop*, 2020, p. 56–60.

[9] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition* (CVPR), 2018, p. 4510–4520.

[10] M. Tan and Q. V. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in *Proc. International Conference on Machine Learning* (ICML), vol. 97, 2019, pp. 6105–6114.

[11] B. Kim, S. Yang, J. Lee, and D. Sung, "Broadcasted residual learning for efficient keyword spotting," in *Proc. INTERSPEECH*, 2021, p. 4538–4542.

[12] B. Kim, S. Yang, J. Ki, and S. Chang, "Domain generalization on efficient acoustic scene classification using residual normalization," in *Proc. Detection and Classification of Acoustic Scenes and Events* (DCASE) *Workshop*, 2021, p. 21–25.

[13] J.-H. Choi, J.-Y. Yang, and J.-H. Chang, "Short-utterance embedding enhancement method based on time series forecasting technique for text-independent speaker verification," in *IEEE Automatic Speech Recognition and Understanding Workshop* (ASRU), 2021, pp. 130–137.

[14] T. Liu, R. K. Das, K. Aik Lee, and H. Li, "MFA: TDNN with multi-scale frequency-channel attention for text-independent speaker verification with short utterances," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing* (ICASSP), 2022, pp. 7517–7521.

[15] Y. Jung, S. M. Kye, Y. Choi, M. Jung, and H. Kim, "Improving multi-scale aggregation using feature pyramid module for robust speaker verification of variable-duration utterances," in *Proc. INTERSPEECH*, 2020, pp. 1501–1505.

[16] S.-H. Gao, M.-M. Cheng, K. Zhao, X.-Y. Zhang, M.-H. Yang, and P. Torr, "Res2Net: A new multi-scale backbone architecture," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 6, p. 652–662, Feb. 2021.

[17] A. Vaswani *et al*., "Attention is all you need," in *Proc. Advances in Neural Information Processing Systems* (NIPS), 2017, p. 3830–3834.

[18] J.-H. Choi, J.-Y. Yang, Y.-R. Jeoung, and J.-H. Chang, "Improved CNN-Transformer using broadcasted residual learning for text-independent speaker verification," in *Proc. INTERSPEECH*, 2022.

[19] S. Chang, H. Park, J. Cho, H. Park, S. Yun, and K. Hwang, "Subspectral normalization for neural audio data processing," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing* (ICASSP), 2021, pp. 850–854.

[20] W. Shi *et al*., "Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition* (CVPR), 2016, pp. 1874–1883.

[21] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," in *Proc. International Conference on Machine Learning* (ICML), 2009, p. 41–48.

[22] D. S. Park *et al*., "SpecAugment: A simple data augmentation method for automatic speech recognition," in *Proc. INTERSPEECH*, 2019, pp. 2613–2618.

[23] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "Mixup: Beyond empirical risk minimization," in *Proc. International Conference on Learning Representations* (ICLR), 2018.

[24] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *Proc. International Conference on Learning Representations* (ICLR), 2019.

[25] P. Goyal, "Accurate, large minibatch SGD: Training imagenet in 1 hour," *arxiv:1706.02677*, 2017.

[26] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "SGDR: Stochastic gradient descent with warm restarts," in *Proc. International Conference on Learning Representations* (ICLR), 2017.

[27] J.-H. Lee, J.-H. Choi, P. M. Byun, and J.-H. Chang, "HYU submission for the DCASE 2022: fine-tuning method using device-aware data-random-drop for device-imbalanced acoustic scene classification," *DCASE 2022 Challenge, Tech. Rep.*, 2022.

[28] https://pytorch-lightning.readthedocs.io/en/stable/_modules/pytorch_lightning/callbacks/quantization.html.

[29] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," in *Proc. NIPS Deep Learning and Representation Learning Workshop*, 2015.

UNSUPERVISED ANOMALOUS SOUND DETECTION FOR MACHINE CONDITION MONITORING USING TEMPORAL MODULATION FEATURES ON GAMMATONE AUDITORY FILTERBANK

*Kai Li, Quoc-Huy Nguyen, Yasuji Ota, and Masashi Unoki**

School of Information Science,
Japan Advanced Institute of Science and Technology
1-1, Asahidai, Nomi, Ishikawa, 923-1292, Japan
{kai_li, hqnguyen, y_ota, unoki}@jaist.ac.jp

ABSTRACT

Anomalous sound detection (ASD) is a technique to determine whether the sound emitted from a target machine is anomalous or not. Subjectively, timbral attributes, such as sharpness and roughness, are crucial cues for human beings to distinguish anomalous and normal sounds. However, the feature frequently used in existing methods for ASD is the log-Mel-spectrogram, which is difficult to capture temporal information. This paper proposes an ASD method using temporal modulation features on the gammatone auditory filterbank (TMGF) to provide temporal characteristics for machine-learning-based methods. We evaluated the proposed method using the area under the ROC curve (AUC) and the partial area under the ROC curve (pAUC) with sounds recorded from seven kinds of machines. Compared with the baseline method of the DCASE2022 challenge, the proposed method provides a better ability for domain generalization, especially for machine sounds recorded from the valve.

Index Terms— Anomalous sound detection, gammatone filterbank, temporal modulation features, timbre information, deep learning

1. INTRODUCTION

Anomalous sound detection (ASD) is a technique to determine whether the sound recorded from a target machine is anomalous or not. It enables workers to arrange maintenance work to fix machine problems in the earliest stages, thus reducing maintenance costs and preventing consequential damages. ASD for machine condition monitoring purposes has received increasing attention.

ASD is often viewed as an unsupervised problem due to difficulties in collecting anomalous sounds that can cover all possible types of anomalies. Autoencoder (AE)-based unsupervised methods, such as those in [1, 2, 3], were popularly used. These methods simulated the distribution of normal sounds by minimizing the reconstruction error of normal training data. Then, the reconstruction scores from the testing data were used to detect the anomalies. Some improved AE models, such as Heteroskedastic Variational AE (HVAE) [4] and Conformer-based AE [5], have also been proposed to improve the performance of ASD. However, the performance of AE-based ASD systems depends significantly on the discrimination of input features.

The log-Mel-spectrogram (LMS) is widely used as input feature in ASD [1, 3, 6]. It is designed in accordance with the pitch perception of the human ear and has high resolution in the low frequency and low resolution in the high frequency [7]. However, the discriminative information of sounds emitted from different kinds of machines may be encoded non-uniformly in the frequency domain. The Mel filterbank may filter out important information concealed in the high-frequency components and hence decrease the performance of an ASD system. Furthermore, the LMS focuses on discriminative information from the frequency domain, making it difficult to capture temporal information.

Because of the drawbacks of the LMS, other ASD methods considered temporal information to improve detection results. In [8], a temporal feature is extracted from the raw waveform by a CNN-based network (TgramNet) to compensate for the anomalous information unavailable from the LMS. This complementary information can further improve the results of ASD systems. However, there is still a lot of redundant information with the raw waveform as a front-end feature and cannot distinguish between normal and anomalous sounds well.

For human beings, it is pretty easy to distinguish anomalous and normal sounds by perceiving auditory attributes (loudness, pitch, and timbre), especially timbral attributes, such as sharpness and roughness [9]. A feature that includes more timbral information is crucial for perceptually distinguish anomalous and normal sounds. However, a specially designed feature from the perspective of human perception for ASD has not been developed.

This paper proposes a method to use temporal modulation features on the gammatone auditory filterbank (TMGF) [10] combined with a simple AE-based detector for the ASD task. This paper assumes that the TMGF feature can provide much more information related to human perception, especially timbral attibutes. The proposed method is evaluated by experiments on the Task 2 dataset of the DCASE2022 challenge [1]. The results show that the proposed method outperforms the baseline system in the target evaluation.

2. BASELINE METHOD

The AE-based system was selected as a baseline [1]. In the baseline system, the LMS of the input audio $X = \{X_t\}_{t=1}^{T}$ was extracted and fed into an AE-based detector, where $X_t \in \mathbb{R}^F$, F and T are the number of Mel-filters and time-frames, respectively.

*Corresponding author. This work was supported by SCOPE Program of Ministry of Internal Affairs and Communications (Grant Number: 201605002) and the Fund for the Promotion of Joint International Research (Fostering Joint International Research (B))(20KK0233).

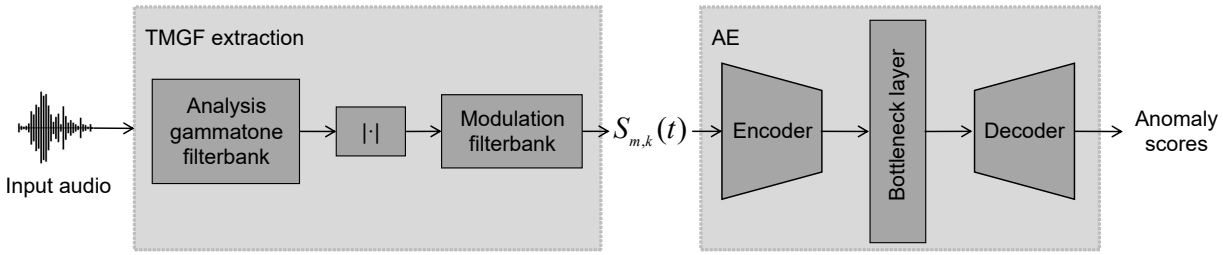


Figure 1: Proposed system using temporal modulation features on the gammatone auditory filterbank (TMGF, $S_{m,k}(t)$) for anomalous sound detection (ASD).

Then, the acoustic feature at t is obtained by concatenating consecutive frames of the log-Mel-spectrogram as $\delta_t = (X_t, ..., X_{t+P-1})$, where $D = P \times F$, P is the number of frames of the context window. The anomaly score is calculated as

$$A_\theta(X) = \frac{1}{DT}\sum_{t=1}^{T} ||\delta_t - \mathbb{F}(\delta_t)||_2^2, \tag{1}$$

where $\mathbb{F}(\cdot)$ is the vector reconstruct function using the AE model, and $||\cdot||_2$ is ℓ_2 norm. As shown in Fig. 1, the AE model includes an encoder, bottleneck layer, and decoder modules. All modules consist of fully-connected layers. The training of the AE model is a regression mission due to the fact that only normal sounds can be used in model training. Finally, the mean squared error (MSE) is used as the cost function to optimize the overall system.

To determine the anomaly detection threshold, the baseline method assumes that A_θ follows a gamma distribution. The gamma distribution parameters are estimated from the histogram of A_θ, and the anomaly detection threshold is determined as the 90th percentile of the gamma distribution. If A_θ for each test clip is greater than this threshold, the clip is judged to be abnormal; otherwise, it is judged to be normal.

3. PROPOSED TMGF FEATURES

The temporal modulation on an auditory filterbank contains important information related to the timbre of a sound, such as the sharpness, roughness, and fluctuation strength [11, 12, 13]. Such information visualizes how humans perceive a sound as well as how we judge a sound (i.e., as "anomalous" or "normal"). Also, different frequencies of temporal modulation contain different levels of speech information such as speech intelligibility, speaker identity, and emotion. Thus, we aim to utilize the temporal modulation feature for detecting anomalous sound. The extraction processes are based on those from Huy. et al. [10].

The gammatone filter [14] is a well-known auditory filter model. The impulse response of a gammatone analysis filter at the center frequency f_c is defined as

$$g(t) = at^{n-1}e^{-2\pi b\mathrm{ERB}(f_c)t}e^{j2\pi f_c t}, \tag{2}$$

where $t \geq 0$ is time in seconds, a is the amplitude, n is the filter order, and b is the bandwidth coefficient. The equivalent rectangular bandwidth $\mathrm{ERB}(f_c)$ is defined as

$$\mathrm{ERB}(f_c) = 24.7 + 0.108 f_c\,. \tag{3}$$

Using K gammatone filters $\{g^{(k)}(t)\}_{k=0}^{K-1}$ with different center frequencies, from an input signal $x(t)$, the output of the filterbank $X_k(t)$ can be expressed as the product of the amplitude modulation $A_k(t)$ and the complex carrier $e^{j\phi_k(t)}$, as

$$\begin{aligned} X_k(t) &= x(t) * g^{(k)}(t) \\ &= A_k(t)e^{j\phi_k(t)}\,. \end{aligned} \tag{4}$$

The gammatone filterbank can be implemented using a wavelet transform where the mother wavelet is $\psi(t) = g(t)$ [15]. Then, with an $\alpha > 1$, the k-th filter $g^{(k)}(t)$ can be defined by scaling $\psi(t)$ with a factor α_k of t, as

$$g^{(k)}(t) = \psi\left(\alpha_k t\right)\,, \tag{5}$$

$$\alpha_k = \alpha^{\frac{2k}{K-1}-1}\,. \tag{6}$$

To analyze different frequency components of $A_{k,t}$, we use a modulation filterbank [16, 17] consisting of M filters $\{h^{(m)}(t)\}_{m=1}^{M}$. The first filter $h^{(1)}(t)$ is a low-pass filter with a cut-off frequency of f_1. For each $m \geq 2$, the filter $h^{(m)}(t)$ is a band-pass filter of which the frequency ranges from $2^{m-2}f_1$ to $2^{m-1}f_1$. Using the designed modulation filterbank, the TMGF features can be extracted from the amplitude modulation $A_{k,t}$ as

$$S_{m,k}(t) = A_k(t) * h^{(m)}(t)\,. \tag{7}$$

4. EXPERIMENTAL SETUP

4.1. Datasets

The datasets used in this task were provided by the DCASE2022 organizers [18, 19]. The data includes normal and anomalous sounds recorded from seven machines: fan, gearbox, bearing, slide, tor car, toy train, and valve. Each recorded sound includes the target machine's sounds and environmental sounds. To simplify the task, only the first channel of multi-channel audio is used. The length of each recorded sound is fixed to 10 s, and the sampling rate is 16 kHz.

The data is divided into three datasets: development, additional training, and evaluation. Each dataset includes audio from these seven types of machines. Machines in the development dataset include sections 01, 02, and 03. Machines in the additional training dataset and evaluation dataset include sections 04, 05, and 06. Each section was divided into source and target domains due to the differences in operating speed, machine load, viscosity, heating temperature, type of environmental noise, signal-to-noise ratio (SNR), etc. Different domains are split into a training and testing subset—the training dataset includes normal sounds only, but the testing dataset includes normal and abnormal sounds. In our experiments, training data in the development dataset was used for model training, and test data in the development dataset was used for testing.

Table 1: Overall results of the proposed (TMGF) and baseline (BL) methods in terms of AUC and pAUC.

Machines	Sections	AUC (source)		AUC (target)		pAUC	
		BL (%)	TMGF (%)	BL (%)	TMGF (%)	BL (%)	TMGF (%)
Toy car	0	85.54	62.62	45.06	40.78	51.89	47.79
	1	87.22	67.66	42.02	39.76	53.53	48.42
	2	99.04	71.62	26.44	**42.66**	54.32	**55.53**
	Arithmetic mean	90.60	67.30	37.84	**41.07**	53.25	50.58
	Harmonic mean	90.22	67.10	35.79	**41.03**	53.23	50.35
Toy train	0	66.78	44.26	32.94	25.84	51.63	48.74
	1	77.56	61.82	30.58	**45.92**	50.37	49.37
	2	83.42	45.86	15.92	**49.76**	49.47	**51.05**
	Arithmetic mean	75.92	50.65	26.48	**40.51**	50.49	49.72
	Harmonic mean	75.27	49.53	23.83	**37.23**	50.48	49.70
Bearing	0	50.24	62.86	62.88	**63.46**	51.53	**52.84**
	1	66.12	**66.44**	63.96	62.42	52.79	49.53
	2	42.14	**55.70**	54.74	**62.64**	48.47	**66.05**
	Arithmetic mean	52.83	**61.67**	60.53	**62.84**	50.93	**56.14**
	Harmonic mean	51.06	**61.33**	60.23	**62.84**	50.86	**55.29**
Fan	0	82.04	**84.20**	38.66	**42.00**	59.63	50.11
	1	72.46	51.84	46.04	**49.48**	51.63	50.95
	2	81.84	78.58	65.64	**67.50**	63.89	**64.37**
	Arithmetic mean	78.78	71.54	50.11	**52.99**	58.39	55.14
	Harmonic mean	78.52	68.35	47.75	**50.99**	57.93	54.43
Gearbox	0	64.34	36.02	65.00	49.60	61.26	49.60
	1	65.84	59.22	57.40	54.86	53.63	50.58
	2	74.64	67.96	66.04	**66.22**	62.11	58.05
	Arithmetic mean	68.27	54.40	62.81	56.89	59.00	52.74
	Harmonic mean	67.98	50.54	62.57	56.08	58.74	52.48
Slider	0	80.42	46.26	56.82	45.12	62.21	48.26
	1	67.04	50.22	50.18	**63.06**	53.05	53.05
	2	86.78	23.88	40.82	**53.60**	54.37	48.37
	Arithmetic mean	78.08	40.12	49.27	**53.93**	56.54	49.89
	Harmonic mean	77.17	35.97	48.37	**52.93**	56.27	49.80
Valve	0	54.66	**98.66**	51.96	**98.30**	52.26	**94.37**
	1	50.58	**59.80**	52.06	**60.94**	49.95	**54.16**
	2	50.88	**95.86**	43.40	**97.08**	48.79	**89.11**
	Arithmetic mean	52.04	**84.77**	49.14	**85.44**	50.33	**79.21**
	Harmonic mean	51.98	**80.45**	48.78	**81.34**	50.29	**74.47**
Average	Arithmetic mean	70.93	61.49	48.03	**56.24**	54.13	**56.20**
	Harmonic mean	67.57	55.53	42.53	**51.56**	53.76	**54.26**

4.2. Metrics

To evaluate the performance of an ASD system, the area under the curve (AUC) and partial-AUC (pAUC) for receiver operating characteristic (ROC) curves are used. The pAUC is an AUC calculated from a portion of the ROC curve over a pre-specified range of interest. To increase the reliability, the pAUC is calculated as the AUC over a low false-positive-rate (FPR) range $[0, p]$, where $p = 0.1$ is used. According to [20], the AUC and pAUC for each machine type, section, and domain can be calculated as

$$AUC_{m,n,d} = \frac{1}{N_d^- N_n^+} \sum_{i=1}^{N_d^-} \sum_{l=1}^{N_n^+} \mathcal{H}(\mathcal{A}_\theta(x_l^+) - \mathcal{A}_\theta(x_i^-)), \quad (8)$$

$$pAUC_{m,n} = \frac{1}{\lfloor pN_n^- \rfloor N_n^+} \sum_{i=1}^{\lfloor pN_n^- \rfloor} \sum_{l=1}^{N_n^+} \mathcal{H}(\mathcal{A}_\theta(x_l^+) - \mathcal{A}_\theta(x_i^-)), \quad (9)$$

where m represents the index of a machine type, n represents the index of a section, $d = \{\text{source, target}\}$ represents a domain, $\lfloor \cdot \rfloor$ is the flooring function, and $\mathcal{H}(x)$ returns 1 when $x > 0$ and 0 otherwise. $\{x_i^-\}_{i=1}^{N_-}$ and $\{x_l^+\}_{l=1}^{N_+}$ are normal and anomalous test clips in domain d in section n in machine type m, respectively. N_- and N_+ are the number of normal and anomalous test clips in domain d in section n in machine type m, respectively.

4.3. Experimental conditions

To extract the LMS feature, 10-s audio clips were first split into different frames with frame lengths of 64 ms and hop lengths of 32 ms. Then, the Mel-spectrogram feature is extracted using the $melspectrogram$ module in the $librosa$ library with the following parameters: n_fft=1024, hop_length=512, $T = 128$, and power=2.0. Finally, five Mel-spectrogram features ($P = 5$) were concatenated into one feature vector with a dimension of 640 and fed into the detector.

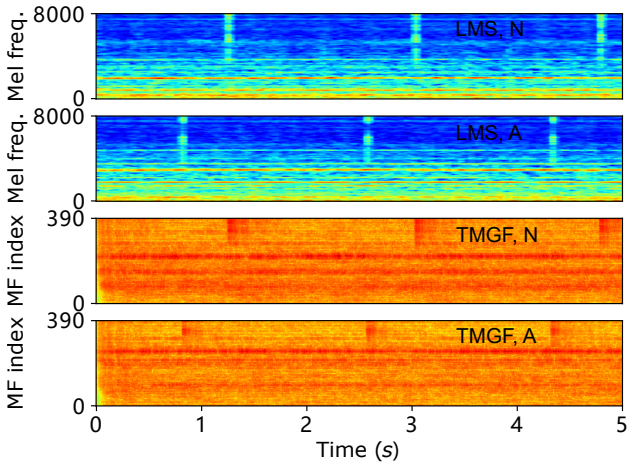


Figure 2: Comparison of the log-Mel-spectrogram (LMS) and the proposed TMGF feature using normal (N) and anomalous (A) sounds emitted from valve. Both sounds are selected from the target domain and have the same pattern. MF: modulation frequency, freq.: frequency.

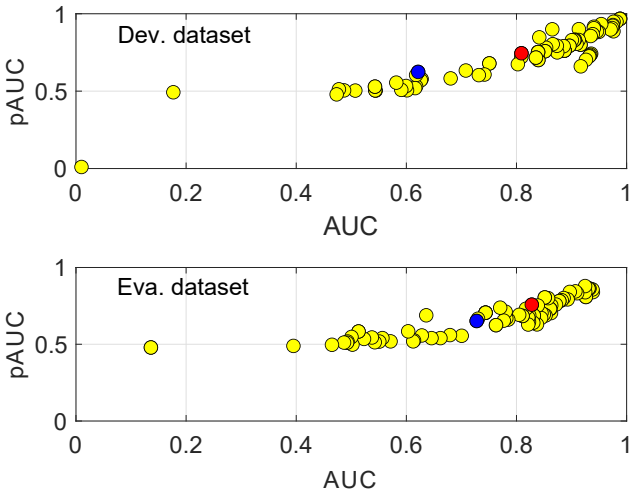


Figure 3: Results of DCASE 2022 challenge using sounds recorded from the valve. Results from both the development (Dev.) dataset and evaluation (Eva.) dataset are depicted. Blue and red circles correspond to baseline and proposed systems, respectively.

In the TMGF feature extraction, we used the gammatone filterbank with $K = 65$ and $\alpha = 10$. For the mother wavelet $\psi(t)$, we set $n = 4$, $b = 1.019$, and $f_c = 600$ Hz. For the modulation filterbank, we used $M = 6$ and $f_1 = 2$ Hz. To decrease the dimension of TMGF feature, downsampling was conducted to decrease the temporal dimension to 1600 Hz. Finally, feature vectors with a fixed dimension of 390 were fed into the detector.

The model had four dense layers with 128 dimensions for the encoder, one bottleneck layer with eight dimensions, and four dense layers with 128 dimensions for the decoder. We trained the model for 100 epochs using the Adam optimizer [21] with a learning rate of 0.0001 and a batch size of 128. The anomaly scores were calculated by the averaged reconstruction error.

5. RESULTS

The overall results are shown in Table 1. This paper compares the results using our proposed method with that of the baseline method. The improved results are highlighted in the table. From these results, we can see that the LMS feature provides better performance in the source evaluations, but the performance significantly degrades in the target evaluation. The proposed method performs better in the target evaluation; even degradation occurs in the source evaluation. This is because of the TMGF feature can capture the sound variances in the time domain easily. It is sensitive to some background noises and irrelevant information. Therefore, the over-fitting problem in the training stage could be alleviated to some extent by using the proposed TMGF feature, hence improving the robustness of a trained ASD system.

The results of the TMGF feature achieve a much better performance in both the source and target evaluation in the valve. This is because timbral information captured by the TMGF feature, such as the sharpness and roughness, is useful for a learning system to find the variance of the 'click' sounds emitted from a valve. By using the TMGF feature, we improved the average arithmetic mean of AUC from 48.03% to 56.24% and the average harmonic mean of AUC from 42.53% to 51.56% in the target evaluation.

Figure 2 shows the differences between the LMS feature and the proposed TMGF using normal and anomalous sounds emitted from the valve. The pattern of these two sounds was consistent. TMGF can capture not only the frequency feature as 'click' sounds but also the time domain feature as timbre-related property.

The results of the DCASE2022 challenge using sounds recorded from the valve are shown in Fig. 3. Each dot corresponds to a different system in the challenge. As we can see, the TMGF can obtain competitive results in the valve even if a simple AE-based detector is used. The AE-based detector has to assume that the learned model cannot reconstruct sounds that are not used in training, that is, unknown anomalous sounds. This assumption is hard to satisfy because the training procedure does not involve anomalous sounds [8, 22]. Therefore, we believe that the performance can be further improved if a more reasonable detector can be used for the TMGF feature.

6. CONCLUSION

This paper presented a method that combines the temporal modulation features on the gammatone auditory filterbank (TMGF) with an AE-based detector in the ASD challenges. With the proposed method, this paper aims to make up for the deficiency of the log-Mel-spectrogram (LMS) feature and provide the TMGF feature, including more timbral information related to timbral attributes such as sharpness and roughness. Experimental results in the DCASE2022 Challenge Task 2 showed that the proposed method could provide a better ability for domain generalization. For machine sounds recorded from the valve, results from both the source and target evaluation have significant improvements compared with the baseline method. Future work will focus on investigating the model architecture of the ASD system to extract more discriminative information from the proposed TMGF feature.

7. REFERENCES

[1] K. Dohi, K. Imoto, N. Harada, D. Niizumi, Y. Koizumi, T. Nishida, H. Purohit, T. Endo, M. Yamamoto, and Y. Kawaguchi, "Description and discussion on dcase 2022 challenge task 2: Unsupervised anomalous sound detection for machine condition monitoring applying domain generalization techniques," *arXiv preprint arXiv:2206.05876*, 2022.

[2] K. Suefusa, T. Nishida, H. Purohit, R. Tanabe, T. Endo, and Y. Kawaguchi, "Anomalous sound detection based on interpolation deep neural network," *Proc. IEEE-ICASSP*, pp. 271–275, 2020.

[3] S. Kapka, "Id-conditioned auto-encoder for unsupervised anomaly detection," *arXiv preprint arXiv:2007.05314*, 2020.

[4] P. Daniluk, M. Goździewski, S. Kapka, and M. Kośmider, "Ensemble of auto-encoder based and wavenet like systems for unsupervised anomaly detection," *Challenge on Detection and Classification of Acoustic Scenes and Events (DCASE 2020 Challenge), Tech. Rep*, 2020.

[5] T. Hayashi, T. Yoshimura, and Y. Adachi, "Conformer-based id-aware autoencoder for unsupervised anomalous sound detection," *DCASE2020 Challenge, Tech. Rep.*, 2020.

[6] K. Dohi, T. Endo, H. Purohit, R. Tanabe, and Y. Kawaguchi, "Flow-based self-supervised density estimation for anomalous sound detection," *Proc IEEE-ICASSP*, pp. 336–340, 2021.

[7] S. Davis and P. Mermelstein, "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences," *IEEE Transactions on Acoustics, Speech, and Signal processing*, vol. 28, no. 4, pp. 357–366, 1980.

[8] Y. Liu, J. Guan, Q. Zhu, and W. Wang, "Anomalous sound detection using spectral-temporal information fusion," *Proc. IEEE-ICASSP*, pp. 816–820, 2022.

[9] R. Sottek and K. Genuit, "Perception of roughness of time-variant sounds," *Journal of the Acoustical Society of America*, vol. 19, no. 1, p. 050195, 2013.

[10] Q.-H. Nguyen, K. Li, and M. Unoki, "Automatic mean opinion score estimation with temporal modulation features on gammatone filterbank for speech assessment," *Proc. INTERSPEECH*, 2022.

[11] H. Fastl and E. Zwicker, *Psychoacoustics - Facts and Models*. Berlin: Springer, 2007.

[12] B. C. Moore, *An introduction to the psychology of hearing*. Brill, 2012.

[13] A. Pearce, T. Brookes, and R. Mason, "Timbral attributes for sound effect library searching," in *Audio Engineering Society Conference: 2017 AES International Conference on Semantic Audio*, 2017.

[14] R. D. Patterson and J. Holdsworth, "A functional model of neural activity patterns and auditory images," *Advances in speech, hearing and language processing*, vol. 3, pp. 547–563, 1996.

[15] M. Unoki and M. Akagi, "A method of signal extraction from noisy signal based on auditory scene analysis," *Speech Communication*, vol. 27, no. 3-4, pp. 261–279, 1999.

[16] T. Dau, B. Kollmeier, and A. Kohlrausch, "Modeling auditory processing of amplitude modulation. i. detection and masking with narrow-band carriers," *Journal of the Acoustical Society of America*, vol. 102, no. 5, pp. 2892–2905, 1997.

[17] ——, "Modeling auditory processing of amplitude modulation. ii. spectral and temporal integration," *Journal of the Acoustical Society of America*, vol. 102, no. 5, pp. 2906–2919, 1997.

[18] K. Dohi, T. Nishida, H. Purohit, R. Tanabe, T. Endo, M. Yamamoto, Y. Nikaido, and Y. Kawaguchi, "Mimii dg: Sound dataset for malfunctioning industrial machine investigation and inspection for domain generalization task," *arXiv preprint arXiv:2205.13879*, 2022.

[19] N. Harada, D. Niizumi, D. Takeuchi, Y. Ohishi, M. Yasuda, and S. Saito, "Toyadmos2: Another dataset of miniature-machine operating sounds for anomalous sound detection under domain shift conditions," *arXiv preprint arXiv:2106.02369*, 2021.

[20] Y. Koizumi, Y. Kawaguchi, K. Imoto, T. Nakamura, Y. Nikaido, R. Tanabe, H. Purohit, K. Suefusa, T. Endo, M. Yasuda, *et al.*, "Description and discussion on dcase2020 challenge task2: Unsupervised anomalous sound detection for machine condition monitoring," *arXiv preprint arXiv:2006.05822*, 2020.

[21] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[22] Y. Koizumi, S. Saito, H. Uematsu, Y. Kawachi, and N. Harada, "Unsupervised detection of anomalous sound based on deep learning and the neyman–pearson lemma," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 27, no. 1, pp. 212–224, 2018.

FEW-SHOT BIOACOUSTIC EVENT DETECTION: ENHANCED CLASSIFIERS FOR PROTOTYPICAL NETWORKS

Ren Li[1], Jinhua Liang[1], Huy Phan[1,2]

[1] School of Electronic Engineering and Computer Science, Queen Mary University of London, United Kingdom
[2] The Alan Turing Institute, United Kingdom
{ml20986, jinhua.liang, h.phan}@qmul.ac.uk

ABSTRACT

Few-shot learning has emerged as a novel approach to bioacoustic event detection since it is useful when training data is insufficient, and the cost of labelling data is high. In this paper, we explore the Prototypical Networks for developing a few-shot learning system to detect mammal and bird sounds from audio recordings. To enhance the deep networks, we use a ResNet-18 variant as the classifier, which can learn the embedding mapping better with stronger architecture. Another method is proposed to focus on domain shift problem during learning the embedding by taking advantage of autoencoders to learn the low-dimensional representations of input data. A reconstruction loss is added to the training loss to perform regularization. We also utilize various data augmentation techniques to boost the performance. Our proposed systems are evaluated on the validation set of DCASE 2022 task 5 and improve the F1-score from 29.59% to 47.88%.

Index Terms— Few-shot learning, sound event detection, Prototypical Networks, embedding space

1. INTRODUCTION

Bioacoustic event detection is the task of recognizing biological sound events present in a set of audio recordings and predicting their time boundaries [1]. This technology is now benefitting from the power of deep learning and becomes an effective way to gain information on the activities of animals that reflects human's impact on the environment [2]. Traditionally, researchers have conducted the work through manually labelling on huge datasets, which is consuming both in time and resources [2]. In addition, collecting labelled data in some certain animal sounds can be challenging, and the scarcity of supervised data can lead to poor generalization and overfitting problem [1].

To address the data scarcity and reduce the cost of labelling data, few-shot learning has been proposed; this approach learns a classifier that can recognize new classes with a limited amount of labelled data [3]. One applicable advantage of few-shot learning is its ability to gain experience from prior similar tasks, so few-shot learning can be characterized as a kind of meta-learning [4]. A meta-learning algorithm gains experience over a set of learning "episodes" and uses this experience to improve its future performance for a new task [4]. For *N*-way-*K*-shot classification, each episode includes *N* classes with *K* examples. For the DCASE 2022 task 5, the first *K*=5 events are used for the class of interest for each test file to detect all the events of this class in the rest of the recording [1].

In recent years, an increasing number of meta-learning approaches for few-shot learning have been proposed and applied in many domains, such as sound event detection, image classification and text classification [4]. Among them, the Prototypical Network (ProtoNet) proposed by Snell *et al.* [3] is simple in principle but effective in practice. The ProtoNet transforms the input into an embedding space where the embeddings are simply clustered to the nearest "prototype" [3]. Therefore, it is desirable for the deep networks to produce adequate embedded features and calculate a useful prototype for each class.

In this work, we propose two enhanced methods to build a stronger ProtoNet. The first method uses a ResNet-18 variant as the embedding features extractor, which is capable of learning and extracting more advanced features with deeper and wider residual networks. The second method merges the ideas from autoencoders and ProtoNet to learn low-dimensional representations and to preserve the information contained in original low-level features. We also apply various spectrogram augmentation techniques to increase the amount of training data for model generalization. Our proposed systems are evaluated on the validation set of DCASE 2022 task 5 and achieve the best F1-score of 47.88%.

2. RELATED WORK

The use of convolutional neural layers allows feature extractor to extract complex features that express the raw data in much more detail and learn representations more efficiently. However, as the layers get deeper, the learned features can deteriorate due to vanishing gradient, leading to performance deterioration [5]. For this reason, residual networks (ResNets) were proposed and widely applied in deep learning tasks [5]. Sharma *et al.* [6] used a pre-trained ResNet-50 model for bird song classification, producing an accuracy of 97.1%, which was far superior to that of the VGG16 model. Soumya *et al.* [7] improved the ProtoNet using a customized ResNet as the feature embedding network for facial emotion recognition and proved its capability of extracting minute details.

In addition, most few-shot learning methods can suffer from domain shift problems during learning the embedding [8]. Since the embedding is only learned from the seen classes, when performing testing with the unseen classes, the embedding features are likely to be shifted due to the bias of the seen classes used for training [8]. Sometimes it can make the query data points far away from the correct corresponding unseen class prototypes, thus affecting the accuracy of the *k*-NN search. An effective Semantic Autoencoder (SAE) [8] is proposed to solve this problem by adding a reconstruction constraint to learn the low-dimensional representation. Moreover, Liu *et al.* [9] improved the SAE using graph structure and another L2-norm constraint, which preserves

the intrinsic data structure and has more discriminating power. Inspired by the ideas of above works, we enhance the ProtoNet for the few-shot bioacoustic event detection.

3. METHOD

This section introduces our methods developed upon the baseline system, including model design and data augmentation. First, we present a modified version of ResNet-18 used as a feature embedding network. After that, we describe how to combine autoencoders to perform data reconstruction from embeddings for better generalization to new unseen classes. Finally, we describe spectrogram augmentation techniques to boost the system's performance.

3.1. ResNet-based prototypical network

For the task of detecting bird and mammal sounds, it is important for the embedding module to extract adequate features since the sound samples are often too short and imperceptible to detect and distinguish them. However, the embedding module of the original ProtoNet only consists of four Conv blocks. If we use the original ProtoNet with multiple Conv blocks to learn the sound's features, it is prone to encounter gradient vanishing problems, which reduces the quality of the embeddings.

We thus choose residual networks as the embedding encoder, which can avoid the vanishing gradients thanks to skip connections. The skip connections add the output from a preceding layer to a later layer, allowing information to get fast-forwarded and go deeper with less deterioration [6]. Another salient feature of ResNets is the use of batch normalization (BN) to normalize the input of the activation function of the previous layer, which helps mitigate the covariate shift problem [6].

Our implementation is based on the ResNet-18. We modify the network to obtain a less deep model which only has 3 residual blocks to fit the size of the features. Each residual block contains 3 convolution layers using a kernel size of 3 × 3, followed by a batch normalization and a Leaky ReLU activation. Importantly, a shortcut with a 1 × 1 convolutional layer is added over the 3 layers. The architecture of our residual network is shown in Table 1, while Table 2 shows the architecture of the original embedding module, which provides a comparison.

Table 1. Architecture of the presented residual network

Encoder		Residual Block	
Layers	Channels	Layers	Kernel
Conv2D+ BN +ReLU	16	Conv2D+ BN +ReLU	3 × 3
Residual Block	64	Conv2D+ BN +ReLU	3 × 3
Residual Block	128	Conv2D+ BN	3 × 3
Residual Block	64	Shortcut: Conv 2D+BN	1 × 1
Adaptive AvgPooling+SoftMax	-	ReLU+MaxPool-ing+Dropout	1 × 1

Table 2. Architecture of original encoder and Conv block

Encoder		Conv Block	
Layers	Channels	Layers	Kernel Size
Conv Block	64	Conv2D	3 × 3
Conv Block	64	BatchNorm	-
Conv Block	64	ReLU	-
Conv Block	64	Max pool	2 × 2

3.2. Combination of Autoencoder and ProtoNet

To overcome the domain shift problems described in Section 2, we enhance the network based on the encoder-decoder paradigm. The encoder compresses the spectrograms of input data into an embedding space while the decoder reconstructs the expected original input features from the embedding space [10]. The output of the decoder is then compared with the original input features. This additional reconstruction task imposes a new constraint in learning the input features that guarantees the embedding features preserve more distinctive information contained in the original input features [8]. Therefore, it is effective in mitigating the domain shift problem. Although the appearance of features changes from seen classes to unseen classes, the demand for a more truthful reconstruction of the input features is unchanged; thus, the embedding function is generalizable across seen and unseen domains [8].

Inspired by AutoProtoNet proposed by Sandoval-Segura *et al.*[11], we use the 4 original sequential convolution blocks for the encoder and 4 sequential transpose convolution blocks for the decoder. The transpose convolution blocks are utilized to reproduce the high-dimensional low-level features by deconvolutional operations. The detail of these blocks is displayed in Table 2. Each convolutional block consists of a Conv2D layer, a Batch Normalization, a ReLU activation and followed by a Max Pooling layer with pool-size of 2×2. Each transpose convolutional block is made up of a transpose layer, a Batch Normalization layer, and a Conv2D layer and followed by a ReLU activation.

Table 2. Components of Conv Block and Transpose Conv Block

Conv Block		Transpose Conv Block	
Layers	Kernel Size	Layers	Kernel Size
Conv2D	3 × 3	Conv2D Transpose	2 × 2, stride 2
BatchNorm	-	BatchNorm	-
ReLU	-	Conv2D	3 × 3
Max pool	2 × 2	ReLU	-

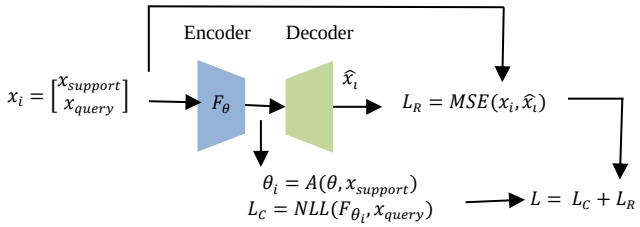


Figure 1: Overview of the forward pass through the autoencoder model

The training procedure of the autoencoder model is based upon the original training framework presented by Snell *et al* [3]. The main improvement is that we tailor the training loop with a reconstruction loss to regularize the embedding features and purpose it to preserve more useful details. The overview of the forward pass through the autoencoder is shown in Figure 1.

In the new training framework, the support data $x_{support}$ and query data x_{query} are randomly sampled from the current episode in form of 5 classes with 5 examples. They are then passed through the encoder and decoder to produce a reconstruction set

$\hat{x}_i$. This reconstruction set is then compared with the original input features x_i using mean square error (MSE) loss [11], defined as:

$$MSE = (x_i - \hat{x}_i)^2. \quad (1)$$

The finetuning algorithm A computes a set of prototypes p_k for each class k by computing the class-wise mean of embedded support examples and updates the model's parameters [3], and both are contained in θ_i. Eq. (2) defines the computation of prototypes:

$$p_k = \frac{1}{|S_k|} \sum_{x \in S_k} F_\theta(x), \quad (2)$$

where the S_k denotes the set of support samples for class k and x denotes the embeddings of class k.

Given the Euclidean distance function d and a set of query sound samples, the ProtoNet produces a probability distribution over classes for a query sample x belonging to true class k by Eq. (3) [3]. Then the training proceeds by minimizing the negative log-likelihood (NLL) of the true class k by Eq. (4). Finally, the classification loss L_C and the reconstruction loss L_R are summed to jointly optimize the training.

$$p_\theta(y = k|x) = \frac{\exp(-d(F_\theta(x), p_k))}{\sum_{k'} \exp(-d(F_\theta(x), p_{k'}))} \quad (3)$$

$$L(\theta) = -log\ p_\theta(y = k|x). \quad (4)$$

3.3. Data augmentation

In order to increase the diversity of data and the generalization ability of the model, we use *SpecAugment* [12] as the data augmentation technique. It essentially consists of three transformations: time warping, frequency masking, and time masking. Specifically, they modify a spectrogram by warping it in the time direction with a distance factor, masking blocks of consecutive frequency channels, and masking blocks of time steps, respectively [12]. In our case, we warp the feature to the left by 0.5 s and mask one block of one frequency mel bin and one block of 10 time steps. We choose these values according to the size of the time-frequency representation, which is appropriate to produce the diversity of the training data. If the values are too large or small, the augmented features could be very different from the originals or not changing enough; thus, the model is unable to achieve improved performance.

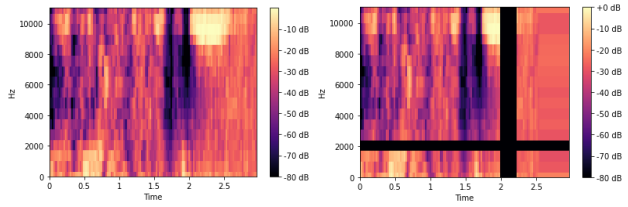


Figure 2: Example of spectrogram augmentation: orginial spectrogram (left) and augmented spectrogram with SpecAugment (right).

4. EXPERIMENTS

4.1. Dataset

The DCASE 2022 challenge provides a development set which is predefined as a training set and validation set. They were acquired from multiple bioacoustic sources, including sounds of worldwide birds, spotted hyenas, jackdaws, meerkats, and wetlands birds [1]. As a result, the sounds can be long or very short across the subsets; the sampling rate of each audio varies from 6 kHz to 44 kHz [1]. The training set consists of 174 audio recordings, 47 classes and 14,229 event instances. In addition, multi-class annotations are provided for the training set with positive, negative, and unknown; we only extracted and made use of the positive event instances for training. The validation set consists of 18 audio recordings, 5 classes and 1,077 positive event instances [1].

4.2. Data pre-processing

Mel-spectrogram. All audio files in both the training set and validation set were first resampled to a sampling rate of 22,050 Hz. The audio files were then transformed to Mel-spectrograms with 128 Mel bins using an FFT size of 1024 samples and a hop size of 256 samples. The *librosa* library was employed for this purpose. Afterwards, spectrogram images of size $F \times T$ where F=17 by T=128 were used as inputs.

PCEN. PCEN has been proposed to normalize a time-frequency representation by performing automatic gain control, followed by nonlinear compression [13]. Former research used PCEN to mitigate the effects of background noise, demonstrating its effectiveness as a preprocessing step prior to convolutional methods in sound event detection [13]. Bioacoustic data recorded in the wild often have multiple sound sources and uncleaned background. Therefore, we utilized PCEN to reduce noise presented in the Mel-spectrograms and improve robustness to channel distortion.

4.3. Training

Prototypical networks adopt an episodic training procedure where in each episode, a mini batch is randomly sampled from the training data [4]. A subset of mini batch was used as the support set and the remaining is used as query set. The models were trained with 2,000 episodes and 5 classes in each minibatch with the Adam optimizer and the learning rate of 0.001. Euclidean distance was selected as the metric that measures the distance between query samples to a prototype.

4.4. Post-processing

The preliminary experiments confirms that the task of detecting bioacoustics events from nature is challenging; most classification methods can produce a large number of false-positive predictions, substantially reducing the model's F1-score [14]. Therefore, we applied post-processing to the outputs to remove possible false positives. Specifically, we removed the predictions that were shorter than 20% of the average duration calculated by the first 5 shots for each audio file. It was because participants were expected to treat the task as a 5-shot setting.

Table 3. Comparison of models using different classifier and feature. The best results are highlighted in **boldface**.

Model components			Validation set scores (%)			Subset F1-score (%)		
Exp No.	Classifer	Feature	F1-score	Precision	Recall	HB	ME	PB
1	CNN (Baseline)	PCEN	29.59	36.34	24.96	/	/	/
2	ResNet	PCEN	45.64	48.34	43.22	50.00	57.14	26.18
3	Autoencoder	PCEN	37.94	38.95	36.97	44.53	52.05	25.68
4	CNN	PCEN+Augment	37.16	42.09	33.26	38.86	**72.01**	15.33
5	ResNet	PCEN+Augment	**47.88**	**52.11**	**44.30**	**53.45**	**50.98**	**17.65**
6	Autoencoder	PCEN+Augment	**47.61**	**50.18**	**45.34**	**52.68**	**53.10**	**22.44**

5. RESULTS AND DISCUSSIONS

We conducted several ablation experiments on the validation set to verify the effectiveness of the components and tricks in our proposed models. To further investigate the capabilities of our models, we computed the F1-score for three difference subsets. The experiments results are shown in Table 3. The brief information about each subset is as follows: the HB subset records the mosquito's events that are very long with low noise; the ME subset contains the sounds of meerkats that are short with low noise; the PB subset records the bird flight calls that are very short and unclear with high noise.

5.1. Effects of using ResNets

From the results of Experiments 1 and 2, it can be seen that the ResNet model outperforms the baseline CNN and achieves a noticeable improvement of over 15%. It is primarily due to the ResNet's deep architecture, which has many more parameters to capture the features better, allowing the learned features to fit the input data better. In addition, the residual networks make use of skip connections, enabling the model to carry gradients to a very deep layer. It also allows the model optimally tuning the number of the layers during training, so that the model parameters can be updated more optimally. However, since it adopts a complex networks architecture, the computation and memory cost increase intensively. To compare the model complexity, we measure the number of trainable parameters for different models as shown in Table 4. Compared to CNN with 112k parameters, the number of parameters for ResNet has grown significantly to 724k, and the number of parameters for the autoencoder is roughly twice that of the CNN.

Table 4. The number of parameters for different models

Model	Parameters
CNN (Baseline)	111,936
ResNet	724,096
Autoencoder	272,717

5.2. Effects of adding reconstruction loss

As shown by the results of Experiments 1, 2 and 3, the autoencoder model also improves the performance but with a slightly lower gain compared to that of the ResNet model. By adding a reconstruction loss, it is likely that the autoencoder model can learn the high-level low-dimensional representation and preserve more useful details, resulting in a better generalization to unseen classes for few-shot learning task. However, the reconstruction loss is served as a constraint during learning, so its influence is limited. Making an embedding classifier to learn the representation of the input data in a fundamentally different way is more meaningful and challenging. That could be the reason why the ResNet model outperforms the autoencoder in this case.

5.3. Effects of data augmentation

Applying *SpecAugment* has been the popular choice for sound event detection. When training data is unbalanced and insufficient, it has shown to be effective. Overall, this technique also workes well for our systems. Despite being augmented by *SpecAugment*, the performance of detecting the very short bird sounds in PB subset decreases. The reason could be that since the masking and warping were randomly applied from a uniform distribution over the value of factors, the blocks and warping steps could be too excessive in some cases. This could lead to some useful information in the minor sounds being masked, or even the augmented features became quite different from the originals, thus preventing the networks from learning. It gets even worse under high-noise conditions. Furthermore, the model trained on CNN and augmented data was observed to achieve much higher results for the ME subset than other approaches. This is because ResNet and autoencoder have many more parameters to optimize, likely leading to overfitting the training set. In contrast, the CNN with a simple architecture is just capable of generalizing better to ME data.

6. CONCLUSION

In this paper, we present two solutions to enhance Prototypical Networks for the task of bioacoustics sound event. We show a list of ablation studies and discussed the effects of each component or trick used in our systems. Overall, using ResNets and autoencoder (or construction loss) contribute to learn a more adequate embedding space and boost the model's performance. Joining with data augmentation techniques, our enhanced models achieve the best F1-score of 47.88%, which improves over the baseline by a large margin.

Our studies also imply that it is challenging to detect the very short and unclear bioacoustics sound events. This can be an important subject to be explored in the future work. Recent research show that *model adaptation* is an effective solution to improve the general robustness of SED method [15].

7. REFERENCES

[1] https://dcase.community/challenge2022/task-few-shot-bioacoustic-event-detection

[2] C. Chalmers, P. Fergus, S. Wich and S. N. Longmore, "Modelling Animal Biodiversity Using Acoustic

Monitoring and Deep Learning," in *2021 International Joint Conference on Neural Networks (IJCNN)*, 2021, pp. 1-7.

[3] J. Snell, K. Swersky, R. Zemel, "Prototypical networks for few-shot learning," Advances in neural information processing systems. 2017, vol. 30.

[4] Z. Zhan, J. Zhou, and B. Xu, "Fabric defect classification using prototypical network of few-shot learning algorithm," Computers in Industry. 2022, vol. 138, p.103628.

[5] C. S. Wickramasinghe, D. L. Marino and M. Manic, "ResNet Autoencoders for Unsupervised Feature Learning From High-Dimensional Data: Deep Models Resistant to Performance Degradation," *IEEE Access*, 2021 vol. 9, pp. 40511-40520.

[6] N. Sharma, A. Vijayeendra, V. Gopakumar, P. Patni and A. Bhat, "Automatic Identification of Bird Species using Audio/Video Processing," in *2022 International Conference for Advancement in Technology (ICONAT)*, 2022, pp. 1-6.

[7] K. Soumya and S. Palaniswamy, "Emotion Recognition from Partially Occluded Facial Images using Prototypical Networks," in *2020 2nd International Conference on Innovative Mechanisms for Industry Applications (ICIMIA)*, 2020, pp. 491-497.

[8] E. Kodirov, T. Xiang, and S. Gong, "Semantic autoencoder for zero-shot learning," *in Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 3174-3183.

[9] Y. Liu, D.Y. Xie, Q. Gao, J. Han, S. Wang, and X. Gao, Graph and autoencoder based feature extraction for zero-shot learning. In IJCAI, 2019, vol.1(2), pp.6.

[10] S. Hong and S. -K. Song, "Kick: Shift-N-Overlap Cascades of Transposed Convolutional Layer for Better Autoencoding Reconstruction on Remote Sensing Imagery," *IEEE Access*, 2020, vol. 8, pp. 107244-107259.

[11] P. Sandoval-Segura, W. Lawson. AutoProtoNet: Interpretability for Prototypical Networks. arXiv preprint arXiv:2204.00929. 2022 Apr 2.

[12] D.S. Park, W. Chan, Y. Zhang, C.C. Chiu, B. Zoph, E.D. Cubuk, and Q.V. Le. Specaugment: A simple data augmentation method for automatic speech recognition. 2019, arXiv preprint arXiv:1904.08779.

[13] C. Ick and B. McFee, "Sound Event Detection in Urban Audio with Single and Multi-Rate Pcen," *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 880-884.

[14] V. Morfi, I. Nolasco, V. Lostanlen, S. Singh, A. Strandburg-Peshkin, L.F. Gill, H. Pamula, D. Benvent, and D. Stowell. Few-Shot Bioacoustic Event Detection: A New Task at the DCASE 2021 Challenge. In DCASE, 2021, November, pp. 145-149.

[15] A. Mesaros, T. Heittola, T. Virtanen and M. D. Plumbley, "Sound Event Detection: A tutorial," in *IEEE Signal Processing Magazine*, 2021, vol. 38, pp. 67-83.

A HYBRID SYSTEM OF SOUND EVENT DETECTION TRANSFORMER AND FRAME-WISE MODEL FOR DCASE 2022 TASK 4

Yiming Li[1,2], *Zhifang Guo*[1,2], *Zhirong Ye*[1,2], *Xiangdong Wang*[1,†], *Hong Liu*[1], *Yueliang Qian*[1], *Rui Tao*[3], *Long Yan*[3], *Kazushige Ouchi*[3]

[1] Beijing Key Laboratory of Mobile Computing and Pervasive Device, Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China, eamon.y.li@gmail.com, {guozhifang21s, yezhirong19s, xdwang, hliu, ylqian}@ict.ac.cn
[2] University of Chinese Academy of Sciences, Beijing, China
[3] Toshiba China R&D Center, Beijing, China, {taorui, yanlong}@toshiba.com.cn, kazushige.ouchi@toshiba.co.jp

ABSTRACT

In this paper, we describe in detail our system for DCASE 2022 Task4. The system combines two considerably different models: an end-to-end Sound Event Detection Transformer (SEDT) and a frame-wise model, Metric Learning and Focal Loss CNN (MLFL-CNN). The former is an event-wise model which learns event-level representations and predicts sound event categories and boundaries directly, while the latter is based on the widely-adopted frame-classification scheme, under which each frame is classified into event categories and event boundaries are obtained by post-processing such as thresholding and smoothing. For SEDT, self-supervised pre-training using unlabeled data is applied, and semi-supervised learning is adopted by using an online teacher, which is updated from the student model using the Exponential Moving Average (EMA) strategy and generates reliable pseudo labels for weakly-labeled and unlabeled data. For the frame-wise model, the ICT-TOSHIBA system of DCASE 2021 Task 4 is used. Experimental results show that the hybrid system considerably outperforms either individual model, and achieves psds1 of 0.420 and psds2 of 0.783 on the validation set without external data. The code is available at *https://github.com/965694547/Hybrid-system-of-frame-wise-model-and-SEDT*.

Index Terms— Sound Event Detection Transformer, Online Pseudo-labelling, Hybrid System

1. INTRODUCTION

Sound Event Detection (SED) aims at identifying the category of foreground sound events as well as their corresponding onset and offset timestamps. Task4 of the DCASE challenge has been focusing on weakly supervised SED for several years. The DCASE 2022 Task4 [1] is a follow up of last year's challenge [2]. This year, in addition to exploring a heterogeneous development dataset containing unlabeled data, synthetic data and weakly labeled data, participants are allowed to incorporate external dataset or pre-trained embeddings. As last year, the SED system will be evaluated by Polyphonic Sound Detection Score (PSDS) [3] under two different real-life settings.

For weakly supervised SED, most existing works follow the Multiple Instance Learning (MIL) framework, and formulate SED as a seq2seq classification task. They usually design Convolutional Neural Networks (CNNs) or Convolutional Recurrent Neural Networks (CRNNs) to obtain frame-level classification probability and then apply pooling mechanism to aggregate frame-level predictions to event-level results. However, such methods do not take sound events as a whole, which may ignore some global information, such as the correlation between frames or event duration. Recently, an event-wise model, namely SEDT, is proposed to handle such problems [4]. It models SED as a set prediction problem, which directly maps audio spectrogram to a set of candidate events, thus freeing SED models from trivial post-processing, namely frame-level thresholding or median filtering. Empirical study has shown that SEDT can achieve competitive performance compared with its frame-wise counterparts [4]. Moreover, we find that the two models can supplement each other, as they solve the SED task in different ways. Therefore, combining them together may be an intuitive approach to reach promising SED performance.

In this paper, we describe our system participating in DCASE 2022 Task 4. It is a combination of SEDT and frame-wise CNN model. For SEDT, specially-designed training formulas, including supervised learning, self-supervised learning and semi-supervised learning, are studied to help it learn from the heterogeneous development dataset. For frame-wise CNN model, metric learning is applied to narrow the domain gap between real and synthetic data, mean-teacher framework is implemented to provide supervision for unlabeled data and a tag-conditioned CNN model is used to generate final predictions based on audio tags. After obtaining each well-trained model, we explore the fusion strategy and post-processing methods of the ensemble model. By using the methods above, the hybrid system achieves competitive results on the validation dataset.

2. SEMI-SUPERVISED SEDT

2.1. Sound Event Detection Transformer

An overview of SEDT is shown in Fig. 1. It represents each sound event as $y_i = (c_i, b_i)$, where c_i is the event category and $b_i = (m_i, l_i)$ denotes the event temporal boundary containing normalized event center m_i and duration l_i, and directly seeks a mapping between input features and ground-truth events. Given the input spectrogram, the backbone CNN is adopted to extract its feature map, which is then added with one-dimensional positional encoding and fed into transformer encoder for further feature processing. The transformer decoder takes $N + 1$ learnable embeddings (N event queries and 1 audio query) as input event query, where each of them

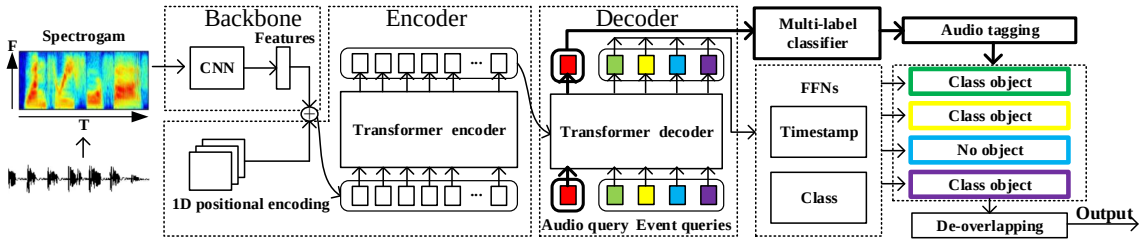


Figure 1: Overview of Sound Event Detection Transformer

gathers information of a potential event from the encoder output feature via encoder-decoder cross-attention mechanism to generate event-level representations, while audio query gathers the whole audio information to generate clip-level representations. Finally, Feed Forward Networks (FFNs) are utilized to transform the event-level representations and clip-level representations from the decoder into event detection and audio tagging results, which are then fused together to get the candidate detection results. De-overlapping is implemented on overlapped candidate events of the same category. Specifically, it only reserves the events with the highest class probability. More details can be found in [4].

2.2. Supervised learning for SEDT

SEDT incorporates event-level loss and clip-level loss to optimize its event detection and audio tagging performance. For strongly-labeled data, both loss terms will be involved during the SEDT model training, while for weakly-labeled data, the event-level loss will be excluded since the strong annotations are not available.

Event-level loss. SEDT adopts a label assignment scheme before computing event-level loss: it tries to find a matching $\hat{\sigma}_i$ between each event prediction $\hat{y}_i$ and its corresponding ground-truth annotation y_i through Hungarian algorithm, which is efficient for above bipartite graph matching problem. To equip SEDT with sound event classification and localization ability, the loss for SEDT supervised training is formulated as the weighted linear combination of localization loss $\mathcal{L}_{\text{loc}}$ and classification loss $\mathcal{L}_{\text{cls}}$. For each event prediction, the two loss functions are calculated as:

$$\mathcal{L}_{\text{loc}} = \sum_{i}^{N} \left(\lambda_{\text{IOU}} \mathcal{L}_{\text{IOU}} \left(b_i, \hat{b}_{\hat{\sigma}(i)} \right) + \lambda_{\text{L1}} \left\| b_i - \hat{b}_{\hat{\sigma}(i)} \right\|_1 \right) \quad (1)$$

$$\mathcal{L}_{\text{cls}} = \frac{1}{N} \sum_{i=1}^{N} -\log \hat{p}_{\hat{\sigma}(i)} \left(c_i \right) \quad (2)$$

where λ_{IOU} and λ_{L1} are weights for Intersection Over Union (IOU) loss [5] and L1 loss.

Clip-level loss. The audio tagging loss is defined as the binary cross-entropy between the clip-level class label l_{tag} and predicted audio tagging y_{tag}:

$$\mathcal{L}_{\text{at}} = \text{BCE} \left(l_{\text{tag}}, y_{\text{tag}} \right) \quad (3)$$

2.3. Self-supervised learning for SEDT

To better use the unlabeled or external datasets, such as AudioSet and SINS, we adopt a self-supervised learning method to pre-train SEDT on unlabeled data, which is named as Self-supervised Pre-training SEDT (SP-SEDT). Specifically, we randomly crop spectrogram along the time axis to obtain several patches, and then pre-train the model to predict corresponding locations of the patches. To preserve the category information in SP-SEDT, classification loss and feature reconstruction loss are also adopted as sub-objective terms. By means of such pre-text task, we hope that SEDT can localize sound event and maintain most category-related features at the same time. More details can be found in [6].

2.4. Semi-supervised learning for SEDT

Pseudo-labelling [7] is one of the mainstream approaches of semi-supervised learning. It requires a well-trained model to generate pseudo labels on unlabeled data, so that in the next stage, the converged model can be re-optimized on both labelled data and unlabeled data jointly. Based on that, we propose an improved pseudo-labelling method for the Semi-Supervised learning of SEDT (SS-SEDT). SS-SEDT splits the training process into two stages: the burn-in stage and the teacher-guided stage. In the burn-in stage, SEDT is simply trained on the labeled dataset to initialize the model. At the beginning of the teacher-guided stage, the initialized model is copied into two models (a student model and a teacher model), and then the teacher model generates pseudo labels on unlabeled data so that the student model can gain knowledge from both labeled data and unlabeled data. To guarantee the quality of the pseudo labels, we revisit the following off-the-shelf techniques, and apply them in the teacher-guided process. The detailed training process of teacher-guided stage is shown in Algorithm 1.

- **EMA**: Unlike previous methods supervised by offline pseudo labels, we resort to a progressing teacher model to generate pseudo labels. The teacher model is updated from the student model through EMA and thus can be viewed as implicit ensemble models and provide more reliable guidance. Notice that although the usage of EMA is similar to that in the mean-teacher framework, the proposed method is different since pseudo labels involved are hard ones and no consistency loss is adopted.
- **Asymmetric augmentation**: Asymmetric augmentation has been introduced into semi-supervised image recognition [8] and SED [9]. Inspired by that, we adopt similar idea in the teacher-guided stage, during which weakly-augmented (frequency mask and frequency shift) spectrograms are fed into the teacher model to get pseudo labels and the student model make predictions on the strongly augmented (frequency mask, frequency shift, time mask and gaussian noise) version of the same data batch.
- **Mixup** [10]: We mix labeled data with ground-truth and unlabeled data with pseudo annotations together, which is supposed to improve the model robustness to pseudo annotation noise and alleviate the overfitting problem in model training.
- **Focal loss** [11]: Focal loss is adopted to handle the unbalanced event categories in SED, without which the model may be overwhelmed by easily classified samples and produce biased out-

Algorithm 1: Pseudocode for teacher-guided stage

Require: $\mathcal{B}_L$ = labeled batch, $\mathcal{B}_U$ = unlabeled batch
Require: $S_\theta(x)$ = student model, $T_{\theta'}(x)$ = teacher model
Require: $A_w(x)$ = weak augmentation function
Require: $A_s(x)$ = strong augmentation function
Require: α = learning rate, γ = EMA ratio
Require: $\mathcal{L}$ = loss function
Ensure : θ, θ'

for $i \to 1$ **to** *max_epochs* **do**
  foreach $\mathcal{B}_L \cup \mathcal{B}_U \in \mathcal{B}$ **do**
    $\mathcal{J}_{\text{sup}} \leftarrow \frac{1}{|\mathcal{B}_L|} \sum_{(x_i, y_i) \in \mathcal{B}_L} \mathcal{L}\left(S_\theta\left(A_w(x_i)\right), y_i\right)$;
    foreach $x_i \in \mathcal{B}_U$ **do** $y_i \leftarrow T_{\theta'}(A_w(x_i))$;
    $\hat{\mathcal{B}} \leftarrow Mixup(\mathcal{B}_L, \mathcal{B}_U)$;
    $\mathcal{J}_{\text{unsup}} \leftarrow \frac{1}{|\hat{\mathcal{B}}|} \sum_{(\hat{x}_i, \hat{y}_i) \in \hat{\mathcal{B}}} \mathcal{L}\left(S_\theta\left(A_s(\hat{x}_i)\right), \hat{y}_i\right)$;
    $\theta \leftarrow \theta - \alpha(\frac{\partial \mathcal{J}_{\text{sup}}}{\partial \theta} + \frac{\partial \mathcal{J}_{\text{unsup}}}{\partial \theta})$;
    $\theta' \leftarrow \gamma\theta' + (1-\gamma)\theta$;
  end
end

puts. It should be noted that focal loss is merely used in the teacher-guided stage, we believe such curriculum learning pattern may help our model learn from easy to difficult.

3. FRAME-WISE CNN MODEL

The pipeline of the frame-wise CNN model is illustrated in Fig. 2. At first, MLFL-CNN is preliminarily trained with weakly labeled data and strongly labeled synthetic data to acquire basic event detection and audio tagging ability. Then, it attaches pseudo strong labels to the weakly-labeled and unlabeled data, and the model is jointly trained with all these data in a self-training manner. Finally, the trained MLFL-CNN provides audio tags and strong pseudo labels for the weakly-labeled data and unlabeled data to train the tag-conditioned CNN [12], which gives the final SED results.

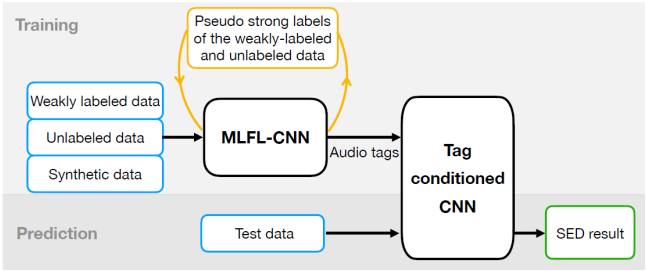


Figure 2: Overview of the frame-wise model

The MLFL-CNN model contains three branches. The first branch is the embedding-level attention pooling branch based on the MIL framework, which is the same with [13]. The second branch is the sound event detection branch which is introduced to exploit the strong labels of synthetic data and uses focal loss as its supervision. The third branch is the domain adaptation branch which uses metric learning by inter-frame distance contrastive loss, more details of which can be found in [14]. During training process, the MLFL-CNN adopts the mean-teacher architecture and pseudo-labelling framework simultaneously [15]. It combines clip-level loss (for weakly-labeled data), frame-level loss (for data with strong labels and pseudo strong labels), inter-frame distance contrastive loss (for real data and synthetic data), and consistency loss together. And the tag-conditioned CNN takes spectrograms and audio tags predicted by the MLFL-CNN as inputs, and uses the strong labels of synthetic data and pseudo strong labels of real data as ground-truth to train.

4. FUSION OF THE TWO MODELS

4.1. Preliminary: Class-specific PSDS

The essence of PSDS is to obtain a function $r(e)$ of effective TP rate (eTPR) changing with effective FP rate (eFPR), and calculate the integral of this function over $(0, e_{\max})$, where $e_{\max}$ represents the maximum value of eFPR value [3]. We notice that the original calculation of eTPR relies on two class-averaged indicators μ_{TP} and σ_{TP}. To decouple the eTPR according to the event category, we simply replace class-averaged indicators with class-dependent ones and finally redefine the PSDS value of given category as follow:

$$\mu_{\text{TP,c}} = r_{\text{TP,c}} \quad \sigma_{\text{TP,c}} = r_{\text{TP,c}} - \mu_{\text{TP,c}} \tag{4}$$

$$\text{eTPR}_c: \quad r_c(e) \triangleq \mu_{\text{TP,c}}(e) - \alpha_{ST} * \sigma_{\text{TP,c}}(e) \tag{5}$$

$$\text{PSDS}_c \triangleq \frac{1}{e_{\max}} \int_0^{e_{\max}} r_c(e) de \tag{6}$$

where PSDS_c, eTPR_c, $\mu_{\text{TP,c}}$ and $\sigma_{\text{TP,c}}$ are corresponding class-wise indicators for specific event class c.

4.2. Model fusion method

The core of model fusion is to calculate the class-wise fusion coefficients of each model's prediction during the evaluation stage. Assume that there are N models $m_i (i = 1, 2, \ldots N)$, for each sound event class c, the PSDS of model m_i on c is denoted as $\text{PSDS}_{i,c}$. Then the fusion coefficient of model i on category c is defined as:

$$w_{i,c} = \frac{\text{PSDS}_{i,c}}{\sum_{i=1}^{N} \text{PSDS}_{i,c}} \tag{7}$$

Therefore, for specific event category c, the final fusion probability $\hat{p_c}$ is formulated as the weighted linear combination of each model's predicted probability $p_{i,c}$:

$$\hat{p_c} = \sum_{i=1}^{N} w_{i,c} * p_{i,c} \tag{8}$$

It is noteworthy that the above PSDS in Eq.(7) can be interpreted as PSDS1 or PSDS2 for this year's DCASE task4, so two different sets of parameters $w_{i,c}$ can be obtained on the development set and utilized to improve PSDS1 and PSDS2 respectively.

5. POST-PROCESSING

In order to reduce the noise in frame-level probability and make sound events continuous, it is necessary to perform a smoothing operation, such as mean filter or median filter, on the frame-level probability. Currently, median filtering with a fixed window length or with the average length of each event calculated on the development set is generally utilized [16]. In this paper, we perform median filtering and mean filtering (with larger window size) on frame-level probabilities in sequence, and propose a method to search for optimal class-wise window lengths on the development set.

Table 1: The PSDS on the validation set

System	Extra data	PSDS1	PSDS2
Baseline 1		0.336	0.536
Baseline 2	✓	0.351	0.552
System 1	✓	0.449	0.645
System 2	✓	0.115	0.816
System 3		0.420	0.618
System 4		0.099	0.783

Table 2: Ablation study on techniques in SS-SEDT

MU	FL	AA	EMA	PSDS1	PSDS2
	✓	✓	✓	0.372	0.570
✓		✓	✓	0.349	0.540
✓	✓		✓	0.369	0.566
✓	✓	✓		0.357	0.538
✓	✓	✓	✓	0.388	0.573

Specifically, for a given event class c, we enumerate window length wl_c from 1 to 500, and find the optimal length $wl_c{}^*$ to optimize PSDS1 and PSDS2 respectively:

$$wl_c{}^* = \arg\max_{wl_c} \frac{\mathrm{PSDS}_c}{\mathrm{PSDS}} \tag{9}$$

Finally, in the hybrid system, the event-level predictions of SEDT are firstly obtained in an end-to-end manner and then converted into frame-level probabilities, before being fused with frame-wise model and finally post-processed to get the ultimate results.

6. EXPERIMENT

6.1. Experiment Setup

For SEDT not using external data, we firstly pre-train it on unlabeled real subset (14412 clips), then simply train it on the weakly labeled training set (1578 clips) and synthetic 2019 subset (2045 clips) during burn-in stage, and finally use weakly labeled set, synthetic 2019 subset, synthetic 2021 subset (10000 clips), and unlabeled subset to conduct teacher-guided learning. For SEDT using external data, the two main differences compared to the above lie in 1) models are pre-trained on both unlabeled real subset and SINS subset (72894 clips), 2) an additional strongly labeled set (3470 clips) is further included in the teacher-guided stage. The detailed settings can be found in our repository [1].

For frame-wise model not using external data, the training set contains the weakly labeled training set, the unlabeled training set, and synthetic 2021 subset. While for systems using external data, we add the same strongly labeled set taken from AudioSet to the original strong labeled set. The detailed settings of training hyperparameters and configurations can be found in [17].

6.2. Results of Submitted Systems

Table 1 shows the performance of our submitted systems, all of which are fused models of ensemble frame-wise CNN models and ensemble SEDT. Among them, system 1 and 2 incorporate external data, while system 3 and 4 do not. Besides, model fusion and window tuning methods proposed in Section 4 and Section 5 are utilized in system 1, 3 to improve their PSDS1 and in system 2, 4 to improve their PSDS2 separately. As shown in Table 1, our hybrid systems outperform the official baseline considerably whatever the usage of external data. Moreover, our systems ranked 6th / 9th in the challenge respectively. While they are inferior to the winner models, our designed components are orthogonal to network architecture and data augmentation, which means that they may generalize to other models and bring about promising improvements.

Table 3: Ablation study on window tuning and model fusion

Id	Model	MF	WT	PSDS1	PSDS2
1	Single SEDT			0.415	0.582
2	Ensemble SEDT			0.431	0.607
3	Single frame			0.349	0.668
4	Ensemble frame			0.392	0.673
5	Hybrid system	✓		0.437	0.740
6	Hybrid system	✓	✓	0.449	0.816

6.3. Ablation Study

Techniques in SS-SEDT. To verify the effectiveness of techniques in SS-SEDT, we conduct ablation study using single SS-SEDT model without external data. Table 2 shows the results of models trained without specific technique, where MU, FL, AA denotes Mixup, Focal Loss, Asymmetric Augmentation mentioned in Section 2.4 respectively, and the model trained without AA means that the inputs of teacher and student model are both weakly augmented. It can be seen that all techniques can improve the performance of SS-SEDT and it can finally reach a PSDS1 of 0.388 and a PSDS2 of 0.573 while incorporating all techniques.

Window tuning and model fusion. To investigate the effects of window tuning and model fusion strategy, we conduct ablation study using SEDT and frame-wise model trained with external data. Table 3 compares the performance between models under different settings. In the above table, MF and WT denote Model Fusion and Window Tuning methods proposed in Section 4 and 5 respectively, and frame-wise model is abbreviated to "frame". Among all these models, model 2 and 4 are ensemble models of top 1-5 single models, while hybrid system represents the fused model of ensemble SEDT and ensemble frame-wise model. By comparing model 1, 2 with model 3, 4, it is obvious that SEDT can achieve higher PSDS1 while frame-wise model is better at PSDS2. Moreover, by comparing model 5 with model 2, 4, we can see that while SEDT and frame-wise model have their own edges, they can complement each other, since the hybrid system achieve further improvements compared to single ensemble models. By comparing model 6 with model 5, the effectiveness of window tuning can be validated, since model 6 provides the best PSDS1 (0.449) and PSDS2 (0.816).

7. CONCLUSIONS

In this paper, we developed a framework to fuse the detection results of the frame-wise model and event-wise model, which leads to an improved PSDS1 of 0.420 and PSDS2 of 0.783 on the validation set compared to individual ensemble models.

8. ACKNOWLEDGMENT

This work was partly supported by the National Natural Science Foundation of China (62276250) and the Beijing Natural Science Foundation (Z190020).

[1] https://github.com/Anaesthesiaye/sound_event_detection_transformer

9. REFERENCES

[1] https://dcase.community/challenge2022/task-sound-event-detection-in-domestic-environments.

[2] https://dcase.community/challenge2021/task-sound-event-detection-in-domestic-environments.

[3] Ç. Bilen, G. Ferroni, F. Tuveri, J. Azcarreta, and S. Krstulović, "A framework for the robust evaluation of sound event detection," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 61–65.

[4] Z. Ye, X. Wang, H. Liu, Y. Qian, R. Tao, L. Yan, and K. Ouchi, "Sound event detection transformer: An event-based end-to-end model for sound event detection," *arXiv preprint arXiv:2110.02011*, 2021.

[5] J. Yu, Y. Jiang, Z. Wang, Z. Cao, and T. Huang, "Unitbox: An advanced object detection network," in *Proceedings of the 24th ACM International Conference on Multimedia*, 2016, pp. 516–520.

[6] Z. Ye, X. Wang, H. Liu, Y. Qian, R. Tao, L. Yan, and K. Ouchi, "Sp-sedt: Self-supervised pre-training for sound event detection transformer," *arXiv preprint arXiv:2111.15222*, 2021.

[7] D.-H. Lee *et al.*, "Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks," in *Workshop on Challenges in Representation Learning, ICML*, vol. 3, no. 2, 2013, p. 896.

[8] K. Sohn, D. Berthelot, N. Carlini, Z. Zhang, H. Zhang, C. A. Raffel, E. D. Cubuk, A. Kurakin, and C.-L. Li, "Fixmatch: Simplifying semi-supervised learning with consistency and confidence," *Advances in Neural Information Processing Systems*, vol. 33, pp. 596–608, 2020.

[9] T. K. Chan and C. S. Chin, "Multi-branch convolutional macaron net for sound event detection," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 2972–2985, 2021.

[10] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," in *International Conference on Learning Representations*, 2018.

[11] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2980–2988.

[12] J. Ebbers and R. Haeb-Umbach, "Forward-backward convolutional recurrent neural networks and tag-conditioned convolutional neural networks for weakly labeled semi-supervised sound event detection," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2020 Workshop*, 2020, pp. 41–45.

[13] L. Lin, X. Wang, H. Liu, and Y. Qian, "Specialized decision surface and disentangled feature for weakly-supervised polyphonic sound event detection," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 1466–1478, 2020.

[14] Y. Huang, L. Lin, X. Wang, H. Liu, Y. Qian, M. Liu, and K. Ouchi, "Learning generic feature representation with synthetic data for weakly-supervised sound event detection by inter-frame distance loss," *arXiv preprint arXiv:2011.00695*, 2020.

[15] T. Rui, Y. Long, O. Kazushige, and X. Wang, "Couple learning for semi-supervised sound event detection," in *Proc. Interspeech 2022*, 2022, pp. 2398–2402.

[16] L. Lin, X. Wang, H. Liu, and Y. Qian, "Guided learning convolution system for dcase 2019 task 4," in *Workshop on Detection and Classification of Acoustic Scenes and Events 2019*, 2019, p. 134.

[17] G. Tian, Y. Huang, Z. Ye, S. Ma, X. Wang, H. Liu, Y. Qian, R. Tao, L. Yan, K. Ouchi, and R. Ebbers, Janek Haeb-Umbach, "Sound event detection using metric learning and focal loss for dcase 2021 task 4," DCASE2021 Challenge, Tech. Rep., June 2021.

LEVERAGING LABEL HIERARCHIES FOR FEW-SHOT EVERYDAY SOUND RECOGNITION

Jinhua Liang, Huy Phan, Emmanouil Benetos

Centre for Digital Music, Queen Mary University of London, United Kingdom
{jinhua.liang, h.phan, emmanouil.benetos}@qmul.ac.uk

ABSTRACT

Everyday sounds cover a considerable range of sound categories in our daily life, yet for certain sound categories it is hard to collect sufficient data. Although existing works have applied few-shot learning paradigms to sound recognition successfully, most of them have not exploited the relationship between labels in audio taxonomies. This work adopts a hierarchical prototypical network to leverage the knowledge rooted in audio taxonomies. Specifically, a VGG-like convolutional neural network is used to extract acoustic features. Prototypical nodes are then calculated in each level of the tree structure. A multi-level loss is obtained by multiplying a weight decay with multiple losses. Experimental results demonstrate our hierarchical prototypical networks not only outperform prototypical networks with no hierarchy information but yield a better result than other state-of-the-art algorithms. Our code is available in: `https://github.com/JinhuaLiang/HPNs_tagging`

Index Terms— Everyday sound recognition, few shot learning, hierarchical prototypical network

1. INTRODUCTION

Everyday sound recognition (or audio tagging) is to classify the types of environmental sound events in a recording or online stream, which involves many potential scenarios such as hearing aids [1], smart cities [2], and advanced healthcare [3]. In the past decades, a great amount of deep learning methods have emerged [4, 5] exploring how to boost audio networks' performance using large-scale datasets [6, 7]. While many works turned to focus on some more practical scenarios, such as mismatched domains [8], weakly supervised learning [9], and noisy labels [10], most of these methods are still restricted by the size of available datasets. This is a practical problem in the field of everyday sound recognition as it usually takes annotators more effort to mark the categories in a recording. In addition, everyday sounds cover thousands of categories, which makes it impossible to collect sufficient instances per class for supervised learning. This is thus how few-shot learning comes into the picture.

Inspired by the human ability to learn novel items with just a few examples, few-shot learning aims to capture the pattern of an unseen category using a handful of instances [11]. A typical few-shot learning framework is depicted as an N-way K-shot problem where there are N classes in a task and each class contains K instances for training. Currently only a few studies have attempted to apply few-shot learning to environmental sound recognition tasks. Although these works pioneered few-shot audio recognition, most of them were restricted to implementing off-the-shelf few-shot learning methods from other fields explicitly, which ignores exploiting the relationship between labels in audio taxonomy.

This work is motivated by the fact that we humans learn unseen concepts not only by observing their own features, but also by connecting them to existing knowledge. We thus assume that leveraging a priori knowledge helps a model to learn an unseen category with a few examples. Based on this assumption, this paper applies hierarchical prototypical networks (HPNs) to leverage the audio taxonomy knowledge drawn from the taxonomy of the dataset. Specifically, a few-shot classification problem is considered as a multi-task classification problem where both ancestor classes and descendant classes are used in separate classification tasks. Furthermore, prototypical networks are adopted as classifiers by measuring distance between query points and prototypes in the embedding space. Experimental results on the ESC-50 dataset [12] show that our HPNs yield a superior performance over prototypical networks with no hierarchy knowledge and outperform other state-of-the-art models.

The contributions of our work are three-fold:

i) Several state-of-the-art few-shot learning algorithms are benchmarked for generic everyday sound recognition. Different experimental setups are carried out to investigate the impact of data splits on model evaluation.
ii) A hierarchical prototypical network is proposed and applied to leverage a priori knowledge of sound event taxonomy by taking samples' ancestor classes into consideration.
iii) The impact of data splits on overall performance is investigated. The code is also released to benchmark state-of-the-art few-shot algorithms and to set up an evaluation environment on the ESC-50 dataset.

The remainder of this paper is organised as follows. Section 2 briefly summarises work related to our research and Section 3 introduces our proposed hierarchical prototypical network and the implementation details. In the Section 4, experimental results are discussed to demonstrate the superior performance of our network compared with other few-shot methods. Discrepancy in performance is then discussed among different data splits. Section 5 concludes the work and points out directions for future work.

2. RELATED WORK

2.1. Few-shot learning for everyday sound recognition

Few shot learning aims to use a limited amount of labeled examples to train a model that can be generalised to unseen categories easily. Suppose C_{base} and C_{novel} are two non-overlapping label sets (or splits) drawn from the whole label set C. The task is to train a classifier f with labelled samples of classes from C_{base} and to evaluate f on samples of classes belonging to C_{novel}. Transfer learning [13] and meta learning [14, 15] are two of the most frequently used techniques. On the one hand, transfer learning strategies train f

with abundant data from C_{base} and then fine-tune f through some iterations using a limited amount of data from C_{novel}. On the other hand, meta learning (or *episodic learning*) strategies [16, 15, 14] update model parameters through a series of independent tasks in the training process. A task herein is formed by drawing N classes from C_{base} each of which contains K "training" data and Q "test" data. To differentiate the above "training" and "test" data with the conventional terms, we refer to them as *support* and *query* data instead. The model is then trained to predict the categories of query data out of N classes by offering support data. The underlying assumption of episodic learning is aligning the training process with the evaluation benefits a model's ability to generalise to novel categories. However, there exist some works [13] stating that transfer learning can have a competitive performance as well. Therefore, it still remains an open problem to work out the best practice for few-shot learning.

One of the biggest challenges for everyday sound recognition nowadays is that it covers thousands of sound classes, while only hundreds of them are available in the existing large-scale datasets [6]. There are some attempts regarding how to apply few-shot learning in the everyday sound domain. Shi et al. [17] implemented several few-shot learning algorithms on a subset of the AudioSet dataset and demonstrated the advantage of meta-learning methods in the audio domain. Wang et al. [18] curated a synthesized audio dataset for few-shot audio recognition based on FSD50K [7] and compared the state-of-the-art by controlling annotated samples, polyphony levels, and signal-to-noise ratio (SNR). Heggan et al. [19] attempted to setup a benchmark for few-shot learning techniques in audio domains. In addition to generic everyday sound recognition, the Detection and Classification of Acoustic Scenes and Events (DCASE) challenge [1] has been holding a task on fine-grained few-shot learning in the past two years which attracts increasing amount of attention. Although the previous works introduced few-shot learning paradigms to everyday sounds successfully, most of them ignored leveraging the relationship between labels in the audio taxonomy. Different from those works, this paper integrates audio taxonomy knowledge within existing few-shot algorithms.

2.2. Knowledge-based few-shot learning

Knowledge-based learning incorporates label taxonomy knowledge into the supervised learning [20]. There are some knowledge-based few-shot methods in few-shot scenarios [21, 22]. Peng et al. proposed a Knowledge Transfer Network (KTN) to incorporate visual features and semantic information for image recognition [21]. They used two independent classifiers to capture visual patterns and to conduct knowledge inference, followed by an integration network to merge them together. Due to the difference between sound and image, however, the definition of sound classes is more abstract and obscure for annotators compared with images. Garcia et al. designed hierarchical prototypical networks for music instrument recognition [22]. They aggregated classes according to a predefined instrument hierarchy and calculated the hierarchical loss by adding a weight decay to each level in the tree structure. Compared with music instrument classification, everyday sound recognition cannot be sorted into a tree structure by their physical properties merely. Some intermediate classes, such as "domestic sound" and "wild animal", are connected with psycho-acoustics directly, which makes the classification task even more complicated. Inspired by [22], this paper applies hierarchical prototypical networks to leverage the audio taxonomy knowledge derived from the dataset. Compared with their original implementation, our proposed models lower the limit on the number of prototypes for ancient prototypes generation, which encourages the model to use the audio taxonomy knowledge in more circumstances.

[1] https://dcase.community/

3. HIERARCHICAL PROTOTYPICAL NETWORKS

3.1. Prototypical networks

Prototypical networks were proposed in [16] to train a few-shot model using a series of independent tasks. Prototypical networks learn an embedding space where classification is performed by computing distances between each query sample and prototypes. Suppose a support set of $N \times K$ examples be $S = \{(x_1, y_2), ..., (x_{N\times K}, y_{N\times K})\}$ where each $x_i \in \mathbb{R}^D$ is the D-dimensional feature vector of an example obtained from an encoder f and $y_i \in \{1, ..., K\}$ be the corresponding label. A prototype could be calculated by averaging the embeddings of support samples belonging to its class:

$$c_k = \frac{1}{|S_k|} \sum_{(x_i, y_i)\in S_k} f_\phi(x_i) \tag{1}$$

where S_k denotes the set of examples whose ground truth is class k and ϕ are the learnable parameters in the encoder f.

Prototypical networks then produce a distribution over classes for a query embedding x using a softmax function over distances to the prototypes:

$$q_\phi(y = k|x) = \frac{\exp(-dist(f_\phi(x), c_k))}{\sum_m \exp(-dist(f_\phi(x), c_m))} \tag{2}$$

We applied the Euclidean distance as the distance function in our experiments. The probability distribution over classes is used to calculate cross-entropy loss.

$$L_{CE} = -\sum p(x) \log q(x) \tag{3}$$

where p, q are distributions of ground truth and predictions.

3.2. Hierarchical prototypical networks

Based on prototypical networks in Section 3.1, we devise hierarchical prototypical networks (HPNs) to incorporate the audio taxonomy knowledge into the training process. As shown in Figure 1, each level of the tree structure is treated as an independent multi-class classification task in the training stage. We thus build a HPN to extract the acoustic features with a shared encoder f and predict multiple labels corresponding to each level. Similar to (1), the acoustic features of support samples are used to calculate prototypes of the bottom level as:

$$c_k^{(0)} = \frac{1}{|S_k|} \sum_{(x_i, y_i)\in S_k} f_\phi(x_i) \tag{4}$$

To generate prototypical nodes of a higher level in the HPN, prototypes of the lower level are clustered together as per their parent level and aggregated to obtain the prototypes of a higher level:

$$c_j^{(h)} = \frac{1}{|C_k^{(h)}|} \sum_{c_j^{(h)}\in C_j^{(h)}} c_j^{(h-1)} \tag{5}$$

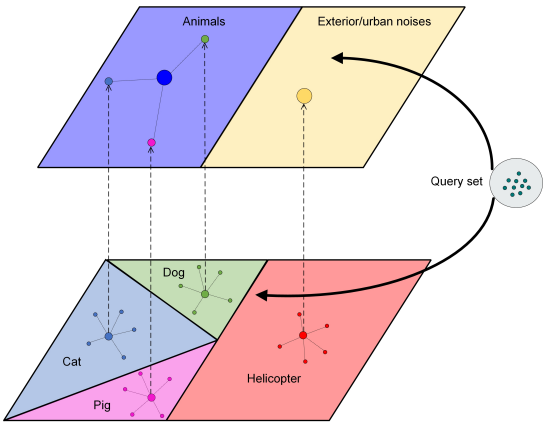


Figure 1: Illustration of training a hierarchical prototypical network in a multi-task scenario. The model learns to solve a 4-way 5-shot problem on the bottom level while trying to conduct a 2-way classification on a higher level.

Table 1: The architecture of the designed encoder.

Block name	Filter size	Output shape
Conv block 1	3×3@64	(64, 32, 215)
Conv block 2	3×3@128	(128, 16, 107)
Conv block 3	3×3@256	(256, 8, 53)
Conv block 4	3×3@512	(512,)
Fully connected layer	256	(256,)

where $C_j^{(h)}$ is a set of prototypes belonging to the class j at the h-th level in the taxonomy. The HPN iterates the clustering and aggregation process until the prototypes of the highest level are calculated.

Different from the original hierarchical network [22] which ignores an ancestor node if the number of its child nodes is less than two, HPN takes this ancestor class into consideration to fully exploit the hierarchical information. We believe this can leverage the audio taxonomy knowledge even better. The Euclidean distance between the query embedding and prototypes in each level is then calculated and used for classification. The evaluation process is similar to the training except only the classification task of the bottom level in HPNs will be taken into consideration for a fair comparison.

Following the architecture of VGGNet [23], we design a convolutional neural network containing 8 convolutional layers as the backbone of our HPNs, as shown in Table 1. Each block consists of two identical convolutional filters. Except for the last block, a max pooling layer with strides equal to 2 is appended to the block. A global pooling operation is used in the last block. Finally, the network outputs audio embedding sized 256.

3.3. Structural loss

Let h be the hierarchical level of a node in the tree structure, we can calculate the hierarchical loss [22] by using the cross-entropy loss function as follows:

$$L_{hierarchical} = \sum_{h=0}^{H} e^{\alpha h} L_{CE}^{(h)} \quad (6)$$

where $L_{CE}^{(h)}$ is the cross-entropy loss in the level h, H is the height of the taxonomy, and α is a hyper-parameter to control the loss decay of each level with respect to their height in the hierarchy. We set α equal to 1 in all experiments.

4. EVALUATION

4.1. Dataset

Table 2: Parent classes and examples of the child classes in ESC-50

Parent class	**Examples of child classes**
Animals	Dogs, Rooster, Pig, Cow, ...
Natural & water soundscapes	Rain, Sea waves, Crickets, ...
Human, non-speech sounds	Crying baby, Sneezing, ...
Interior/domestic sounds	Door knock, Clock tick , ...
Exterior/urban noises	Engine, Chainsaw, Siren, ...

The few-shot learning methods are evaluated on the ESC-50 dataset [12]. ESC-50[2] is a collection of sound events which consists of 2000 5-second recordings in total. Theses recordings are assigned one label out of 50 child classes. Table 2 shows the tree structure of ESC-50. It can be observed that these child classes are loosely arranged into 5 parent categories: "Animals", "Natural & water soundscapes", "Human, non-speech sounds", "Human, non-speech sounds", "Interior/domestic sounds", "Exterior/urban noises". It should be noted that neither the original nor our experimental setting uses the parent classes in the evaluation stage.

4.2. Comparative methods

In addition to prototypical networks, we also use a selection of few-shot algorithms for comparison [13, 14]. The transfer learning method in [13] trains an encoder from scratch using the base split. It then applies this trained model with the best validation performance for few-shot evaluation using the novel split. Another work in [14] used matching networks to calculate an attention matrix between query samples and support samples. The attention matrix is multiplied by matrix of support labels to obtain the logits of the query ones.

Except matching networks, we apply the identical encoder as described in Table 1. For the matching network we build a model following the implementation in [14]. As all of the comparative methods are metric-based, we apply the Euclidean distance to assess the similarity between two samples.

4.3. Evaluation metrics

As with Sect.6.5 in [24], this work uses *accuracy* to measure the ability of a classifier to make the correct decisions. The F_1 score is calculated to trade off between the ability to make correct decisions and to retrieve positive samples.

4.4. Experiment setup

Inputs for the few-shot methods are log Mel spectrograms. We set the sampling rate to 44100 Hz. The window length is 1024 sample points (roughly 20ms) with 50% overlap, and the number of Mel bank filters is 64. We finally get spectrograms sized 431×64. Before forwarding the extracted feature into network, frequency normalisation is applied along each Mel bin.

[2]Dataset available in https://github.com/karolpiczak/ESC-50

Following [22], in addition to a typical 5-way 5-shot problem, all few-shot learning methods were evaluated in a 12-way classification as well. For both experiments, each task (or episode) contains 5 support samples and 5 query samples. i.e. $K = 5$ and $Q = 5$. There are 32 episodes for each epoch. For a fair comparison, all models are trained through 100 epochs using the Adam optimiser with learning rate equal to 0.0001. 5-fold cross-validation was used throughout experiments. We randomly split the label set into two non-overlapping datasets, train and evaluation sets, as per the ratio 7:3. Following [22] we ensure ratios of children classes under same parent classes are identical between the train and evaluation process. We refer to this data split method as *uniform split* hereafter. It should be noted that the split label sets are changed with respect to different folds in the cross-validation, but all methods in the same folds share the same sets of train and evaluation.

4.5. Experimental results

Table 3: The performance of 12-way 5-shot learning methods on the ESC-50 dataset. The best one is highlighted in **bold**.

	$Accuracy$	F_1
Transfer Learning [13]	72.90%	72.87%
Proto [16]	77.70%	77.52%
Matching [14]	71.81%	71.75%
HPN (ours)	**78.65%**	**78.51%**

Table 4: The performance of three episodic learning methods for 5-way 5-shot problems on the ESC-50 dataset. The best one is highlighted in **bold**.

	$Accuracy$	F_1
Proto [16]	88.18%	88.18%
Matching [14]	86.83%	86.83%
HPN (ours)	**88.90%**	**88.88%**

Table 3 compares four few-shot learning methods in terms of accuracy and F1-score. Our hierarchical prototypical network yields the best performance among the transfer learning method, prototypical network (Proto), and matching network. The hierarchical prototypical network outperforms the best baseline (i.e. the prototypical network) by 0.95% and 0.99% in terms of accuracy and F1-score, respectively. This indicates that audio taxonomy knowledge can help an encoder to learn a better embedding space. Table 4 compares three episodic learning methods for 5-way 5-shot problems. Our HPNs can still yield a superior performance than prototypical networks and matching networks.

4.6. Impact of data split

In order to investigate the impact of data splits on overall performance, we compare different data split methods using the same few-shot model. Table 5 shows results of baseline prototypical networks with three data split methods. The *random split* method herein is to select 15 classes as novel classes randomly, so that the ratios of children classes under the same parent ones are not fixed. The *parent split* method is to select all child classes under the same parent category and fill this selected class set with the classes belonging to the rest parent categories. In this way, we make the number of classes in the selected set obtained by parent split method identical to the one by other methods. The descending order of the performance of three data split methods is "Uniform" > "Random" > "Parent". This observation adheres to our intuition that the performance on evaluation drops as the gap between the distribution of a base split and a novel split gets bigger. It also demonstrates that the characteristics between child classes from the same ancestor class are more similar than those from different classes, suggesting the importance of audio taxonomy knowledge in classification tasks.

Table 5: The performance of prototypical networks with different data splits on the ESC-50 dataset.

	acc	F_1
Random	74.59%	74.59%
Parent	73.35%	73.35%
Uniform	77.70%	77.52%

5. CONCLUSION AND FUTURE WORK

This work designs a hierarchical prototypical network for everyday sound recognition. The network extracts acoustic features and generates prototypical nodes corresponding to multiple levels in the tree structure. Distances between query samples and prototypical nodes in each level are then calculated and used for classification separately. Compared with prototypical networks with no hierarchy information, our model achieved a better performance in terms of accuracy and F_1-score.

Although hierarchical prototypical networks proves that it is promising to incorporate the audio taxonomy knowledge in few-shot everyday sound recognition, it still suffers from some limitations. First, HPNs cannot be applied to some datasets where an explicit taxonomy is not available. Second, some taxonomies are too complex to assign a hierarchical level to each label (e.g., a label having multiple paths to the root). In the future we plan to extend HPNs to a more complicated large-scale dataset. In addition, it is also intriguing to explore knowledge-based methods without an explicit taxonomy.

6. ACKNOWLEDGEMENT

This work was supported by the Engineering and Physical Sciences Research Council [grant number EP/T518086/1]. The research utilised Queen Mary's Apocrita HPC facility, supported by QMUL Research-IT, http://doi.org/10.5281/zenodo.438045.

7. REFERENCES

[1] X. Fan, T. Sun, W. Chen, and Q. Fan, "Deep neural network based environment sound classification and its implementation on hearing aid app," *Measurement*, vol. 159, p. 107790, July 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0263224120303286

[2] T. Spadini, D. L. d. O. Silva, and R. Suyama, "Sound Event Recognition in a Smart City Surveillance Context," *arXiv:1910.12369 [cs, eess, stat]*, Feb. 2020, arXiv:

1910.12369. [Online]. Available: http://arxiv.org/abs/1910.12369

[3] B. W. Schuller, D. M. Schuller, K. Qian, J. Liu, H. Zheng, and X. Li, "COVID-19 and Computer Audition: An Overview on What Speech & SoundAnalysis Could Contribute in theSARS-CoV-2 Corona Crisis," *Frontiers in digital health*, vol. 3, p. 14, 2021, publisher: Frontiers.

[4] Y. Gong, Y.-A. Chung, and J. Glass, "AST: Audio Spectrogram Transformer," in *Proc. Interspeech 2021*, 2021, pp. 571–575.

[5] T. Zhang, J. Liang, and B. Ding, "Acoustic scene classification using deep CNN with fine-resolution feature," *Expert Systems with Applications*, vol. 143, p. 113067, Apr. 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957417419307845

[6] J. F. Gemmeke, D. P. W. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio Set: An ontology and human-labeled dataset for audio events," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. New Orleans, LA: IEEE, Mar. 2017, pp. 776–780. [Online]. Available: http://ieeexplore.ieee.org/document/7952261/

[7] E. Fonseca, X. Favory, J. Pons, F. Font, and X. Serra, "FSD50K: An Open Dataset of Human-Labeled Sound Events," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 30, pp. 829–852, 2022, conference Name: IEEE/ACM Transactions on Audio, Speech, and Language Processing.

[8] B. Kim, S. Yang, J. Kim, and S. Chang, "QTI Submission to DCASE 2021: Residual Normalization for Device-Imbalanced Acoustic Scene Classification with Efficient Design," DCASE2021 Challenge, Tech. Rep., June 2021.

[9] Y. Xu, Q. Kong, W. Wang, and M. D. Plumbley, "Large-Scale Weakly Supervised Audio Classification Using Gated Convolutional Neural Network," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Calgary, AB: IEEE, Apr. 2018, pp. 121–125. [Online]. Available: https://ieeexplore.ieee.org/document/8461975/

[10] T. Iqbal, Y. Cao, A. Bailey, M. D. Plumbley, and W. Wang, "ARCA23K: An Audio Dataset for Investigating Open-Set Label Noise," in *Proceedings of the 6th Detection and Classification of Acoustic Scenes and Events 2021 Workshop (DCASE2021)*, Barcelona, Spain, Nov. 2021, pp. 201–205.

[11] X. Li, Z. Sun, J.-H. Xue, and Z. Ma, "A concise review of recent few-shot meta-learning methods," *Neurocomputing*, vol. 456, pp. 463–468, Oct. 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925231220316222

[12] K. J. Piczak, "ESC: Dataset for Environmental Sound Classification," in *Proceedings of the 23rd ACM international conference on Multimedia*. Brisbane Australia: ACM, Oct. 2015, pp. 1015–1018. [Online]. Available: https://dl.acm.org/doi/10.1145/2733373.2806390

[13] Y. Tian, Y. Wang, D. Krishnan, J. B. Tenenbaum, and P. Isola, "Rethinking Few-Shot Image Classification: A Good Embedding is All You Need?" in *Computer Vision – ECCV 2020*, ser. Lecture Notes in Computer Science, A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, Eds. Cham: Springer International Publishing, 2020, pp. 266–282.

[14] O. Vinyals, C. Blundell, T. Lillicrap, k. kavukcuoglu, and D. Wierstra, "Matching Networks for One Shot Learning," in *Advances in Neural Information Processing Systems*, vol. 29. Curran Associates, Inc., 2016. [Online]. Available: https://proceedings.neurips.cc/paper/2016/hash/90e1357833654983612fb05e3ec9148c-Abstract.html

[15] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *International Conference on Machine Learning*. PMLR, 2017, pp. 1126–1135.

[16] J. Snell, K. Swersky, and R. Zemel, "Prototypical Networks for Few-shot Learning," *Advances in Neural Information Processing Systems*, vol. 30, pp. 4077–4087, 2017.

[17] B. Shi, M. Sun, K. C. Puvvada, C.-C. Kao, S. Matsoukas, and C. Wang, "Few-Shot Acoustic Event Detection Via Meta Learning," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 76–80, iSSN: 2379-190X.

[18] Y. Wang, N. J. Bryan, J. Salamon, M. Cartwright, and J. P. Bello, "Who Calls The Shots? Rethinking Few-Shot Learning for Audio," in *2021 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*. IEEE, 2021, pp. 36–40.

[19] C. Heggan, S. Budgett, T. Hospedales, and M. Yaghoobi, "MetaAudio: A Few-Shot Audio Classification Benchmark," *arXiv:2204.02121 [cs, eess]*, Apr. 2022, arXiv: 2204.02121. [Online]. Available: http://arxiv.org/abs/2204.02121

[20] H. Phan, L. Hertel, M. Maass, P. Koch, R. Mazur, and A. Mertins, "Improved Audio Scene Classification Based on Label-Tree Embeddings and Convolutional Neural Networks," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 6, pp. 1278–1290, June 2017, conference Name: IEEE/ACM Transactions on Audio, Speech, and Language Processing.

[21] Z. Peng, Z. Li, J. Zhang, Y. Li, G.-J. Qi, and J. Tang, "Few-Shot Image Recognition With Knowledge Transfer," 2019, pp. 441–449. [Online]. Available: https://openaccess.thecvf.com/content_ICCV_2019/html/Peng_Few-Shot_Image_Recognition_With_Knowledge_Transfer_ICCV_2019_paper.html

[22] H. F. Garcia, A. Aguilar, E. Manilow, and B. Pardo, "Leveraging Hierarchical Structures for Few-Shot Musical Instrument Recognition," in *Proceedings of the 22nd International Society for Music Information Retrieval Conference, ISMIR 2021, Online, November 7-12, 2021*, J. H. Lee 0001, A. Lerch 0001, Z. Duan, J. Nam, P. Rao, P. v. Kranenburg, and A. Srinivasamurthy, Eds., 2021, pp. 220–228. [Online]. Available: https://archives.ismir.net/ismir2021/paper/000027.pdf

[23] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," in *International Conference on Learning Representations*, 2015.

[24] T. Virtanen, M. D. Plumbley, and D. Ellis, Eds., *Computational Analysis of Sound Scenes and Events*. Cham: Springer International Publishing, 2018. [Online]. Available: http://link.springer.com/10.1007/978-3-319-63450-0

SEGMENT-LEVEL METRIC LEARNING FOR FEW-SHOT BIOACOUSTIC EVENT DETECTION

Haohe Liu[1], *Xubo Liu*[1], *Xinhao Mei*[1], *Qiuqiang Kong*[2], *Wenwu Wang*[1], *Mark D. Plumbley*[1]

[1] Centre for Vision, Speech, and Signal Processing (CVSSP), University of Surrey, UK
[2] Speech, Audio, and Music Intelligence (SAMI) Group, ByteDance, China

ABSTRACT

Few-shot bioacoustic event detection is a task that detects the occurrence time of a novel sound given a few examples. Previous methods employ metric learning to build a latent space with the labeled part of different sound classes, also known as positive events. In this study, we propose a segment-level few-shot learning framework that utilizes both the positive and negative events during model optimization. Training with negative events, which are larger in volume than positive events, can increase the generalization ability of the model. In addition, we use transductive learning on the validation set during training for better adaptation to novel classes. We conduct ablation studies on our proposed method with different setups on input features, training data, and hyper-parameters. Our final system achieves an F-measure of 62.73 on the DCASE 2022 challenge task 5 (DCASE2022-T5) validation set, outperforming the performance of the baseline prototypical network 34.02 by a large margin. Using the proposed method, our submitted system ranks 2nd in DCASE2022-T5 with an F-measure of 48.2 on the evaluation set. The code of this paper is open-sourced[1].

Index Terms— few-shot learning, transductive learning, metric learning, audio event detection

1. INTRODUCTION

Few-shot learning (FSL) [1] is a machine learning problem that makes predictions based on the training data that contains limited information. Sound event detection (SED) [2] is a task that locates the onset and offset of certain sound classes. By combining the idea of FSL with SED [3], a system can detect a new type of sound with only a few examples. Few-shot SED is useful for audio data labeling, especially when the user needs to detect a new type of sound.

Most prior studies use a prototypical network [4] as the main architecture [5, 6, 7, 8]. Yang et al. [5] propose a mutual learning framework that employs transductive learning to iteratively improve the feature extractor and classifier, where transductive learning means the model has access to the test set without labels during the training process. A smoother manifold of embedding space can help extend the decision boundary and reduce the noise in data representation [9]. Tang et al. [6] propose to use embedding propagation [9] in few-shot SED to learn a smoother manifold by interpolating between the model output features based on a similarity graph. Data augmentations such as spec-augment and mixup are used in the method described in [7, 8]. There is also a spectrogram-cross-correlation-based method called template matching [10], which performs detection based on the normalized cross-correlation between example sound event and unlabeled data.

[1] https://github.com/haoheliu/DCASE_2022_Task_5

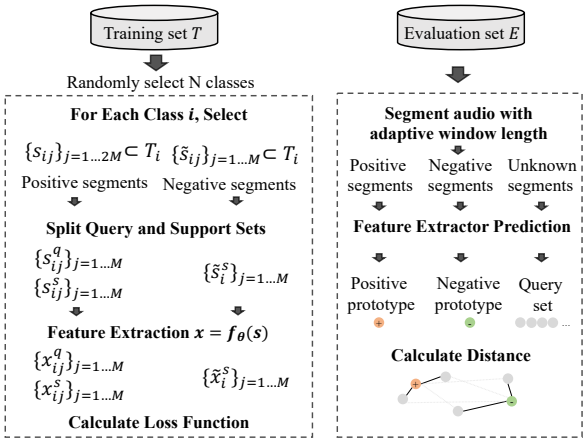
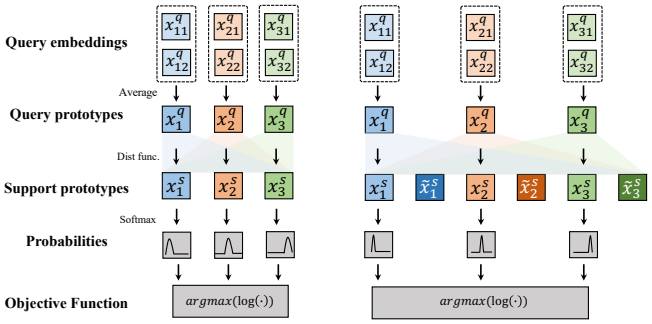


Figure 1: Training and evaluation procedure of the N-way-M-shot segment-level metric learning. M denotes the number of segments or embeddings.

Metric learning [11] refers to learning a distance function and feature space for a task. Previous metric-learning-based studies [5, 6] usually optimize the model with the labeled positive events, by grouping and separating the latent prototypes of the events with the same and different classes, respectively. The audio chunks that do not contain target events, which we refer to as negative events, are larger in volume but receive less attention. For example, in the DCASE 2022 task 5 development set [10], the duration of the negative events is 19.18 hours, accounting for 91.3% of the training data with a total duration of 21 hours.

In this paper, we propose a segment-level metric learning method that achieves state-of-the-art results on the few-shot bioacoustic detection task. As shown in Figure 1, our system operates on a segment level. Each sound event can contain multiple segments. We train a feature extraction network that maps the segments into latent embeddings, which are averaged into prototypes to represent different sound classes. To learn a robust latent space, we use a transductive learning scheme and propose to build contrastive loss with negative events. We also improve our method by using feature selection, data augmentation, and post-processing. We perform ablation studies to measure the effectiveness of each component. Our proposed method achieves an F-measure of 62.73 on the DCASE task 5 validation set.

This paper will be organized as follows. Section 2 provides an overview of our system. Section 3 introduces our methodology. Section 4 discusses the experimental setup. Section 5 reports the result and the ablation studies. Section 6 summarizes this work and provides a conclusion.



(a) Previous method. (b) Proposed method.

Figure 2: This figure illustrates N-way-M-shot metric learning when N=3 and M=2. (a) Visualization of the previous method, which only uses positive classes. (b) The proposed metric learning with the negative segments we used in our system. Support embeddings are omitted for simplicity. x_i and $\widetilde{x}_i$ stand for the positive and negative prototypes of class i.

2. SYSTEM OVERVIEW

We build our system using a prototypical network [4], which is widely used for metric-based few-shot learning. The training data $\mathbb{T} = (\mathbb{S}_i, \mathbb{Y}_i)|_{i=1}^{N_{\text{train}}}$ contains audio feature set $\mathbb{S}_i = \{s_i|y_i = 1\} \sqcup \{\widetilde{s}_i|y_i = 0\}$ and its corresponding label set $\mathbb{Y}_i = \{y_i|y_i \in \{0, 1\}\}$, where $\{s_i\}$ and $\{\widetilde{s}_i\}$ are the sets of positive and negative segments for class i, respectively, and N_{train} is the total number of training classes. The evaluation dataset $\mathbb{E} = (\mathbb{S}'_i, \mathbb{Y}'_i)|_{i=1}^{N_{\text{eval}}}$ also contains an audio feature set $\mathbb{S}'_i = \{s'_i\}$ and a label set $\mathbb{Y}'_i = \{y'_i\}$, where N_{eval} is the number of classes in the evalution set, $|\mathbb{S}'_i| = L_i$ and $|\mathbb{Y}'_i| = K$. Here we have $L_i \geq K$ because the evaluation set is partially labeled with only first K events. The validation set has the same structure as the evaluation set. The objective of our system is properly mapping different audio features into a latent embedding within a high-dimensional space, where similar audio features are closer together.

We use episodic training [12] to optimize our system in an N-way-M-shot way. As illustrated in Figure 2(a), N-way-M-shot means each training batch will select data from N classes. And for each classes i, the system will randomly select M segments $\{s^s_{ij}\}_{j=1...M}$ as support segments and another M segments $\{s^q_{ij}\}_{j=1...M}$ as query segments. All the segments in different classes have the same length. Then a feature extraction network (Section 3.1) will map these segments into fix-length embeddings, which are later averaged into query prototypes x^q_i and supporting prototypes x^s_i. The system is optimized by minimizing the distance between the query and support prototypes with the same class. To build a robust latent space and generalize better to the new class, we propose to use the negative event in metric learning and the transductive learning scheme in Section 3.2 and 3.3.

During evaluations, the audio file will be segmented using a sliding window with an adaptive segment length (Section 3.4). The segments in the labeled parts will be used to build positive and negative prototypes, which are treated as the latent representation of the positive and negative events in an audio file. The segments in the unlabeled part are the query set, which can be classified by calculating and comparing the distance with the positive and negative prototype (Section 3.5). And if the probability of one query belonging to a positive prototype is greater than a threshold h, it will be classified as positive. Consecutive positive predictions will be merged into one single event.

3. METHODOLOGY

3.1. Feature extraction network

Our feature extraction network f_θ is a convolutional neural network (CNN) based architecture that maps the audio feature s into a latent embedding x. In a similar way to the architecture proposed by [13], the network f_θ consists of three convolutional blocks with hidden channels of sizes 64, 128, and 64. Each convolutional block consists of three two-dimensional CNN layers with batch normalization and leaky rectified linear unit activations [14]. As a common trick in CNN-based network [13, 15], we apply 2×2 max-pooling after each block for downsampling and enlarging the reception field. The input and output of each convolutional block have a residual connection processed by a downsampling CNN layer. In order to maintain the same output dimension with different input lengths, we apply an adaptive average pooling at the end of the network. The final output feature map after adaptive pooling is a $C \times T \times F$ size block, which is the final latent embedding of s.

3.2. Segment-level metric learning

We propose to utilize negative segments within negative events during model optimization to learn a more robust representation, as illustrated in Figure 2(b). In a similar way to [3], we first divide the audio features into segments with equal length for metric learning. Then f_θ maps all the segments into latent embeddings. During optimization, we will calculate the class probabilities distributions of the query prototype x^q_i, which involves the distance calculation with all the positive and negative support prototypes. In this case, the model can learn a larger amount of contrastive information from the negative events on building the latent space. Specifically, we first calculate a distance matrix $\mathbf{D} = [\mathbf{d}^{(1)}, \mathbf{d}^{(2)}, ..., \mathbf{d}^{(N)}]^T$ according to Equation 1,

$$\mathbf{d}^{(i)}_{2j} = \|x^q_i - x^s_j\|_2, \ \mathbf{d}^{(i)}_{2j+1} = \|x^q_i - \widetilde{x}^s_j\|_2, \tag{1}$$

where $\mathbf{d}^{(i)} \in \mathbb{R}^{2N}$ stands for the distance between x^q_i and $2N$ support prototypes, and $\widetilde{x}^s_j$ denotes the support prototype for the negative events of class j. Then we optimize our model by maximizing the probability that x^q_i is close to the positive support prototype of class i, x^s_i, given by

$$\mathbf{d}'^{(i)} = \log(\text{Softmax}(-\mathbf{d}^{(i)})), \ l = \arg\max_\theta(\Sigma^N_{i=1}(\mathbf{d}'^{(i)}_{2i}), \tag{2}$$

where $0 \leq i, j \leq N, i, j \in \mathbb{N}$, and l is the objective function. Note that the learning process does not involve the query prototypes for negative events $\widetilde{x}^q_i$, because $\widetilde{x}^q_i$ and $\widetilde{x}^s_i$ are not guaranteed to have the same type of sound.

Data balancing is important in this task because different sound classes have different total durations [10]. In order to balance between classes, we sample each class with equal probability during the episodic training. In this way, the model has equal probabilities to attend to each class and will be less prone to overfitting [16].

3.3. Transductive learning

We adopt a transductive learning [17] approach during training, which means our model will be optimized both on the fully-labeled training set and the partially labeled evaluation data. Each file in

evaluation data has first K labeled events for a particular type of sound. We treat these K events as positive events and the remaining K chunks of audio in the labeled part as negative events. In the evaluation set, although the sound class of each file is not available, files with the same sound class should be in the same subfolder, and we treat each subfolder of the evaluation set as a different sound class. Even though the files within each subfolder may not always contain the same target sound, our experiment shows transductive learning in this way can still help the model gain better adaptation to the evaluation set (Section 3).

3.4. Adaptive segment length

We use the same segment length among all classes during training for the convenience of batch processing. But during evaluation, using the same segment length is not ideal. For example, using a segment length that is too long or too short will tend to have a high false negative rate or false positive rate, respectively. In the evaluation set, different animal or bird species have drastically different lengths of vocalization, ranging from 30 milliseconds to 5 seconds. Thus we choose to use adaptive segment lengths during evaluation.

$t_{\max}$ (s)	[0,0.1]	(0.1,0.4]	(0.4,0.8]	(0.8,3.0]	(3.0, ∞)
Length	8	$t_{\max}$	$t_{\max}$ / 2	$t_{\max}$ / 4	$t_{\max}$ / 8

Table 1: The segment length we use on dividing the evaluation audio file for different values of $t_{\max}$.

As shown in Table 1, we set different segment length for each audio file based on the max length of the labeled events $t_{\max} = \max(t_1, ..., t_K)$, where $t_1, ..., t_K$ denotes the duration of the K labeled positive events. We set the hop length as one-third of the window length. Note the parameters here are chosen by experience and not necessarily optimal.

3.5. Positive and negative prototypes

During the evaluation, we assume the first K labeled positive events do not contain too much variety, therefore we calculate the positive prototype by averaging the embeddings of the labeled positive segments. By comparison, building negative prototypes is more tricky because negative segments can contain many different kinds of sounds. So simply averaging all the negative embeddings would result in a sub-optimal representation of negative prototypes. To address these challenges, we choose to run our evaluation six times, each selecting 30 randomly selected negative segments within the labeled negative parts, and we average the predicted probabilities across time of six runs as the final prediction. The negative prototype in each run can have a chance to represent different sounds. This process is similar to the random subspace method [18], in which the ensemble of several estimators trained with a different subset of training data can outperform a single estimator optimized on full training data.

4. EXPERIMENTS

4.1. Dataset

DCASE2022-T5 The DCASE 2022 task 5 dataset[2] contains a training set, a validation set, and an official evaluation set. The training and validation set are both fully labeled. The official evaluation set has the labels of the first five positive events. Our result on the evaluation set is available on the DCASE 2022 Challenge result page[3]. The full label of the official evaluation set is not released at the time of writing, hence, we mainly report the result on the validation set in this paper. The validation during training is not meant to pick the best model. That's because we perform validation in a different way from evaluation. Similar to the training process, we calculate validation accuracy on a fix-length segment level without adaptive segment length. Therefore the best model on validation does not necessarily perform the best during evaluation. Nevertheless, we use the same validation process in our experiments, so the comparisons are fair in different settings. There are also similar ideas in [19, 20], which utilize the evaluation set for validations.

AudioSet-Aminal-SL AudioSet [21] is a large-scale dataset for audio research [13, 22]. Considering that the training set of DCASE2022-T5 only contains 47 different sound classes, we choose to use the strongly labeled part of the AudioSet dataset[4] to augment training data with a wider variety of sounds. To alleviate the domain mismatch problem, we only use sound labels that are related to animal vocalizations and do not overlap with other non-animal sounds. After data cleaning, we have 1796 pieces of audio with 37 classes from AudioSet. However, even if the sounds have the same label in the AudioSet, they can still sound very different. To alleviate this problem, we treat each audio file in AudioSet as its own class, so we have 1796 classes in this dataset, which is named AudioSet-Aminal-SL, where SL means strongly labeled. To balance the 1796 classes and 47 classes in AudioSet-Aminal-SL and DCASE2022-T5, we choose half classes from each dataset during episodic training.

4.2. Evaluation metric

We use the F-measure score, the official evaluation metric provided by the organizers of DCASE task 5, as our main evaluation metric. We also report system performance with the Polyphonic Sound Detection Score (PSDS) [23], which is a robust intersection-based sound event detection evaluation metric. In PSDS, we set the detection tolerance criterion (DTC) and the ground truth intersection criterion (GTC) to 0.5, and the maximum effective false positive rate to 100.0. Other parameters like the cross-trigger tolerance criterion (CTTC) are not used because our task is not polyphonic detection.

4.3. Experimental setup

Following [5], all the audio data are resampled to a 22.5 kHz sampling rate. The input feature of our system is the stack of PCEN [24] and ΔMFCC [25] features. In the short-time Fourier transform, we set the window length as 1024 and the hop size as 256. We set the mel-frequency dimension as 128. The input length of our model during training is 0.2 seconds. If the sound event is less than 0.2 seconds, zero-padding will be applied. The size of the embedding mentioned in Section 3.1 is 2048, in which $C = 64, T = 4, F = 8$. All the experiments use an initial learning rate of 0.001 with 0.65 exponential decay every 10 epochs. We perform validation after every epoch. We perform validation in a 3-way-5-shot manner since there are only three classes (HB, ME, PB) in the validation set. We will stop model training if the validation accuracy does not improve

[2] https://zenodo.org/record/6482837

[3] https://dcase.community/challenge2022/task-few-shot-bioacoustic-event-detection-results

[4] https://research.google.com/audioset

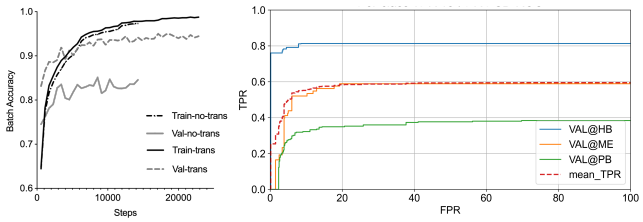
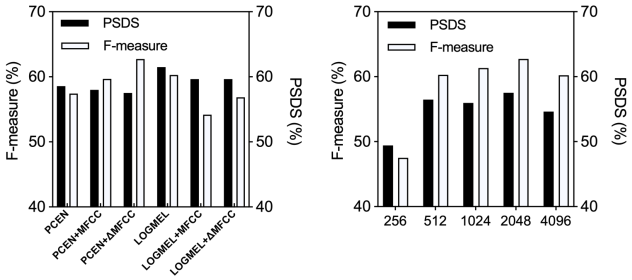


Figure 3: The training and validation accuracy at different training steps, both with and without transductive learning, are shown in the left figure. The right figure is the PSD-ROC curve of our proposed system. *HB*, *ME*, and *PB* are three subsets in the validation set. The area under *mean_TPR* curve is PSDS, which indicates the system's overall performance. TPR and FPR stand for true positive rate and false positive rate, respectively.



(a) Input features (b) Embedding dimensions

Figure 4: Ablation study on (a) input features; and (b) embedding dimension. We report the PSDS and F-measure on the DCASE2022-T5 validation dataset.

for 10 consecutive epochs. And the model with the best validation accuracy is used for calculating metrics scores. To make full use of training data, we implement a dynamic data loader that generates training data with a random starting time on the fly. We assume the duration of one vocalization for a certain animal does not vary significantly. Therefore, we design the post-processing strategy for a sound class based on maximum length of positive event $t_{\max} = \max(t_1, ..., t_K)$. We will remove a positive detection if its length is smaller than $\alpha * t_{\max}$ or greater than $\beta * t_{\max}$. We use different combinations of $\beta = 2.0, \alpha = [0.1, 0.2, ..., 0.9], h = [0.0, 0.05, ..., 0.95]$ to calculate data points [23], draw the PSD-ROC curve, and calculate PSDS. We choose the best F-measure among all β, α, h combinations as the final F-measure.

5. RESULT

Method	Pre.	Rec.	F-measure	PSDS
Template Matching [10]	2.42	18.32	4.28	N/A
ProtoNet (official) [10]	36.34	24.96	29.59	N/A
ProtoNet (our impl)	23.26	63.27	34.02	46.10
Proposed	69.30	57.30	**62.73**	**57.52**

Table 2: Comparisons with baseline template matching and prototypical network methods. *Pre.* and *Rec.* stand for precision and recall, respectively. The first two methods [10] did not report PSDS results. All the metrics are written in percentages.

The performance of our system on the validation set is reported in Table 2. The F-measure score of template matching and our re-implemented prototypical network baseline [10] is 4.28 and 34.02, respectively. Our system outperforms the baselines by a large margin with an F-measure score of 62.73 and a PSDS of 57.52.

As is shown in Figure 3, using transductive learning can significantly improve the validation accuracy. And the class-wise ROC indicates the *HB* class, which is mostly mosquito sounds, is the easiest one to detect. Class *PB* is the hardest class perhaps because it mainly consists of sparse bird calls with strong background noise. Class *ME* achieves an average performance in the validation set.

We perform a study on the effect of the input feature. As shown in Figure 4(a), the performance of F-measure and PSDS is not always consistent, and we use F-measure to guide our selection considering it is widely used in prior studies [10]. By comparing the F-measure score, PCEN+ΔMFCC appears to be a good feature combination on the validation set. We also compare different embedding dimension in Figure 4(b). We change the dimension by altering the dimension of F in the adaptive average pooling. We notice a dimension of 512 can considerably improve over 256, and 2048 has the best performance among all the settings.

We perform ablations on each of the components we proposed. As shown in Table 3, if we remove the negative segments, the performance drops considerably. The trend is the same with transductive learning and post-processing. We also study the effect of training data. In Table 4, we can see that the best F-measure score is achieved using the DCASE2022-T5 only. Using AudioSet-SL leads to an F-measure of 46.83 and a PSDS of 51.00. By combining two datasets we got an F-measure of 58.48 and a best PSDS of 58.77. We hypothesize that the degradation of F-measure using AudioSet is caused by domain mismatch on training data. However, combining two datasets yield the best PSDS, which means using AudioSet data can lead to a general improvement across all threshold and post-processing settings instead of getting a single best system with a high F-measure. This indicates that PSDS might be a suitable metric for the community to reference in this task.

Setting	F-measure (%)	PSDS (%)
Proposed	**62.73**	**57.52**
w/o Negative contrast	55.25	54.95
w/o Transductive learning	56.37	54.50
w/o Post processing	57.27	55.90

Table 3: Ablation study of the proposed method.

Training data	F-measure (%)	PSDS (%)
DCASE	**62.73**	57.52
AudioSet-Aminal-SL	46.83	51.00
AudioSet-Aminal-SL & DCASE	58.48	**58.77**

Table 4: A study on using different training datasets. DCASE stands for the DCASE2022-T5 dataset.

6. CONCLUSIONS

This paper proposes a new framework for few-shot sound event detection. Our proposed metric learning with negative segments and the transductive learning scheme can significantly improve model performance. On the input feature, our experiment shows that PCEN with ΔMFCC yields the best performance in our settings. Our result also indicates that PSDS might be a useful metric to evaluate the model's overall performance by considering multiple thresholds and post-processing settings.

7. ACKNOWLEDGMENT

This research was partly supported by a Newton Institutional Links Award from the British Council (Grant number 623805725), BBC Research and Development, Engineering and Physical Sciences Research Council (EPSRC) Grant EP/T019751/1 "AI for Sound", and a PhD scholarship from the Centre for Vision, Speech and Signal Processing (CVSSP), Faculty of Engineering and Physical Science (FEPS), University of Surrey. For the purpose of open access, the authors have applied a Creative Commons Attribution (CC BY) license to any Author Accepted Manuscript version arising.

8. REFERENCES

[1] S. Ravi and H. Larochelle, "Optimization as a model for few-shot learning," *International Conference on Learning Representations*, 2017.

[2] A. Mesaros, T. Heittola, T. Virtanen, and M. D. Plumbley, "Sound event detection: A tutorial," *IEEE Signal Processing Magazine*, vol. 38, no. 5, pp. 67–83, 2021.

[3] Y. Wang, J. Salamon, N. J. Bryan, and J. P. Bello, "Few-shot sound event detection," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2020, pp. 81–85.

[4] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[5] D. Yang, H. Wang, Y. Zou, Z. Ye, and W. Wang, "A mutual learning framework for few-shot sound event detection," in *IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2022, pp. 811–815.

[6] T. Tang, Y. Liang, and Y. Long, "Two improved architectures based on prototype network for few-shot bioacoustic event detection," DCASE2021 Challenge, Tech. Rep., 2021.

[7] Y. Zhang, J. Wang, D. Zhang, and F. Deng, "Few-shot bioacoustic event detection using prototypical network with background classs," DCASE2021 Challenge, Tech. Rep., 2021.

[8] M. Anderson and N. Harte, "Bioacoustic event detection with prototypical networks and data augmentation," *arXiv:2112.09006*, 2021.

[9] P. Rodríguez, I. Laradji, A. Drouin, and A. Lacoste, "Embedding propagation: Smoother manifold for few-shot classification," in *European Conference on Computer Vision*. Springer, 2020, pp. 121–138.

[10] V. Morfi, I. Nolasco, V. Lostanlen, S. Singh, A. Strandburg-Peshkin, L. F. Gill, H. Pamula, D. Benvent, and D. Stowell, "Few-shot bioacoustic event detection: A new task at the dcase 2021 challenge." in *DCASE*, 2021, pp. 145–149.

[11] B. Kulis *et al.*, "Metric learning: A survey," *Foundations and Trends® in Machine Learning*, vol. 5, no. 4, pp. 287–364, 2013.

[12] D. Li, J. Zhang, Y. Yang, C. Liu, Y.-Z. Song, and T. M. Hospedales, "Episodic training for domain generalization," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 1446–1455.

[13] Q. Kong, Y. Cao, T. Iqbal, Y. Wang, W. Wang, and M. D. Plumbley, "PANNs: Large-scale pretrained audio neural networks for audio pattern recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 2880–2894, 2020.

[14] B. Xu, N. Wang, T. Chen, and M. Li, "Empirical evaluation of rectified activations in convolutional network," *arXiv:1505.00853*, 2015.

[15] H. Liu, L. Xie, J. Wu, and G. Yang, "Channel-wise subband input for better voice and accompaniment separation on high resolution music," *Proc. Interspeech 2020*, pp. 1241–1245, 2020.

[16] Y. Gong, Y.-A. Chung, and J. Glass, "Psla: Improving audio tagging with pretraining, sampling, labeling, and aggregation," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 3292–3306, 2021.

[17] M. Boudiaf, H. Kervadec, Z. I. Masud, P. Piantanida, I. Ben Ayed, and J. Dolz, "Few-shot segmentation without meta-learning: A good transductive inference is all you need?" in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 13 979–13 988.

[18] T. K. Ho, "The random subspace method for constructing decision forests," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 8, pp. 832–844, 1998.

[19] Q. Kong, Y. Cao, H. Liu, K. Choi, and Y. Wang, "Decoupling magnitude and phase estimation with deep resunet for music source separation," *arXiv:2109.05418*, 2021.

[20] Y. Gong, Y.-A. Chung, and J. Glass, "AST: Audio spectrogram transformer," *arXiv:2104.01778*, 2021.

[21] J. F. Gemmeke, D. P. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio set: An ontology and human-labeled dataset for audio events," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2017, pp. 776–780.

[22] Q. Kong, H. Liu, X. Du, L. Chen, R. Xia, and Y. Wang, "Speech enhancement with weakly labelled data from audioset," *arXiv:2102.09971*, 2021.

[23] Ç. Bilen, G. Ferroni, F. Tuveri, J. Azcarreta, and S. Krstulović, "A framework for the robust evaluation of sound event detection," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2020, pp. 61–65.

[24] Y. Wang, P. Getreuer, T. Hughes, R. F. Lyon, and R. A. Saurous, "Trainable frontend for robust and far-field keyword spotting," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2017, pp. 5670–5674.

[25] M. A. Hossan, S. Memon, and M. A. Gregory, "A novel approach for mfcc feature extraction," in *2010 4th International Conference on Signal Processing and Communication Systems*, 2010, pp. 946–953.

EXPLAINING THE DECISIONS OF ANOMALOUS SOUND DETECTORS

Kimberly T. Mai[1,3]*, Toby Davies*[1,3]*, Lewis D. Griffin*[1]*, Emmanouil Benetos*[2,3]

[1] University College London, UK
[2] Queen Mary University of London, UK
[3] The Alan Turing Institute, UK
{kimberly.mai, toby.davies, l.griffin}@ucl.ac.uk emmanouil.benetos@qmul.ac.uk

ABSTRACT

Deciding whether a sound is anomalous is accomplished by comparing it to a learnt distribution of inliers. Therefore, learning a distribution close to the true population of inliers is vital for anomalous sound detection (ASD). Data engineering is a common strategy to aid training and improve generalisation. However, in the context of ASD, it is debatable whether data engineering indeed facilitates generalisation or whether it obscures characteristics that distinguish anomalies from inliers. We conduct an exploratory investigation into this by focusing on frequency-related data engineering. We adapt local model explanations to anomaly detectors and show that models rely on higher frequencies to distinguish anomalies from inliers. We verify this by filtering the input data's frequencies and observing the change in ASD performance. Our results indicate that sifting out low frequencies by applying high-pass filters aids downstream performance, and this could serve as a simple pre-processing step for improving anomaly detectors.

Index Terms— anomaly detection, DCASE challenge, data engineering, interpretablity

1. INTRODUCTION

Anomalous sound detection (ASD) is the task of deciding whether a sound produced from an object is normal or anomalous. This is conducted by comparing the sound to a learnt distribution of inliers. ASD is useful for machine condition monitoring and can result in more timely repairs after unusual sounds are identified. In practice, this task is difficult because only inliers are available for training. In addition, it is hard to anticipate the types of anomalies that may appear (for example, if they manifest in certain frequency bands or at specific times). The task has received more attention in recent years, featuring within the Detection and Classification of Acoustic Scenes and Events (DCASE) challenges for three consecutive years to date [1]. In 2021, the ASD challenge received a total of 75 submissions [2]. Analysing the top submissions, we note they share many similarities. The majority utilise ensembles and use metadata to pre-train their anomaly detection models. However, there is more variability in the usage of data engineering. Notably, the top three entrants do not apply any data engineering through either data augmentations or pre-processing [3, 4].

Data engineering typically modifies the training data using pre-defined rules to improve model generalisation. Whether data engineering helps or harms anomaly detection is debatable within the machine learning community. However, existing studies have focused on the computer vision domain [5, 6]. In this paper, we seek to address this question for ASD. There are many types of data engineering steps that could be applied to audio such as speed perturbations, adding noise, and frequency masking [7]. We choose to focus on frequency-related data engineering, as [8] proposes the *high-frequency hypothesis*: anomaly detectors use higher frequencies to identify anomalies. If this is the case, methods such as random frequency masking could remove discriminative features between inliers and anomalies. This hypothesis could partly explain why the top three submissions outperform the others.

However, [8] does not explicitly substantiate their claim. We conduct a series of experiments in this paper to verify the high-frequency hypothesis. We firstly adapt Sound LIME (SLIME) [9] to a one-class anomaly detector. SLIME can provide an insight into how modifications at an individual sound clip level can affect model predictions through approximating changes in output using a linear model, hence even providing explanations for models with complex global behaviour. We find the majority of the datasets in the DCASE ASD challenge rely on higher-frequency information to make the correct decisions. Moreover, we quantitatively verify SLIME's result by re-training and re-testing our anomaly detectors by applying low and high pass filters to input data in a sequential manner. Our results corroborate that focusing on higher frequencies helps ASD, and simple frequency filtering instead of complicated data engineering is a more measured approach to improving performance.

2. APPROACH

2.1. Dataset

We use the DCASE 2021 Task 2 dataset for our experiments. This dataset combines subsets of the ToyADMOS2 [10] (*ToyCar, ToyTrain*) and the MIMII DUE [11] (*Fan, Gearbox, Pump, Slider, Valve*) datasets. All recordings are single channel, 10 seconds in duration and downsampled to 16 kHz. We refer to the subsets as different machines.

Each machine is subdivided into six sections, which correspond to different properties. Anomaly detectors are evaluated at the section level, using area under the curve (AUC) as the performance metric. Furthermore, each section contains data from two domains: the source domain (for which there is an abundant number of audio clips available) and the target domain representing a domain shift (where only a few clips are available).

The dataset is curated in a one-class fashion: the training split only contains inliers, whereas the test split contains both inliers and anomalies. The training split contains about 1,000 inliers for each of the six sections and is further subdivided into a development set

K. T. Mai, T. Davies and E. Benetos are supported by The Alan Turing Institute under grant EP/N510129/1. K. T. Mai is also supported by EPSRC under grant EP/R513143/1.

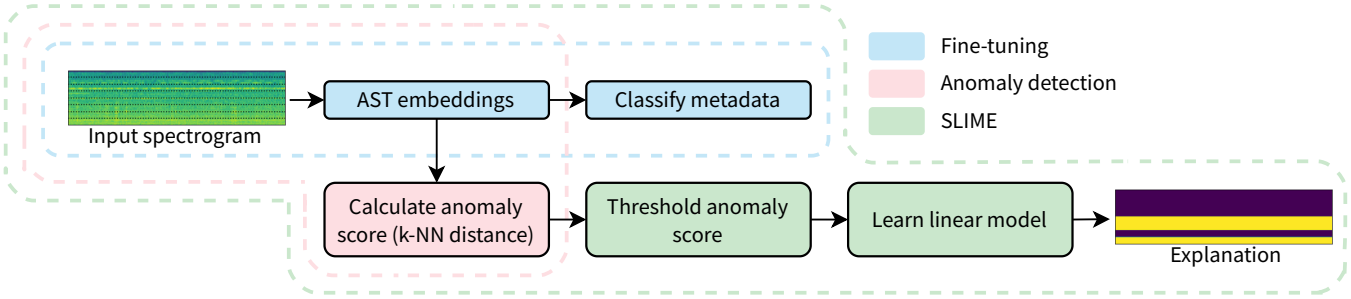


Figure 1: Schematic of the experimental workflow. The different coloured backgrounds correspond to the various stages. For fine-tuning, the AST is tasked with classifying metadata (the correct subset and section). For anomaly detection, the metadata and corresponding classification layer are disregarded. Instead, the embedding at the pre-logits layer of the fine-tuned model are extracted and used to train a k-NN. For SLIME, the input spectrograms are perturbed by segmenting via the frequency axis (denoted by the dotted lines in the spectrogram). The predicted distances produced by the trained k-NN are converted to a classification decision by using the equal error rate as a threshold. SLIME then approximates the classifier locally with an interpretable linear model to produce a heatmap explanation.

containing the first three sections and an evaluation set consisting of the latter three sections. The test split contains the same number of samples for both domains: about 100 inliers and 100 anomalies per section.

2.2. Anomaly detection model

Table 1: Mean AUC scores (%) across all six subsections (source and target) by anomaly detection model. **Bold** denotes the best result.

	Fan	Gearbox	Pump	Slider	ToyCar	ToyTrain	Valve
Autoencoder (Baseline)	64.0	66.8	63.7	69.2	63.2	63.0	53.7
MobileNetV2 (Baseline)	64.7	68.2	64.2	62.6	60.0	59.2	57.1
AST with Mel spectrogram	**74.5**	73.9	69.3	**76.5**	75.9	64.9	**76.4**
AST with STFT spectrogram	71.6	**81.0**	**73.2**	75.5	**78.9**	**65.0**	75.1

We use the same anomaly detection method for our experiments. Namely, we extract features from fine-tuned Audio Spectrogram Transformers (ASTs, 87m parameters) [12] and feed these to a shallow anomaly detection model. We choose to use ASTs as Transformers have demonstrated good anomaly detection performance in other domains [13, 14, 15]. Based on the success of using metadata in the DCASE 2021 submissions for learning representations, we fine-tune ASTs to classify both the machine and section (resulting in a 42-dimensional classification layer) using a cross-entropy loss. We choose to include all seven machines in the fine-tuning stage instead of fine-tuning an AST per machine, as we found that this improved anomaly detection performance. We choose cross-entropy instead of angular losses to reduce the number of hyperparameters that need to be tuned. Starting from ASTs pre-trained on AudioSet [16], we trained our models for a maximum of 10 epochs using the Adam optimiser and a learning rate of $1e-5$. Our fine-tuned model achieved a classification accuracy of 94%.

As an initial experiment to examine the importance of high frequencies, we use both STFT spectrograms and Mel spectrograms (128 bands) as input. We use Torchaudio [17] to compute the input features, using a frame length of 25ms and a hop size of 10ms. In line with the original AST implementation, we also normalise the spectrograms across both the time and frequency axes so that the dataset mean and standard deviation are 0 and 0.5 respectively. After fine-tuning, we extract training features at the pre-logits layer, resulting in a 768-dimensional representation. We use the features to train a k-NN. For inference, we extract test features in the same manner and use the mean distance from a test datum to its nearest neighbours as the anomaly score. We set $k = 1$ after validation. We tried other methods such as maximum softmax probability [18], Mahalanobis distance and isolation forest [19] and found the nearest neighbour approach worked best.

Table 1 illustrates our initial results. Excluding *Fan*, the AST trained with STFT spectrograms either matches or exceeds the AST trained with Mel spectrograms. As Mel weighting loses spectral resolution at high frequencies, this substantiates the high-frequency hypothesis for the majority of machines.

2.3. Preliminaries: Is it appropriate to analyse frequencies?

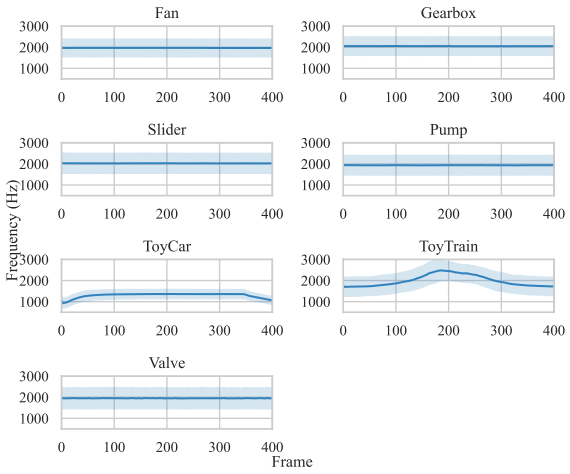


Figure 2: Mean spectral centroids (with standard deviations) across all inlier training samples, calculated using Librosa [20] with `n_fft`= 2048 and `hop_length`= 512.

To verify whether it is appropriate to examine how frequencies affect ASD, we checked whether the frequency distributions remain stable across different audio clips and times. We measured stability

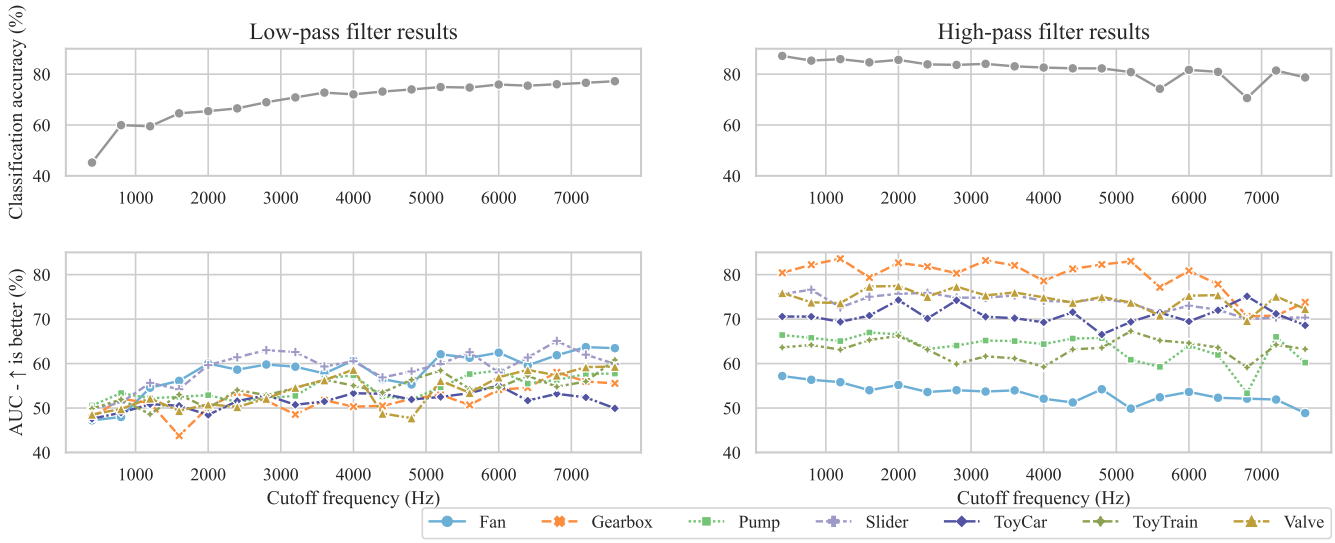


Figure 3: Top: change in classification accuracy after applying frequency filters on Mel spectrograms. Bottom: changes in mean AUC (across all six sections) after applying frequency filters. Applying low-pass filters (left) significantly affects fine-tuning classification accuracy and downstream ASD performance. Conversely, referring to Table 1, classification performance and ASD is remains stable after applying high-pass filtering (right). This indicates low-frequencies potentially obscure discriminative features. The same trends are observed when using STFT spectrograms as input.

by computing the average spectral centroids per frame in the training split for each machine. The spectral centroid is stable across time for most datasets (Figure 2), suggesting it is appropriate to examine the aggregate frequency features across time. We observe some variations to this in *ToyCar* and *ToyTrain*. For these machines, the spectral centroids are stable for most of the clips but dip at the start and end. We speculate this may be due to the 10-second clips being collected in ToyADMOS2 in a different systematic way.

2.4. Adapting local explanations to one-class detectors

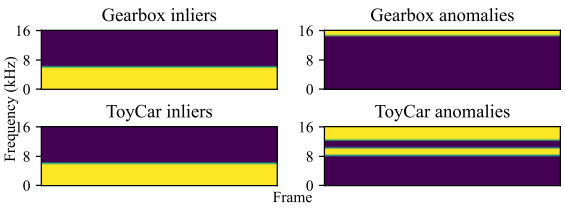


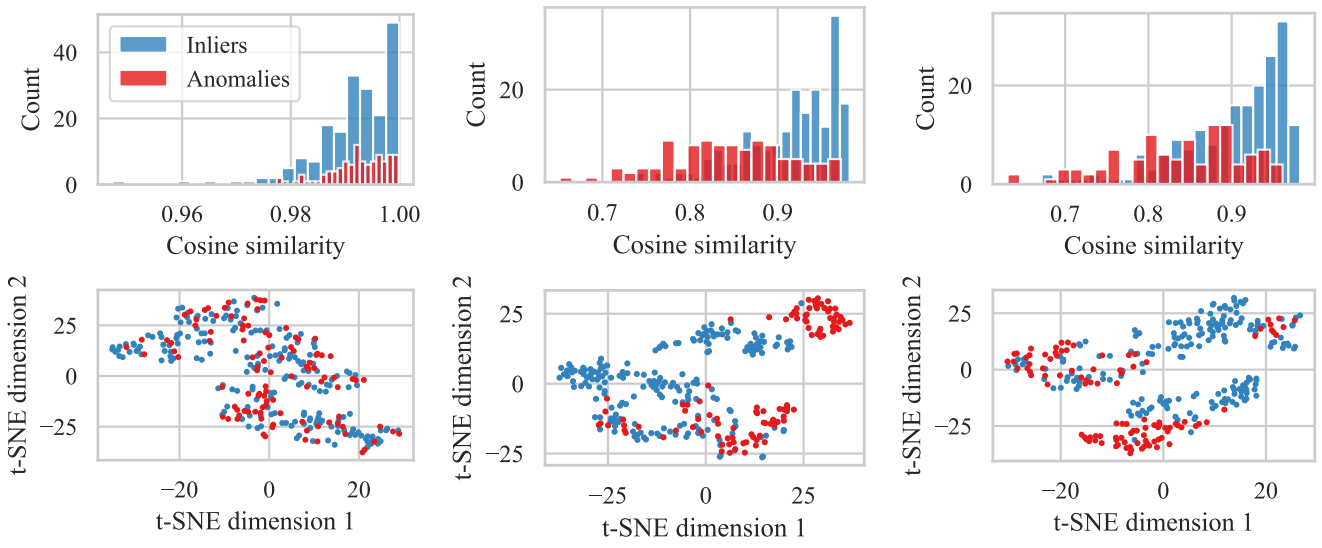
Figure 4: Median heatmaps generated by SLIME on Source Section 0, segmented by subset and condition {inlier, anomaly} . Yellow regions indicate frequency ranges that are more important for correct decisions while purple regions do not contribute to the ASD decision.

The differences in ASD performance using STFT spectrograms instead of Mel spectrograms in subsection 2.2 indicates that generally, maintaining spectral resolution at high frequencies is helpful. However, these AUC scores only give us a global insight into a detector's behaviour and do not account for differences at a section level. Sound local interpretable model-agnostic explanations (SLIME) [9] can provide a localised understanding. SLIME generates synthetic samples by firstly segmenting input spectrograms into fixed components in the time or frequency axis. It then randomly occludes input components using the segments and observes the changes in a model's behaviour by approximating the output using a linear model. The changes in a model's decision are visualised in a heatmap. In particular, the heatmaps depict regions contributing or detracting from the prediction. However, SLIME is typically used to interpret classifiers with distinct decision boundaries. Such boundaries are not necessarily present in anomaly detectors. Instead, a detector's output often measures the distance from the inlier training distribution. Examples of these types of anomaly detectors include Mahalanobis distance, k-nearest neighbours and one-class support vector machines.

To adapt SLIME to work for distance-based ASD, we convert the distances to labels $L = \{0 = \text{inlier}, 1 = \text{anomaly}\}$ using the equal error rate (EER) as the threshold. EER indicates the distance where the false-positive rate equals the false-negative rate. We choose EER as the threshold because it summarises overall ASD performance. Given λ as the value at which the EER occurs, and s_i as the score output by the anomaly detector for a test datum x_i, this decision can be can be expressed as follows:

$$L = \mathbb{1}[s_i \geq \lambda] \tag{1}$$

The SLIME region of Figure 1 (denoted by green dashes) illustrates our workflow. We run adapted SLIME to conduct a post-hoc evaluation on the test splits and by section. We segment the spectrograms by frequency as section 2.3 indicates the frequency distributions are relatively stable across time for the subsets. The time domain is not in scope for our experiments. As SLIME's outputs are sensitive to its training hyperparameters [9, 21], we divide the input spectrograms into eight fixed interpretable components and set the number of synthesised samples to 1,000. Figure 4 shows median heatmaps for *Gearbox* and *ToyCar*, split by condi-



(a) 400Hz low-pass filter. AUC: 46.6%

(b) No filters applied. AUC: 86.7%

(c) 400Hz high-pass filter. AUC: 93.0%

Figure 5: Qualitative visualisations of *Gearbox* Source Section 1. Top: Histograms of normalised (cosine) k-nearest neighbour similarities. Bottom: t-SNE scatter plots. Applying a low-pass filter (left) decreases separability between inliers and anomalies while applying a high-pass filter increases separability and improves inlier generalisation, as evidenced through the change in scoring distribution.

tion. The heatmaps show that occluding the low-frequency regions makes inliers appear more anomalous, leading to more incorrect decisions. Correspondingly, occluding the high-frequency regions makes anomalies appear more benign. We observe similar patterns across sections and machines, and on the STFT spectrograms. This behaviour appears to validate the high-frequency hypothesis.

2.5. Re-training the detectors with filtered audio

Our final step to empirically validate the high-frequency hypothesis is to apply low and high-pass filters to the input data, repeat fine-tuning on the ASTs and redo anomaly detection. If high frequencies are important, then we would expect low-pass filters to decrease ASD. We filtered the input spectrograms at both the fine-tuning and anomaly detection stages to prevent discrepancies between the training and test distributions [22]. This intervention removes the possibility that any changes in performance are caused by distribution shifts. In line with previous studies that looked at frequency artefacts in the speech domain [23], we vary the cutoff frequencies between 400 Hz and 8 kHz in increments of 400 Hz for both the low-pass and high-pass filters. We applied the filters directly on the input audio before converting to spectrograms.

Figure 3 depicts our Mel spectrogram results. The low-pass results indicate removing high-frequency information decreases separability between classes, causing fine-tuning classification accuracy and ASD performance across subsets to drop significantly. Separability between classes is still good after applying high-pass filters, and so downstream ASD performance is retained or even improves. We note that classification accuracy drops from the baseline fine-tuning accuracy of 94% (Section 2.2), which suggests separability between classes is an important but not necessary component for ASD. Overall, our results illustrate that high frequencies are helpful for ASD, except for *Fan*, where informative frequencies likely lie in sub-bands. Although *Fan* exhibits similar trends to the other machines, applying high-pass filters causes performance to drop significantly compared to the benchmark AST. We show Figure 3's results qualitatively by visualising the change in normalised k-NN (cosine) similarities and t-SNE [24] on the 768-dimensional learnt embeddings. Figure 5 illustrates an example on *Gearbox*. Aggressive low-pass filtering decreases separability between inliers and anomalies while high-pass filtering increases the dissimilarity. We also visualise a similar phenomenon for the other machines.

3. CONCLUSION AND FUTURE WORK

We test if higher frequencies aid ASD in three ways: (1) by varying the input spectrograms used to train the anomaly detector, (2) evaluating the heatmaps produced by SLIME, and (3) re-training the anomaly detectors after filtering frequencies of differing magnitudes. Our results corroborate the high-frequency hypothesis and suggest high-pass filtering to remove noisy information is a simple pre-processing step to boost performance. The scope of adapted SLIME in our experiments was isolated to frequency segmentations. Some avenues for future work include analysing changes in ASD after segmenting input spectrograms by time, investigating SLIME outputs on alternative anomaly detection architectures, or using more sophisticated thresholds in place of EER. Furthermore, the exceptional results on *Fan* suggest that some anomalies may manifest in sub-band frequencies. Future work could extend the re-training approach by using band-pass filters, which could provide a more fine-grained analysis of anomalous artefacts.

4. ACKNOWLEDGMENT

This project made use of time on Tier 2 HPC facility JADE2, funded by EPSRC under grant EP/T022205/1.

5. REFERENCES

[1] http://dcase.community/.

[2] Y. Kawaguchi, K. Imoto, Y. Koizumi, N. Harada, D. Niizumi, K. Dohi, R. Tanabe, H. Purohit, and T. Endo, "Description and discussion on dcase 2021 challenge task 2: Unsupervised anomalous detection for machine condition monitoring under domain shifted conditions," in *Proceedings of the 6th Detection and Classification of Acoustic Scenes and Events 2021 Workshop (DCASE2021)*, Barcelona, Spain, November 2021, pp. 186–190.

[3] J. Lopez, G. Stemmer, and P. Lopez-Meyer, "Ensemble of complementary anomaly detectors under domain shifted conditions," DCASE2021 Challenge, Tech. Rep., July 2021.

[4] K. Morita, T. Yano, and K. Tran, "Anomalous sound detection using cnn-based features by self supervised learning," DCASE2021 Challenge, Tech. Rep., July 2021.

[5] J. Frank, T. Eisenhofer, L. Schönherr, A. Fischer, D. Kolossa, and T. Holz, "Leveraging frequency analysis for deep fake image recognition," in *International Conference on Machine Learning (ICML)*, 2020, pp. 3247–3258.

[6] D. Hendrycks, S. Basart, N. Mu, S. Kadavath, F. Wang, E. Dorundo, R. Desai, T. Zhu, S. Parajuli, M. Guo, *et al.*, "The many faces of robustness: A critical analysis of out-of-distribution generalization," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 8340–8349.

[7] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "Specaugment: A simple data augmentation method for automatic speech recognition," *Proc. Interspeech 2019*, pp. 2613–2617, 2019.

[8] Y. Liu, J. Guan, Q. Zhu, and W. Wang, "Anomalous sound detection using spectral-temporal information fusion," in *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022, pp. 816–820.

[9] S. Mishra, B. L. Sturm, and S. Dixon, "Local interpretable model-agnostic explanations for music content analysis," in *Proceedings of the 18th International Society for Music Information Retrieval Conference*. ISMIR, Oct 2017, p. 537–543.

[10] N. Harada, D. Niizumi, D. Takeuchi, Y. Ohishi, M. Yasuda, and S. Saito, "ToyADMOS2: Another dataset of miniature-machine operating sounds for anomalous sound detection under domain shift conditions," *arXiv preprint arXiv:2106.02369*, 2021.

[11] R. Tanabe, H. Purohit, K. Dohi, T. Endo, Y. Nikaido, T. Nakamura, and Y. Kawaguchi, "MIMII DUE: Sound dataset for malfunctioning industrial machine investigation and inspection with domain shifts due to changes in operational and environmental conditions," *arXiv e-prints: 2006.05822*, 2021.

[12] Y. Gong, Y.-A. Chung, and J. Glass, "AST: Audio Spectrogram Transformer," in *Proceedings of Interspeech 2021*, 2021, pp. 571–575.

[13] D. Hendrycks, X. Liu, E. Wallace, A. Dziedzic, R. Krishnan, and D. Song, "Pretrained transformers improve out-of-distribution robustness," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, July 2020, pp. 2744–2751.

[14] S. Fort, J. Ren, and B. Lakshminarayanan, "Exploring the limits of out-of-distribution detection," in *Advances in Neural Information Processing Systems*, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, Eds., 2021.

[15] K. T. Mai, T. Davies, and L. D. Griffin, "Self-supervised losses for one-class textual anomaly detection," *arXiv preprint arXiv:2106.02369*, 2022.

[16] J. F. Gemmeke, D. P. W. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio set: An ontology and human-labeled dataset for audio events," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017, pp. 776–780.

[17] Y.-Y. Yang, M. Hira, Z. Ni, A. Astafurov, C. Chen, C. Puhrsch, D. Pollack, D. Genzel, D. Greenberg, E. Z. Yang, J. Lian, J. Hwang, J. Chen, P. Goldsborough, S. Narenthiran, S. Watanabe, S. Chintala, and V. Quenneville-Bélair, "Torchaudio: Building blocks for audio and speech processing," in *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022, pp. 6982–6986.

[18] D. Hendrycks and K. Gimpel, "A baseline for detecting misclassified and out-of-distribution examples in neural networks," in *5th International Conference on Learning Representations (ICLR)*, 2017.

[19] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in *2008 Eighth IEEE International Conference on Data Mining*, 2008, pp. 413–422.

[20] B. McFee, A. Metsai, M. McVicar, S. Balke, C. Thomé, C. Raffel, F. Zalkow, A. Malek, Dana, K. Lee, O. Nieto, D. Ellis, J. Mason, E. Battenberg, S. Seyfarth, R. Yamamoto, viktorandreevichmorozov, K. Choi, J. Moore, R. Bittner, S. Hidaka, Z. Wei, nullmightybofo, D. Hereñú, F.-R. Stöter, P. Friesch, A. Weiss, M. Vollrath, T. Kim, and Thassilo, "librosa/librosa: 0.8.1rc2," May 2021.

[21] S. Mishra, E. Benetos, B. L. Sturm, and S. Dixon, "Reliable local explanations for machine listening," in *International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2020.

[22] S. Hooker, D. Erhan, P.-J. Kindermans, and B. Kim, "A benchmark for interpretability methods in deep neural networks," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. Curran Associates, Inc., 2019, pp. 9737–9748.

[23] H. Tak, J. Patino, A. Nautsch, N. Evans, and M. Todisco, "An explainability study of the constant q cepstral coefficient spoofing countermeasure for automatic speaker verification," in *Proc. Odyssey 2020 The Speaker and Language Recognition Workshop*, 2020, pp. 333–340.

[24] L. van der Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, no. 86, pp. 2579–2605, 2008.

A DEVICE CLASSIFICATION-AIDED MULTI-TASK FRAMEWORK FOR LOW-COMPLEXITY ACOUSTIC SCENE CLASSIFICATION

Rohith Mars and Rohan Kumar Das

Fortemedia Singapore, Singapore
{rohithmars, rohankd}@fortemedia.com

ABSTRACT

In this paper, we investigate the use of a multi-task learning framework to address the DCASE 2022 Task 1 on Low-complexity Acoustic Scene Classification (ASC). Specifically, we employ classification of the recording devices as an additional task to improve the performance of the ASC task. Both these tasks utilize our proposed convolutional neural network with a shared layer along with strided and separable convolution operations designed to comply with the model parameter and computational constraints imposed by Task 1 organizers. We also explore the use of various data augmentation techniques to improve the generalization of the model. Evaluations on the development dataset show that our proposed ASC system consisting of 127.2k parameters with 27.5 million multiply and accumulate operations provides a significant improvement on overall log-loss and accuracy over the baseline system.

Index Terms— acoustic scene classification, convolutional neural networks, data augmentation, multi-task learning, quantization aware training.

1. INTRODUCTION

The objective of an Acoustic Scene Classification (ASC) system is to classify a given audio recording of a certain temporal duration into one of the pre-defined acoustic scenes. As a sub-field of computational auditory scene analysis (CASA) [1, 2], ASC has received much attention as an important and challenging research topic. It also finds several applications in consumer devices, which incorporates contextual-aware computing, such as smart-wearables, hearables as well as surveillance [3, 4].

The Detection and Classification of Acoustic Scenes and Events (DCASE) challenge[1], has played an important role in providing standard datasets for setting algorithmic benchmarks and to spearhead research on ASC. Over the years, the ASC task in DCASE challenge series has witnessed several modifications by introducing additional recording devices and constraints for model complexity. In DCASE 2021 Task 1, the model size of the proposed solutions were constrained to 128 kB for non-zero parameters, with no constraints set on the multiply and accumulate (MAC) operations. In addition, the length of audio clip for inference was set as 10 seconds for all the past editions.

A majority of the proposed systems for multi-device ASC task utilize a single task learning (STL) framework with log-Mel spectrogram as input feature to a convolutional neural network (CNN)-based model, which is trained to predict the acoustic scene [5, 6]. To improve the generalization of the model to unseen devices and recordings from unseen locations, many techniques including spectrum correction and frequency instance normalization were proposed [6, 7]. In addition to these techniques, various data augmentation strategies have also been employed extensively in the past to address the same [8, 9]. To meet the model size requirements, approaches such as model compression with pruning and use of knowledge distillation have also been well-explored [10, 11].

The latest DCASE 2022 Task 1 [12] is an extension of the previous editions of DCASE ASC challenges, with an objective to design an ASC system with low computation complexity and sufficiently less number of parameters such that it could be deployed for inference on an edge-device. Specifically, the maximum number of the model parameters is set to 128k (including the zero-valued ones) and the variable type is fixed to 8-bit integer (INT8). Further, the maximum number of MAC operations for each inference is limited to 30 MMAC (million multiply-accumulate operations), while the length of the audio for inference is fixed to 1 second. In addition to these system complexity constraints, the proposed ASC system is expected to be robust across multiple recording devices and locations similar to the previous editions.

We address this ASC task by utilizing a multi-task learning (MTL) framework, where the classification of recording devices is used as an additional task to aid the ASC task. Our proposed CNN model architecture is designed such that it satisfies the system complexity requirements in terms of MAC operations and total number of parameters. To further improve generalization ability of the model to unseen devices, we employ multiple data augmentation techniques. The quantization-aware-training (QAT) and TFLite tools are then utilized to convert the proposed CNN model to INT8 representation for performing final inference and comparisons to other competing systems along with the challenge baseline.

In the following sections, we first describe the proposed MTL framework in Section 2. The details of the experimental setup and results with analysis are reported in Section 3 and Section 4, respectively. Finally, the conclusions are presented in Section 5.

2. DEVICE CLASSIFICATION-AIDED ASC

A majority of the systems developed previously to address ASC using multiple devices have employed an STL framework. In comparison, the use of MTL frameworks for ASC task is less explored. Generally, an MTL framework consists of an encoder block with shared layers, which are subsequently connected to multiple branches with task-specific layers. A solution based on MTL framework was recently proposed for the ASC task of DCASE. The authors of [13] used an additional task of classifying the audio clip to 3 broader acoustic scenes, in addition to the given task of classifying the audio clip to 10 acoustic scenes for joint training. The 3 broader

[1] https://dcase.community/challenge2022/

acoustic scenes are higher-level abstractions of the 10 scenes, which make these tasks highly related to each other.

In DCASE 2022 Task 1 on low-complexity ASC, the development dataset consists of audio samples from 3 real-devices (A, B, C) and 6 simulated devices (S1-S6). In our proposed MTL framework, we leverage recording device information by classifying recording devices corresponding to the given audio clip as a task, in addition to the ASC task. Specifically, we classify the given audio clip to be recorded using one among the seen real-devices (A, B, C) and simulated/unseen devices. All the simulated devices (S1-S6) and unseen devices are grouped to 1 device-class, thereby forming a total of 4 device-classes. By enabling the shared layers in the encoder block to learn complementary features to classify the recording device corresponding to a given audio clip, we envisage that these layers of the network can learn low-level features which could improve the performance of the model for multi-device ASC task.

One of the previous works, explored the use of device classification for ASC as well [14]. However, for joint training using MTL, the one-hot encoding representation of the predicted device is expanded and converted as a feature map to the multiple CNN layers in the ASC branch. Even though the use of such device-information conditioned MTL training shows improved performance compared to STL framework, such an approach would require the device-classification branch to be included during the inference stage as well. As such, it would increase the number of parameters and MAC operations of the inference model for the given ASC task. To ensure that the feature maps of the shared layers can leverage similar information without increasing the computation complexity, instead of utilizing the predicted device as a feature map, we combine the loss corresponding to the device classification task with the loss of the ASC task for joint training.

We use a combined loss function, which is the weighted loss function from the ASC branch and the device classification branch to perform the joint training of both these tasks. It can be expressed mathematically as

$$\mathcal{L}_{\text{MTL}} = \beta \times \mathcal{L}_{\text{device}} + (1 - \beta) \times \mathcal{L}_{\text{ASC}}, \quad (1)$$

where β is the trade-off parameter that controls the weighted loss. $\mathcal{L}_{\text{device}}$ and $\mathcal{L}_{\text{ASC}}$ corresponds to the 4-device classification loss and the 10-scene classification loss, respectively. Setting $\beta = 0$ is similar to the STL framework.

The use of MTL framework with joint training utilizing the weighted loss in Eq (1) for the given ASC task has the additional advantage that once the MTL-based model is trained, the device classification branch can be removed from the model architecture. During the inference stage, only the 10-scene classification branch is utilized. Therefore, the number of parameters and MAC operations remains the same as that of the STL framework.

2.1. CNN model architecture

Our proposed MTL framework based on CNN architecture is shown in Figure 1. It consists of two branches, i.e., a branch corresponding to the 10-scene classification task and another branch corresponding to the 4-device classification task. The 10-scene classification branch consists of a combination of separable and strided convolutions. We use $L = 5$ convolution layers with number of filters set as $(150, 140, 160, 180, 220)$. It is noted that separable convolutions are used for all the layers except the first convolution layer. In addition, a stride of 2 is used for all convolutional layers except the last convolution layer. Strided convolutions are applied uniformly

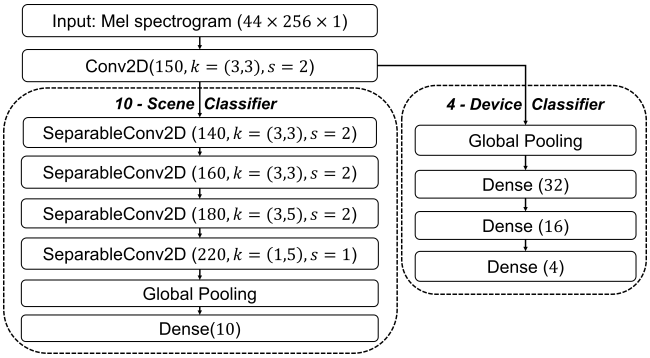


Figure 1: The proposed MTL framework with separate branches for 10-scene classification and 4-device classification task. Conv2D and SeparableConv2D $(n, (p \times q), s)$ represents 2D convolution and separable convolution operation with n filters of kernel size $p \times q$ with a stride of s.

across both frequency and time dimensions. The kernel size for the first three convolution layers is chosen as (3×3), whereas we consider a kernel size of (3×5) and (1×5) for the fourth and fifth convolution layers, respectively. Finally, global pooling operation is performed to gather all the components from the last convolution layer. The output layer consists of a dense layer with 10 units corresponding to the number of acoustic scene classes which undergoes softmax$(\cdot)$ operation to obtain the scene prediction probabilities.

For the 4-device classification branch, the output of the first convolution layer undergoes a global pooling operation and is connected to two dense layers of 32 and 16 units each. The output layer consists of a dense layer with softmax$(\cdot)$ operation with 4 units corresponding to the device classes. Rectified linear unit (ReLU) is chosen as the activation function and a weight decay of $1\text{e}{-5}$ is applied for all convolution layers. We also use dropout of 0.5 after the $L = 2, 4, 5$ convolution layers and a dropout of 0.3 between the dense layers of the 4-device classification branch. The network parameters for both the branches are determined empirically.

2.2. INT8 Quantization

As part of the Task 1 constraints, it is required that the model and the input data used for inference should use INT8 variable type. To minimize the drop in performance after the quantization step, we first perform quantization-aware-training (QAT) utilizing TensorFlow framework with a subset of the training data [15]. After the model is fine-tuned using QAT, it is converted to INT8 TFLite model. The number of parameters and the MAC operations of the TFlite model is computed by using the scripts in Nessi[2] toolkit, which is provided by the Task 1 challenge organizers. The proposed TFLite model has $127,236$ parameters with 27.53 MMAC operations, which satisfies the model size and complexity constraints.

2.3. Data augmentation

To ensure that the proposed model is generalized to recordings using an unseen device or a recording from an unseen location, multiple data augmentation techniques that are well-explored in the past are used [16, 17]. In addition, we also explore the use of AugMix [18]

[2] https://github.com/AlbertoAncilotto/NeSsi

Table 1: Performance comparison of the STL and the proposed MTL framework for various β. The device-wise log loss is shown in columns A-S6. The best results are shown in bold.

Framework	Average Log Loss	Average Accuracy (%)	A	B	C	S1	S2	S3	S4	S5	S6
STL	1.333 ± 0.012	51.25 ± 0.33	0.98	1.24	**1.11**	1.37	1.35	1.35	1.50	1.45	1.61
MTL, $\beta = 0.1$	1.327 ± 0.009	51.65 ± 0.31	0.99	1.24	**1.11**	1.36	1.35	**1.33**	1.48	1.45	1.61
MTL, $\beta = 0.2$	$\mathbf{1.319 \pm 0.017}$	$\mathbf{51.98 \pm 0.55}$	**0.97**	**1.22**	**1.11**	1.36	**1.34**	1.34	1.48	**1.43**	**1.58**
MTL, $\beta = 0.3$	1.323 ± 0.014	51.31 ± 0.55	**0.97**	**1.22**	**1.11**	**1.35**	**1.34**	1.34	**1.47**	1.46	1.61
MTL, $\beta = 0.4$	1.347 ± 0.014	50.53 ± 0.54	0.99	1.25	1.13	1.39	1.37	1.37	1.49	1.47	1.63
MTL, $\beta = 0.5$	1.335 ± 0.010	51.31 ± 0.40	0.98	1.25	1.12	1.38	1.37	1.36	**1.47**	1.45	1.59

and RawBoost [19] technique that have not yet been extensively used for ASC task. Since device classification is utilized in the MTL framework, it is important to assign the device labels according to the augmentation technique being employed. The augmentation techniques utilized are summarised below.

- **Time stretch (TS) :** The audio sample is either slowed down or sped up without altering the pitch by a factor randomly chosen from the uniform distribution $[0.8, 1.2]$.
- **Time shift (TSH) :** A random start time is chosen from the 1 second duration of the given audio clip to obtain two audio segments. The augmented audio clip is the appended audio segments shifted in time.
- **Block mixing (BM) :** For a given audio clip corresponding to a particular device and acoustic scene, an additional audio clip recorded with the same device and belonging to the same acoustic scene is selected randomly. The augmented audio clip is the weighted average of the given audio clip and the additional audio clip.
- **AugMix :** Similar to the AugMix strategy used for image data augmentation, a given audio clip undergoes multiple chains of transformations using the above data augmentation steps (time stretch, time shift, block mix). At the final stage, the original audio clip is combined with the transform-chain augmented audio clip to obtain the augmented audio sample. We set the default values of maximum number and depth of the transform chain to 3 with the α parameter of both Dirichlet and Beta distribution set to 1.0 as in [18].
- **RawBoost :** The recently proposed data augmentation technique RawBoost for anti-spoofing is considered for exploration in the given ASC task. In RawBoost, multiple notch filters with randomly chosen center frequencies and bandwidths are used to design an FIR filter. The use of FIR filter on the given audio clip, introduces variations on the audio spectrum, which could improve generalization of the model for unseen device. We set the similar filter design configurations as used in [19].

Among the above data augmentation techniques, time stretch, time shift, block mixing and AugMix does not change the frequency characteristics of the given audio clip. Therefore, they are device-label invariant techniques. However, the RawBoost technique applies FIR filtering on the given audio clip which modifies its frequency characteristics. As such, it is a device-label variant technique and these clips are labeled as belonging to the simulated and unseen device class. Next, we discuss the experimental setup for the studies conducted in this work.

3. EXPERIMENTAL SETUP

We use the TAU Urban Acoustic Scenes 2022 Mobile, development dataset provided as part of the challenge for the studies in this work [20]. This development set consists of an official training and validation split with audio clips of 1 second in duration sampled at 44.1 kHz. The clips correspond to audio scene recordings from 10 cities using 9 devices: 3 real devices (A, B, C) and 6 simulated devices (S1-S6). The simulated device recordings (S1-S6) are obtained by applying impulse responses of real devices and additional dynamic range compression on recordings from device A. There are $102,150$ clips from device A, while for devices (B, C, S1-S3) there are $\approx 7,500$ clips each. It should be noted that the audio clips corresponding to devices S4-S6 does not appear in the training split and is only present in the validation split. The 10 audio scenes to be classified are namely {"Airport", "Indoor shopping mall", "Metro station", "Pedestrian street", "Public square", "Street with medium level of traffic", "Travelling by a tram", "Travelling by a bus", "Travelling by an underground metro" & "Urban park"}.

We do not utilize any external data or pre-trained models for training our proposed system. For each audio clip, we compute the short-time Fourier transform (STFT) of the signal by using a frame length of 2048 samples and hop length of 1024 samples. After computing the STFT, the corresponding log Mel-spectrogram is computed with 256 Mel-bins. We then obtain a time-frequency (TF) representation of each audio clip with dimension 44×256, which is provided as the input feature to the proposed MTL framework.

During training, the optimization is performed using the Adam optimizer [21], with an initial learning rate of 0.001 and a maximum epoch of 200 with a batch size of 32 samples. The learning rate is reduced by a factor of 0.1 if the validation loss does not decrease after 5 epochs. Early stopping method is used to stop the training if the validation loss does not decrease after 10 epochs. The categorical focal loss [22] is chosen as the loss function for both classification tasks. For all the evaluations, we utilize the validation split from the development set. The multi-class cross-entropy (log loss) is used as the primary metric for evaluating system performance along with average of the class-wise accuracy as a secondary measure. The performances are evaluated over 5 trials and the average results are reported.

4. RESULTS AND ANALYSIS

This section discusses the results and their analysis conducted for various studies in this work. The impact of proposed MTL framework and data augmentation on ASC and comparison to other systems are reported in the following subsections.

Table 2: Performance comparison for various data augmentation approaches and their combination. The best results are shown in bold.

Method	Avg. Log Loss	Avg. Acc. (%)	A	B	C	S1	S2	S3	S4	S5	S6
No Data Augmentation	1.319 ± 0.017	51.98 ± 0.55	0.97	1.22	1.11	1.36	1.34	1.34	1.48	1.43	1.58
TS + TSH + BM (1)	1.314 ± 0.007	52.08 ± 0.33	0.96	1.21	1.13	1.36	1.34	1.32	1.46	1.44	1.56
AugMix (2)	1.309 ± 0.014	52.45 ± 0.56	0.96	1.23	1.12	1.35	1.33	1.33	1.43	1.43	1.56
RawBoost (3)	1.298 ± 0.010	52.22 ± 0.71	0.99	1.18	1.10	1.33	1.33	1.31	1.41	1.41	1.58
Combined (1) + (2) + (3)	$\mathbf{1.265 \pm 0.009}$	$\mathbf{53.59 \pm 0.43}$	**0.94**	**1.17**	**1.07**	**1.29**	**1.32**	**1.28**	**1.39**	**1.40**	**1.50**

4.1. Performance evaluation of STL and MTL framework

We begin our experiments by evaluating the performance of the ASC system using the STL and MTL framework. For both these frameworks, the samples from the training split of the development dataset are used without applying any specific data augmentation technique. For MTL framework, the trade-off parameter, β is varied from 0.1 to 0.5. Since our primary task is the 10-scene classification, we set the maximum value of β as 0.5, which corresponds to equal weight to the loss of both tasks. The average log loss and the average accuracy obtained using the STL and MTL framework for various β are shown in Table 1. It can be seen that the MTL framework with $\beta = 0.2$ achieves the best performance with an average log loss of 1.319 and an average accuracy of 51.98%. In comparison, the STL framework achieves an average log loss of 1.333 and an average accuracy of 51.25%. From the results belonging to device-wise log loss, it can be seen that the use of MTL framework achieves comparatively lower log loss for most of the seen and unseen devices. In addition, we note that the MTL framework with $\beta = 0.2$ achieves an average accuracy of $\approx 97\%$ on the 4-device classification task. This indicates that enabling the shared layer to learn device-specific low-level features helps to improve the ASC task. Since the MTL framework with $\beta = 0.2$ achieves the best ASC performance, we use the same for the rest of our experiments.

4.2. Performance evaluation of data augmentation techniques

In this set of experiments, we evaluate the performance of the proposed MTL framework using various data augmentation methods presented in Section 2.3. A given data augmentation technique is applied separately on all the samples in the training dataset for comparing its effect on the ASC task. We apply the time stretch, time shift and block mixing with equal probability and combine them as one data augmentation technique (TS + TSH + BM). The average log loss, average accuracy as well as the device-wise log loss obtained using none of the data augmentation techniques and applying each of them separately is shown in Table 3. It can be seen that each augmentation technique helps to improve the average log loss and accuracy. Since the objective of RawBoost technique is to generate unseen/simulated devices, its performance especially on devices S4 and S5 is improved, while performance on device A is degraded. Subsequently, all the data augmentation techniques are combined to train our final model. It achieves an average log loss and average accuracy of 1.265 and 53.59% respectively, which are better than those obtained with any of the individual data augmentation methods as well as those without any augmentation.

4.3. Comparison to other systems

We now compare the performance of the proposed MTL-based ASC system on the development set with the well performing systems submitted to the DCASE Task 1 challenge on low-complexity ASC as well as the challenge baseline. To this extent, we perform INT8 quantization on our best performing model as described in Section 2.2 for inference. The INT8 quantized model achieves an average log loss 1.273 and an average accuracy of 53.5%. The reported performances of other systems and the baseline, sorted with reference to average log loss are shown in Table 3. It can be seen that we achieve a comparable performance with other systems with a significant improvement over the baseline system. We note that most of the proposed systems are based on STL frameworks, which utilize knowledge distillation technique for model design. However, Sugahara *et al.* [23] uses an MTL framework with the 3 broader ASC as the additional task. It is also noted that the baseline system uses a 3-layer CNN using log Mel-spectrograms with $46,512$ parameters and 29.23 MMACS [12].

Table 3: Performance comparison of the proposed MTL-based system with other well performing systems on the development set of DCASE 2022 Task 1.

System	Avg. Log Loss	Avg. Accuracy (%)
Lee *et al.* [24]	0.835	70.1
Anastácio *et al.* [25]	1.103	60.5
Schmid *et al.* [26]	1.139	58.0
Sugahara *et al.* [23]	1.182	56.5
Kim *et al.* [27]	1.259	54.0
Proposed MTL **(Ours)**	1.273	53.5
Morocutti *et al.* [28]	1.288	52.7
Xin *et al.* [29]	1.295	60.3
Yu *et al.* [30]	1.305	51.7
Shao *et al.* [31]	1.360	54.1
Baseline [12]	1.575	42.9

5. CONCLUSIONS

In this paper, we explore the use of an MTL framework with the device classification task as an additional task to address the DCASE 2022 Task 1 on low-complexity ASC. The proposed CNN model is designed such that it meets the Task 1 model complexity constraints. We also explore the use of various data augmentation techniques to improve the generalization of the model. Evaluations on the development set show that the use of MTL framework along with the various data augmentation techniques help to improve the performance of the ASC task.

6. REFERENCES

[1] D. Wang and G. J. Brown, *Computational auditory scene analysis: Principles, algorithms, and applications*. Wiley-IEEE press, 2006.

[2] T. Virtanen, M. D. Plumbley, and D. Ellis, *Computational analysis of sound scenes and events*. Springer, 2018.

[3] L. Sicong, Z. Zimu, D. Junzhao, S. Longfei, J. Han, and X. Wang, "UbiEar: Bringing location-independent sound awareness to the hard-of-hearing people with smartphones," *in Proc. ACM Interactive, Mobile, Wearable and Ubiquitous Technologies (IMWUT)*, vol. 1, no. 2, pp. 1–21, 2017.

[4] R. Radhakrishnan, A. Divakaran, and A. Smaragdis, "Audio analysis for surveillance applications," in *Proc. IEEE Workshop Applicat. Signal Process. Audio Acoust.* IEEE, 2005, pp. 158–161.

[5] S. Suh, S. Park, Y. Jeong, and T. Lee, "Designing acoustic scene classification models with CNN variants," DCASE2020 Challenge, Tech. Rep., June 2020.

[6] B. Kim, S. Yang, J. Kim, and S. Chang, "Domain generalization on efficient acoustic scene classification using residual normalization," in *Proc. DCASE Workshop*, 2021, pp. 21–25.

[7] M. Kośmider, "Spectrum correction: Acoustic scene classification with mismatched recording devices," *in Proc. Interspeech*, pp. 4641–4645, 2020.

[8] C.-H. H. Yang, H. Hu, S. M. Siniscalchi, Q. Wang, W. Yuyang, X. Xia, Y. Zhao, Y. Wu, Y. Wang, J. Du, and C.-H. Lee, "A lottery ticket hypothesis framework for low-complexity device-robust neural acoustic scene classification," DCASE2021 Challenge, Tech. Rep., June 2021.

[9] H. Hee-Soo, J. Jee-weon, S. Hye-jin, and L. Bong-Jin, "Clova submission for the DCASE 2021 challenge: Acoustic scene classification using light architectures and device augmentation," DCASE2021 Challenge, Tech. Rep., June 2021.

[10] K. Koutini, S. Jan, and G. Widmer, "CPJKU submission to DCASE21: Cross-device audio scene classification with wide sparse frequency-damped CNNs," DCASE2021 Challenge, Tech. Rep., June 2021.

[11] N. Zhao, "Low-complexity acoustic scene classification using knowledge distillation and multiple classifiers," DCASE2021 Challenge, Tech. Rep., June 2021.

[12] I. Martín-Morató, F. Paissan, A. Ancilotto, T. Heittola, A. Mesaros, E. Farella, A. Brutti, and T. Virtanen, "Low-complexity acoustic scene classification in DCASE 2022 challenge," *arXiv preprint arXiv:2206.03835*, 2022.

[13] H.-j. Shim, J.-w. Jung, J.-h. Kim, and H.-J. Yu, "Attentive max feature map and joint training for acoustic scene classification," in *Proc. ICASSP*. IEEE, 2022, pp. 1036–1040.

[14] Z. Ren, Q. Kong, J. Han, M. D. Plumbley, and B. W. Schuller, "CAA-Net: Conditional atrous CNNs with attention for explainable device-robust acoustic scene classification," *IEEE Transactions on Multimedia*, vol. 23, pp. 4131–4142, 2020.

[15] R. Krishnamoorthi, "Quantizing deep convolutional networks for efficient inference: A whitepaper," *arXiv preprint arXiv:1806.08342*, 2018.

[16] J. Salamon and J. P. Bello, "Deep convolutional neural networks and data augmentation for environmental sound classification," *IEEE Signal processing letters*, vol. 24, no. 3, pp. 279–283, 2017.

[17] H. Hu, C.-H. H. Yang, X. Xia, X. Bai, X. Tang, Y. Wang, S. Niu, L. Chai, J. Li, H. Zhu, *et al.*, "A two-stage approach to device-robust acoustic scene classification," in *Proc. ICASSP*. IEEE, 2021, pp. 845–849.

[18] D. Hendrycks, N. Mu, E. D. Cubuk, B. Zoph, J. Gilmer, and B. Lakshminarayanan, "AugMix: A simple method to improve robustness and uncertainty under data shift," in *Proc. ICLR*, 2020.

[19] H. Tak, M. Kamble, J. Patino, M. Todisco, and N. Evans, "RawBoost: A raw data boosting and augmentation method applied to automatic speaker verification anti-spoofing," in *Proc. ICASSP*. IEEE, 2022, pp. 6382–6386.

[20] T. Heittola, A. Mesaros, and T. Virtanen, "Acoustic scene classification in DCASE 2020 challenge: generalization across devices and low complexity solutions," in *Proc. DCASE Workshop*, 2020, pp. 56–60.

[21] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[22] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proc. ICCV*, 2017, pp. 2980–2988.

[23] R. Sugahara, R. Sato, M. Osawa, Y. Yuno, and C. Haruta, "Self-ensemble with multi-task learning for low-complexity acoustic scene classification," DCASE2022 Challenge, Tech. Rep., June 2022.

[24] J.-H. Lee, J.-H. Choi, P. M. Byun, and J.-H. Chang, "HYU submission for the DCASE 2022: Efficient fine-tuning method using device-aware data-random-drop for device-imbalanced acoustic scene classification," DCASE2022 Challenge, Tech. Rep., June 2022.

[25] R. Anastácio, L. Ferreira, F. Mónica, and C. B. Luís, "Ai4edgept submission to DCASE 2022 low complexity acoustic scene classification task1," DCASE2022 Challenge, Tech. Rep., June 2022.

[26] F. Schmid, S. Masoudian, K. Koutini, and G. Widmer, "CP-JKU submission to DCASE22: Distilling knowledge for low-complexity convolutional neural networks from a patchout audio transformer," DCASE2022 Challenge, Tech. Rep., June 2022.

[27] T. Kim, G. Lee, and J. Park, "KT submission for the DCASE 2022 challenge: Modernized convolutional neural networks for acoustic scene classification," DCASE2022 Challenge, Tech. Rep., June 2022.

[28] T. Morocutti and D. Shalaby, "Receptive field regularized CNNs with traditional audio augmentations," DCASE2022 Challenge, Tech. Rep., June 2022.

[29] Y. Xin, Y. Zou, F. Cui, and Y. Wang, "Low-complexity acoustic scene classification with mismatch-devices using separable convolutions and coordinate attention," DCASE2022 Challenge, Tech. Rep., June 2022.

[30] J. Yu, R. Shi, T. He, and K. Guo, "Acoustic scene classification based on feature fusion and dilated-convolution," DCASE2022 Challenge, Tech. Rep., June 2022.

[31] Y.-F. Shao, X. Zhang, G.-G. Bing, K.-M. Zhao, J.-J. Xu, Y. Ma, and W.-Q. Zhang, "Mini-segnet for low-complexity acoustic scene classification," DCASE2022 Challenge, Tech. Rep., June 2022.

LOW-COMPLEXITY ACOUSTIC SCENE CLASSIFICATION IN DCASE 2022 CHALLENGE

Irene Martín-Morató[1], *Francesco Paissan*[2], *Alberto Ancilotto*[2], *Toni Heittola*[1], *Annamaria Mesaros*[1], *Elisabetta Farella*[2], *Alessio Brutti*[2], *Tuomas Virtanen*[1]

[1]Computing Sciences, Tampere University, Finland
{irene.martinmorato, toni.heittola, annamaria.mesaros, tuomas.virtanen}@tuni.fi
[2] Fondazione Bruno Kessler, Italy
{fpaissan, aancilotto, efarella, brutti}@fbk.eu

ABSTRACT

This paper presents an analysis of the Low-Complexity Acoustic Scene Classification task in DCASE 2022 Challenge. The task was a continuation from the previous years, but the low-complexity requirements were changed to the following: the maximum number of allowed parameters, including the zero-valued ones, was 128 K, with parameters being represented using INT8 numerical format; and the maximum number of multiply-accumulate operations at inference time was 30 million. Despite using the same previous year dataset, the audio samples have been shortened to 1 second instead of 10 second for this year challenge. The provided baseline system is a convolutional neural network which employs post-training quantization of parameters, resulting in 46.5 K parameters, and 29.23 million multiply-and-accumulate operations (MMACs). Its performance on the evaluation data is 44.2% accuracy and 1.532 log-loss. In comparison, the top system in the challenge obtained an accuracy of 59.6% and a log loss of 1.091, having 121 K parameters and 28 MMACs. The task received 48 submissions from 19 different teams, most of which outperformed the baseline system.

Index Terms— Acoustic scene classification, low-complexity, DCASE Challenge

1. INTRODUCTION

The task of acoustic scene classification is defined as classifying a short excerpt of audio into a class of a predefined set of classes, that indicates the context where the audio was recorded [1]. The task has been one of the main topics in the DCASE Challenge from its inception, and has developed from the original setup to include different additional problems, such as multiple devices and low-complexity conditions [2]. The current setup advances further towards real-world applicability by defining the low-complexity constraints, in terms of maximum number of parameters and maximum number of operations permitted at inference time, typical of current IoT devices (or microcontrollers).

For real-world applications, a classification method for acoustic scenes is expected to work in very diverse conditions, including audio captured with different devices and as short as possible inference time. The first task on low-complexity acoustic scene classification was defined in 2020 for only three classes and a single device [2], for which many submissions obtained very high performance. The solutions most commonly imposed restrictions on the model architectures, using slim models and depth-wise separable CNNs. In addition, pruning and post-training quantization of the model weights were popular choices [3]. The data mismatch between training and testing when dealing with multiple devices has been first introduced as a separate task in 2019, and then repeated in 2020. The majority of the systems handled the mismatch through data augmentation [2], with the best performance in the 2020 task being 76.5% accuracy and 1.21 log loss [4].

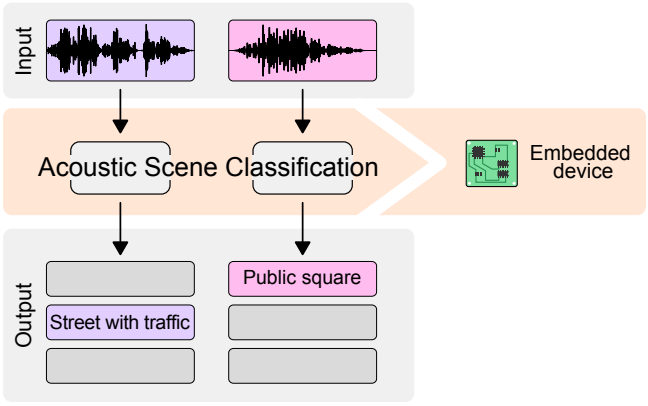


Figure 1: Acoustic scene classification for audio recordings.

The combination of multiple devices and low-complexity requirements was introduced in DCASE 2021 Challenge, with the model complexity limit being set to 128 K for the non-zero parameters. This created the situation in which most high-performing systems were very close to the allowed model size limit. Sparsity used in combination with quantization emerged as a popular and efficient way of reducing the model size, while the most popular system architectures submitted in 2021 were residual models. A few teams used modified versions of available residual models suitable for processing power-constrained devices, like MobileNet [5] and EfficientNet [6]. The best performing system had an accuracy of 76.1% and log loss of 0.724 [7], while 18 submitted systems had an accuracy above 70%.

The current edition formulates the problem of low-complexity acoustic scene classification by defining more concretely the low-complexity limitations by selecting a class of target devices for which the developed system should be suitable. This results in the number of allowed parameters being maximum 128 K, counting all parameters, in contrast from DCASE 2021 when only the non-zero ones were counted. In addition, a limit of 30 million multiply-accumulate operations (MMACs) is approximated based

This work was partially funded by the EU H2020 project MARVEL (project number: 957337).

on the computing power of the target device class.

This paper introduces the results and analysis of the DCASE 2022 Challenge Task 1: Low-Complexity Acoustic Scene Classification with Multiple Devices. The paper is organized as follows: Section 2 introduces the task setup, dataset, and baseline system. Sections 3 and 4 present the challenge participation statistics and analysis of the submitted systems, respectively, while Section 5 presents conclusions and ideas for future development of this task.

2. TASK SETUP

The particular aspect of this task is moving towards practical considerations that bring developed systems closer to possible target devices that impose constraints on computing power and capacity. Generalization across different devices is already a longer running feature of the acoustic scene classification task.

2.1. Dataset and performance evaluation

The task uses **TAU Urban Acoustic Scenes 2022**, a newly released version of the previous acoustic scene datasets. The data consists of recordings from ten acoustic scenes which represent the target classes [8]: *airport*, *indoor shopping mall*, *metro station*, *pedestrian street*, *public square*, *street with medium level of traffic*, *travelling by a tram*, *travelling by a bus*, *travelling by an underground metro* and *urban park*. Data was recorded in multiple European cities, with recordings from ten cities available in the training set and 12 in the evaluation set (two new cities compared to the training).

The audio files have been recorded simultaneously with four devices denoted A, B, C, and D, and another 11 devices denoted S1-S11 were simulated using the audio from device A. The development and evaluation sets consist of 64 and 22 hours of data, respectively. For complete details on the dataset creation and the exact amounts of data per device, we refer the reader to [2]. The difference from the previous datasets is that for this edition the audio data is presented in segments having a duration of 1 s, in order to comply with the inference time and computational limitations imposed by the considered target devices.

The submissions were evaluated using multi-class cross-entropy and accuracy. Accuracy was calculated as macro-average (average of the class-wise performance for each metric), but because the data is balanced, this corresponds to the overall accuracy. The systems were ranked based on the multi-class cross-entropy (log loss), for a ranking independent of the operating point.

As in each edition of the challenge, the audio material in the evaluation data was released two weeks prior to the challenge deadline. The participants were expected to provide class predictions for the provided audio material, and submit the system output for evaluation, together with additional information on the methods. The reference annotation of the evaluation data is only available to task organizers and was used for scoring the submissions.

2.2. System complexity requirements

The computational complexity is measured in terms of parameter count and MMACs (million multiply-accumulate operations) with the requirements modeled after Cortex-M4 devices (e.g. STM32L496@80MHz or Arduino Nano 33@64MHz).

The maximum number of parameters is 128 K, with variable type fixed into INT8, and counting all parameters. This is a major difference from DCASE 2021 in which the 128 K model size limit was only for non-zero parameters, and there was no specific format imposed on the numerical representation. This change was made because in a real operational situation, even with a sparse model, the zero-valued parameters add to the number of MACs performed at inference, and produce additional computational overhead for handling sparsity.

System	Log loss	Accuracy	MMACs
keras	1.575 ($\pm$ 0.018)	42.9% ($\pm$ 0.77)	29.23 M

Table 1: Baseline system size and performance on the development dataset. The value after $\pm$ is the standard deviation for 10 runs.

The maximum number of MACS per inference is 30 MMACs, approximated based on the computing power of the target device class. This limit mimics the fitting of audio buffers into SRAM (fast access internal memory) on the target device for the analysis segment of 1 s, and allows some head space for feature calculation (e.g. FFT), assuming that the most commonly used features fit under this limit. In case learned features (embeddings) are used, e.g. VGGish [9], OpenL3 [10] or EdgeL3 [11], the network used to generate them contributes to the overall model size and complexity. Participants are required to provide full information about the model size and complexity in their technical report accompanying the submission. To facilitate model size calculation for the challenge participant, a script for calculating the number of parameters and the MMACs is provided for Keras, TFLite and PyTorch models[1].

3. BASELINE SYSTEM

The baseline system has the same architecture as the 2021 one, being based on a convolutional neural network (CNN). The system consists of three CNN layers and one fully connected layer, followed by a softmax layer. The model is trained for 200 epochs with a batch size of 16. Complete details about the model and the parameters are provided with the code[2]. The feature extraction step follows a classical approach, where log mel-band energies are extracted every 40 ms with a 50% hop size. This results in an input shape of 40×51 for each 1 second audio file. Post-training quantization to 8 bits is used to reduce the model complexity. The quantization was done after training, using TFLite from TensorFlow 2.0, and setting the weights to *INT8* type. The baseline system has a total number of parameters of 46512. The baseline system overall performance on the development data and system complexity information are provided in Table 1.

4. CHALLENGE RESULTS

The task received 48 submissions from a number of 19 teams. The number of participants in this edition is lower than in previous years, but similar to participation statistics of the other tasks. Only three of the 19 teams have lower performance than the baseline. The best system has a log loss of 1.091 and accuracy of 59.6%, with the four best spots belonging to team Schmid_CPJKU [12].

[1] https://github.com/AlbertoAncilotto/NeSsi

[2] https://github.com/marmoi/dcase2022_task1_baseline

Rank	Label	Log loss (95% CI)	Accuracy [%] (95% CI)
1	Schmid_CPJKU_3	1.091 (1.040 - 1.141)	59.6 (59.4 - 59.9)
5	Chang_HYU_1	1.147 (1.081 - 1.214)	60.8 (60.6 - 61.1)
9	Morocutti_JKU_task1_4	1.311 (1.253 - 1.369)	54.5 (54.2 - 54.8)
11	AI4EDGE_IPL_4	1.330 (1.281 - 1.378)	51.6 (51.3 - 51.9)
14	Sugahara_RION_3	1.366 (1.305 - 1.426)	51.7 (51.4 - 51.9)
19	Park_KT_2	1.431 (1.364 - 1.498)	52.7 (52.4 - 53.0)
20	Zou_PKU_1	1.442 (1.362 - 1.521)	56.3 (56.0 - 56.6)
21	Yu_XIAOMI_1	1.456 (1.409 - 1.504)	46.2 (46.0 - 46.5)
22	Houyb_XDU_1	1.481 (1.416 - 1.547)	49.3 (49.0 - 49.5)
23	Singh_Surrey_3	1.492 (1.441 - 1.544)	45.9 (45.6 - 46.2)
(26)	DCASE2022 baseline	1.532 (1.490 - 1.574)	44.2 (44.0 - 44.5)

Table 2: Performance on the evaluation set of the best systems for top 10 teams. The first column represents the overall rank of the system among 49 (including the baseline). The baseline is not officially ranked, but its log loss corresponds in order to place 26.

4.1. Performance analysis

The performance (log loss and accuracy) obtained by the top 10 teams, best system of each team, are presented in Table 2 and depicted in Figure 2. The submission label was simplified to remove redundant information; submission number was kept for correspondence with the results on the website[3]. The 95% confidence intervals for log loss were calculated using the jackknife procedure [13].

The ranking of the systems is based on log loss, where the top ranked one is the system of Schmid_CPJKU [12] with a log loss of 1.091. Its accuracy of 59.6% is second-best accuracy among the 48 submissions. Team Chang_HYU [14] is ranked second by log loss, but has the overall best accuracy among all submissions. Their accuracy is 60.8%, which appears to be significantly higher than Schmid_CPJKU according to the 95% confidence interval, given the amount of data in the evaluation set. Compared to last edition, the top accuracy has decreased by 16%, and the log loss of the top systems is much higher. While in 2021 there were 21 systems with a log loss under 1, this year there is none. Top 10 systems have a log loss under 1.5, and an accuracy between 45.9%-60.8%. The decrease in performance is mostly a consequence of the data segment size being reduced from 10 to 1 second.

Considering all submissions, the difference in performance between data belonging to devices seen or not in training is generally 10% in accuracy. However, the simulated unseen devices still have a better recognition rate than data from the real device D, which is the GoPro - it appears that its characteristics are very different from those of handheld devices developed for audio (mobile phones and tablets). Among the seen devices, the mobile devices have similar recognition rate, whether real (B, C) or simulated (S); systems have slightly better performance on device A. Given that most data in the development set belongs to device A, the relatively small difference in performance among devices shows that the systems have very strong generalization properties which cover the device mismatch. We also observe good generalization between seen and unseen cities, with almost no difference in classification performance between them. Class-wise performance indicates that some acoustic scenes are more difficult overall: while scenes like bus or park obtain accuracies over 70-80% for many systems, the large majority of systems classify scenes from pedestrian street and public square with around 30% accuracy only.

[3]https://dcase.community/challenge2022/task-low-complexity-acoustic-scene-classification-results

4.2. Machine learning characteristics

Regarding feature extraction, all the systems make use of log-mel energies or mel spectogram, sometimes in combination with other features like deltas, spectral entropy/flatness, CQT or Gammatone. Augmentation techniques are used by most of the systems, only 5 teams do not report use of augmentation[4]. The most popular technique is mixup (used by 33 systems), followed by SpecAugment and pitch shifting (used by 16 systems). Only one team, Zou_PKU [15] uses SpecAugment++, which is applied not only at the input but also at the hidden space of the neural network, to enhance also the intermediate feature representations. The system is ranked 7th based on the accuracy.

The most popular architectures are CNNs (used by 34 systems); some report use of MobileNet [5] (still convolutions, but depthwise separable) or BC-ResNet [7]. The use of residual models is reported by five teams, a significant reduction compared to the 2021 edition when residual networks were the most popular architecture. The top team Schmid_CPJKU uses a teacher-student setup, where the PaSST models pretrained on AudioSet are used as teacher, and the student model is a RF-regularized CNN [16]. The system is based on their previous submission's system reducing its complexity to fit the current constraints. For data augmentation they use Frequency MixStyle, mixing frequency-wise statistics to enhance device generalization.

4.3. System complexity analysis

Almost all submissions are based on inverted residual blocks, or a slight variation of this convolutional block. This is mainly because the common pattern for all participants was to adapt state-of-the-art convolutional networks to meet the computational requirements. Among the adapted networks there are MobileNets and BC-ResNets, and one submission with ShuffleNet. Other notable solutions include the use of very involved feature extraction solutions coupled with very simple neural architectures. While the networks where only slightly modified or carefully designed to meet the computational requirements (without any particular trick), a lot of focus was put on the training and data augmentation strategies. In particular, to boost inference performance, quantization-aware training (QAT) was applied by most of the participants. Another common

[4]The analysis of machine learning characteristics is based on self-reported information provided by the authors with the submission of the system outputs.

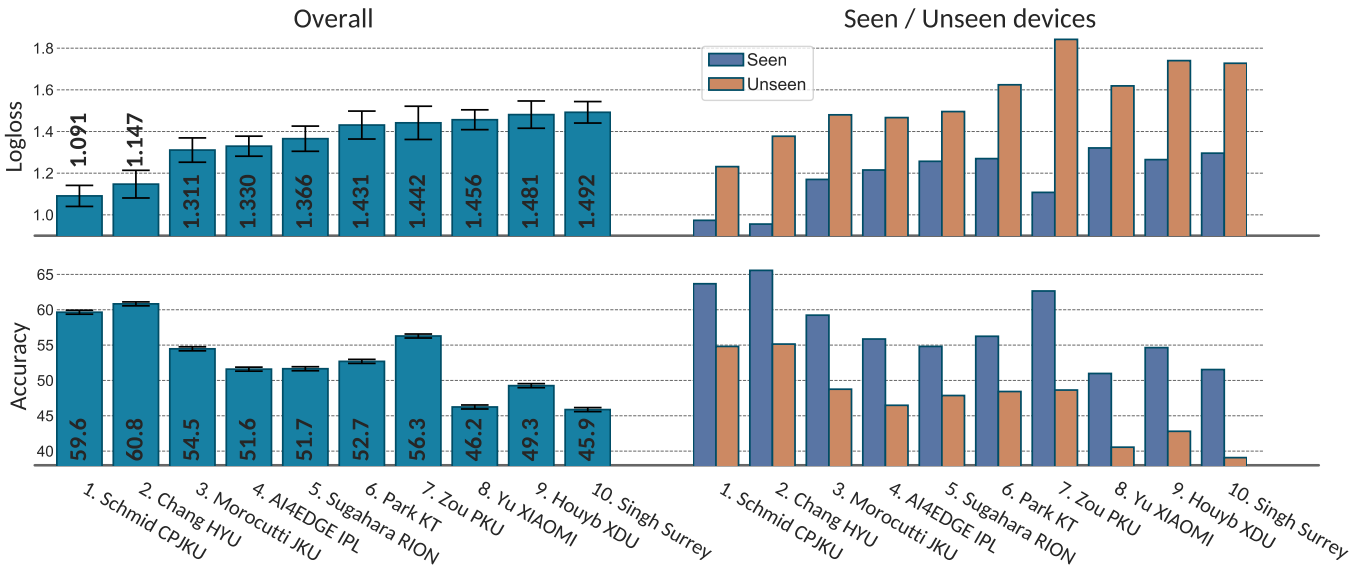


Figure 2: Classification log loss and accuracy for overall and for seen/unseen devices performance of top 10 teams, best system per team.

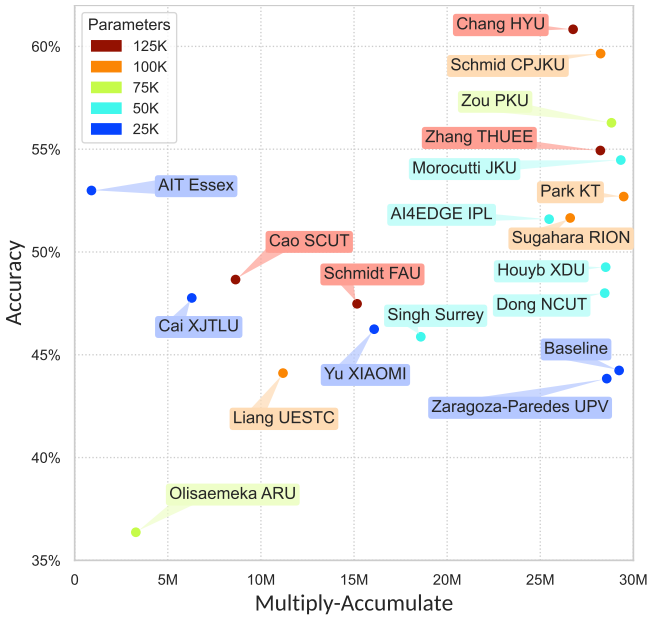


Figure 3: Analysis of performance/complexity tradeoff for different submissions.

alternative was knowledge distillation with pretrained bigger networks fine-tuned on the proposed task. Given the homogeneity in network topology of the submissions, the models proposed perform similarly in acoustic sound classification without being to diverse in computational requirements.

Three out of the top four performing models are based on architectures characterised by large receptive fields employing, respectively, a transformer architecture, coordinate attention and an encoder-decoder architecture. This proved to optimize the performance given the limited resources available, cleverly maximizing the working memory usage of the network, as this parameter was not limited in the task description. Another notable architecture is that proposed in AIT_Essex [17], providing almost optimal performance but very limited MMACs and/or parameter usage. This is possible thanks to their optimized convolutional block, which resembles a grouped convolution whose inputs are a combination of the original input sequence. More standard approaches, based on BC-ResNet, inverted residual blocks or standard bi-dimensional convolutions proved less effective at solving the task with the very limited resources available. This highlights the necessity to develop and optimize neural networks specifically for different hardware platforms. In JH_PM_HYU [14], the authors used clever regularization techniques in order to improve the generalization capabilities of the network. In conclusion, it is clear that, despite clever architectural designs, neural networks trained with optimized preprocessing and training strategies outperform the other approaches. In the future, it would be nice to see such techniques applied to the less computationally expensive models. A performance versus computational cost plot containing the best performing system of each participating team is presented in Figure 3.

5. CONCLUSIONS

This paper presented an analysis of the Low-Complexity Acoustic Scene Classification task in DCASE 2022 Challenge. The task was modeled after devices to bring the research problem closer to real-world applications. The number of MMAC set to 30 M and the total maximum number of parameters set to 128 K have been a sufficient constraint to receive a variety of interesting techniques, even though most systems were close to the imposed limits. The top systems employed large receptive fields, coordinate attention and transformer architectures to optimize performance, while quantization-aware training was the most used technique among participants to fulfill the complexity constrains. The number of submissions has decreased slightly from previous years, which may reflect the increased complexity of the task. However, the use of different devices for context-awareness is a sought after direction for applicability, therefore solutions suitable for limited computational power are needed. Moreover, the task could consider steering development towards solutions where room for improvement is still needed, like minimizing the working memory usage.

6. REFERENCES

[1] E. Benetos, D. Stowell, and M. D. Plumbley, *Approaches to Complex Sound Scene Analysis*. Cham: Springer International Publishing, 2018, pp. 215–242.

[2] T. Heittola, A. Mesaros, and T. Virtanen, "Acoustic scene classification in DCASE 2020 challenge: generalization across devices and low complexity solutions," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2020 Workshop*, Tokyo, Japan, November 2020, pp. 56–60.

[3] K. Koutini, F. Henkel, H. Eghbal-zadeh, and G. Widmer, "CP-JKU submissions to DCASE'20: Low-complexity cross-device acoustic scene classification with rf-regularized CNNs," DCASE2020 Challenge, Tech. Rep., June 2020.

[4] S. Suh, S. Park, Y. Jeong, and T. Lee, "Designing acoustic scene classification models with CNN variants," DCASE2020 Challenge, Tech. Rep., June 2020.

[5] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4510–4520.

[6] M. Tan and Q. V. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *36th Int. Conf. on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, ser. Proceedings of Machine Learning Research, K. Chaudhuri and R. Salakhutdinov, Eds., vol. 97. PMLR, 2019, pp. 6105–6114.

[7] B. Kim, Y. Seunghan, K. Jangho, and C. Simyung, "QTI submission to DCASE 2021: Residual normalization for device-imbalanced acoustic scene classification with efficient design," DCASE2021 Challenge, Tech. Rep., June 2021.

[8] A. Mesaros, T. Heittola, and T. Virtanen, "Acoustic scene classification in DCASE 2019 challenge: closed and open set classification and data mismatch setups," in *Proc. of the DCASE 2019 Workshop*, New York, Nov 2019.

[9] S. Hershey, S. Chaudhuri, D. P. W. Ellis, J. F. Gemmeke, A. Jansen, R. C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold, M. Slaney, R. J. Weiss, and K. Wilson, "CNN architectures for large-scale audio classification," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017, pp. 131–135.

[10] J. Cramer, H.-H. Wu, J. Salamon, and J. P. Bello, "Look, listen and learn more: Design choices for deep audio embeddings," in *IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, Brighton, UK, May 2019, pp. 3852–3856.

[11] S. Kumari, D. Roy, M. Cartwright, J. P. Bello, and A. Arora, "EdgeL3: Compressing L3-net for mote scale urban noise monitoring," in *2019 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, May 2019, pp. 877–884.

[12] F. Schmid, S. Masoudian, K. Koutini, and G. Widmer, "CP-JKU submission to dcase22: Distilling knowledge for low-complexity convolutional neural networks from a patchout audio transformer," DCASE2022 Challenge, Tech. Rep., June 2022.

[13] B. Efron, "Jackknife-after-bootstrap standard errors and influence functions," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 54, no. 1, pp. 83–127, 1992. [Online]. Available: http://www.jstor.org/stable/2345949

[14] J.-H. Lee, J.-H. Choi, P. M. Byun, and J.-H. Chang, "Hyu submission for the DCASE 2022: Efficient fine-tuning method using device-aware data-random-drop for device-imbalanced acoustic scene classification," DCASE2022 Challenge, Tech. Rep., June 2022.

[15] Y. Xin, Y. Zou, F. Cui, and Y. Wang, "Low-complexity acoustic scene classification with mismatch-devices using separable convolutions and coordinate attention," DCASE2022 Challenge, Tech. Rep., June 2022.

[16] K. Koutini, H. Eghbal-zadeh, and G. Widmer, "Receptive field regularization techniques for audio classification and tagging with deep convolutional neural networks," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 1987–2000, 2021.

[17] L. Pham, A. Schindler, H. Tang, and T. Hoang, "DCASE 2021 task 1A: Technique report," DCASE2021 Challenge, Tech. Rep., June 2021.

A SUMMARIZATION APPROACH TO EVALUATING AUDIO CAPTIONING

Irene Martín-Morató, Manu Harju, Annamaria Mesaros

Computing Sciences Tampere University, Finland
{irene.martinmorato, manu.harju, annamaria.mesaros}@tuni.fi

ABSTRACT

Audio captioning is currently evaluated with metrics originating from machine translation and image captioning, but their suitability for audio has recently been questioned. This work proposes content-based scoring of audio captions, an approach that considers the specific sound events content of the captions. Inspired from text summarization, the proposed measure gives relevance scores to the sound events present in the reference, and scores candidates based on the relevance of the retrieved sounds. In this work we use a simple, consensus-based definition of relevance, but different weighing schemes can be easily incorporated to change the importance of terms accordingly. Our experiments use two datasets and three different audio captioning systems and show that the proposed measure behaves consistently with the data: captions that correctly capture the most relevant sounds obtain a score of 1, while the ones containing less relevant sounds score lower. While the proposed content-based score is not concerned with the fluency or semantic content of the captions, it can be incorporated into a compound metric, similar to SPIDEr being a linear combination of a semantic and a syntactic fluency score.

Index Terms— audio captioning, evaluation, content-based retrieval

1. INTRODUCTION

Automated captioning, the description of images, audio, or video content using unrestrained natural language, is an active research topic in all these fields. The first works in image captioning defined it as a machine translation task, and evaluated it using metrics from machine translation such as BLEU [1], METEOR [2] and ROUGE [3], which are primarily based on n-gram overlap between the reference and candidate caption. Subsequently, it was observed that these metrics do not correlate well with human opinion [4], resulting in development of new metrics optimized for image captioning such as SPICE [5] and CIDEr [6]. SPICE measures performance using a graph-based semantic representation that explicitly encodes the objects, attributes and relationships found in image captions, while CIDEr measures how well a candidate sentence matches the consensus of a set of image descriptions, using n-grams weighted using Term Frequency Inverse Document Frequency (TF-IDF) weighting, combining n-grams of varying lengths (typically up to 4-grams). Further optimization in image captioning resulted in SPIDEr [7], a linear combination of SPICE and CIDEr.

Audio captioning was defined and evaluated the same way as image captioning [8]. In the last few years, the DCASE Challenge has accelerated development of audio captioning methods, seen clearly in the significant improvement of the evaluation results. Approaches are based on encoder-decoder systems, with the decoder usually a recurrent network with sequence-to-sequence modeling. Recently, the use of transformers has become very popular [9], with pretrained models such as BERT [10] and BART [11] consistently ranked as state of the art. The evaluation scheme has not changed, and consists of the same set of metrics, from BLEU and ROUGE to CIDEr, SPICE and SPIDEr.

The recent development of large language processing models, notably BERT [10], brought a new approach to measuring similarity of text, initiating research into metrics more suitable for measuring audio captioning outputs. For example the work in [12] proposed FENSE, a new metric which combines Sentence-BERT for semantic similarity [13] with an error detector to penalize erroneously formed sentences. FENSE was shown to correlate with human judgments, in experiments that evaluated the output of four different captioning systems on the Clotho dataset [14], and showed that FENSE ranked the best systems the same way as humans did.

We introduce a novel perspective to audio captioning, namely summarization. Instead of cross-modal machine translation, we regard audio captioning as *cross-modal summarization*. A detailed description of a complex acoustic scene using natural language would include information on all the different sounds present at the scene. However, humans in fact do not care or pay attention to everything, and may consider some content irrelevant. We assume that annotators required to describe audio content would include the most relevant content, subject to their own judgement. In this respect, the textual description can be viewed as a summary.

In this paper we propose a novel measurement for audio captioning, which considers the captions content in terms of sound events. The different sounds mentioned in the captions are given relevance scores based on the annotators' consensus in producing the reference captions. A candidate caption is then evaluated based on the relevance of its content with respect to the reference information, given that the optimal caption for an audio will contain the topmost relevant sounds. In effect, this is a content-based scoring scheme that rewards the captions which retrieve the most relevant sounds in the captioned audio. The contributions of this work are the following: (i) we formulate audio captioning as audio-to-text summarization; (ii) we propose a method to estimate relevance of sound events based on multiple reference captions; (iii) we propose a relevance-based score for evaluation of audio captions content.

The paper is organized as follows: Section 2 introduces the proposed sound relevance estimation scheme and the proposed metric for evaluating candidate captions. Section 3 presents evaluation results using different captioning systems and datasets and introduces ideas for possible further development. Finally, Section 4 presents the conclusions.

This paper has received funding from Jane and Aatos Erkko Foundation, Finland.

2. CONTENT-BASED EVALUATION OF CAPTIONS

This work is inspired by the Pyramid method for evaluating content selection of textual summaries [15]. It was observed that among textual summaries produced by humans, many seem equally good without having identical content. This is valid for human-produced audio captions too, in particular for complex scenes. The Pyramid method starts by annotating "summarization content units" (SCU), then uses these SCUs to produce an optimal summary. The Pyramid score is defined as the ratio of a candidate summary to the ideal summary having the same number of SCUs.

Considering the caption as a summary, we define our content units as sound events. In consequence, we will evaluate a caption based on the sound events that are mentioned in this caption. The sound events are extracted from the captions using the AudioSet ontology [16], on the grounds that while it may be an incomplete list of possible sounds, the ontology provides a large set of the most common sounds. The human-produced reference captions are processed to extract the sound events from each caption, which are then assigned a relevance as explained in the following.

2.1. Estimating relevance of sound events

In its simplest way, the relevance of a sound event to an audio clip can be defined based on how many annotators have referred to it in the caption they provided. Consider the example from Table 1: given ten captions for one audio clip, four of them mention birds singing, two mention car passing by, all ten mention children laughing, and nine mention children talking.

We define relevance of sound event i as its consensus-based weight, calculated as:

$$rel_i = \frac{N_i}{\sum_{j=1}^{M}(N_j)} \tag{1}$$

where N_i is the number of times sound event i is mentioned in the captions assigned to a clip, with M being the total number of sound events mentioned in all captions. The effect of this definition is that a relevance of 1 is distributed among all the mentioned sound event classes M based on how frequently they appear in the captions:

$$\sum_{i=1}^{M} rel_i = 1. \tag{2}$$

For the example in Table 1, bird singing has a relevance of 0.16, while children laughing has a relevance of 0.40.

One may argue that in certain cases the rare sound events may be more relevant to a clip, instead of the most commonly mentioned ones. Depending on the application and the desired output, the relevance of individual sound events to a clip can be estimated based on direct frequency as in the example above, or using TF-IDF weighing to give more weight to sounds that are very specific to a clip. For simplicity, we only use the former approach in this work, and leave other weighing schemes for future development.

2.2. Evaluation of candidate captions

The content-based score (CB-score) of a candidate caption C containing K sound events is defined as the ratio between the relevance of its content and the relevance of the optimal caption C_K containing the same number K of sound events:

Label	Freq	Relevance
Bird singing	4	0.16
Car passing by	2	0.08
Children laughing	10	0.40
Children talking	9	0.36

Table 1: Consensus-based relevance of sound events to a clip

$$\text{CB-score} = \frac{\sum_{j=1}^{K} rel_j}{\sum_{k=1}^{K} rel_k} \tag{3}$$

where sound events j belong to the candidate caption C, and events k belong to the optimal caption C_K. The optimal caption C_K is defined as containing the most relevant K of the M sound events mentioned in the reference captions. This means that for the example in Table 1 the optimal caption containing only one sound event would contain "children laughing", while the optimal caption containing two sound events would contain "children laughing" and "children talking". Table 2 gives examples of CB-score calculation for different captions, using the relevance scores from Table 1.

3. EXPERIMENTS

For each audio clip, the reference captions are processed to extract sound events and estimate their consensus-based relevance. Then, the candidate caption is evaluated against the optimal caption as explained in the previous section. Experiments were performed using three different systems and multiple datasets, in order to verify the behavior of the metric. For comparison, SPIDEr scores (as used in DCASE Challenge 2022) and FENSE scores were calculated.

3.1. Captioning datasets and systems

The datasets used for evaluating the behaviour of the proposed metric are Clotho [14] which has five captions per clip, for audio clips 15 to 30 seconds long that were collected from Freesound [17] and AudioCaps [18], consisting of 10-second long audio clips from AudioSet [16], for which only the test split has five captions per clip, the rest has only one caption.

We use three different systems to generate automatic captions. The first system is the DCASE task6 subtask A baseline system[1], which is a sequence-to-sequence transformer based on BART model[2]. The second system (ED-RNN) consists of an Encoder-Decoder architecture with an attention layer in the decoder and bi-directional RNN in the encoder. The third model (AACTrans) is a sequence-to-sequence transformer similar to the baseline system, but having smaller number of parameters and using greedy token generation. All three models use VGGish [19] features as inputs. For brevity, we do not present more details about the systems, since they are irrelevant for the objective of this study. The systems are separately trained and tested using Clotho and AudioCaps, respectively, using the training/test splits provided with the datasets.

[1] https://github.com/felixgontier/dcase-2022-baseline

[2] https://huggingface.co/docs/transformers/model_doc/bart

Candidate captions:
C1. Children are talking outside.
C2. A dog is barking at a car passing by.
C3. A car is passing by a group of children that are playing and laughing.

	Sound events	Relevance	Ideal caption content	CB-score
C1	Children talking	0.36	Children laughing	0.36/0.40 =0.90
C2	Dog barking, Car passing by	0.08	Children laughing, Children talking	0.08/(0.40+0.36)=0.11
C3	Car passing by, Children laughing	0.08+0.40	Children laughing, Children talking	0.48/0.76=0.63

Table 2: Example of CB-scores for candidate captions, based on relevance scores from Table 1.

	CLOTHO			AudioCaps		
System	SPIDEr	FENSE	CB-score	SPIDEr	FENSE	CB-score
Baseline	0.22 (0.21, 0.24)	0.46 (0.45, 0.47)	0.49 (0.46, 0.51)	0.34 (0.32, 0.37)	0.57 (0.56, 0.58)	0.63 (0.60, 0.65)
ED-RNN	0.15 (0.14, 0.16)	0.41 (0.40, 0.43)	0.40 (0.38, 0.43)	0.30 (0.28, 0.33)	0.54 (0.53, 0.55)	0.62 (0.59, 0.64)
AACTransformer	0.19 (0.18, 0.21)	0.40 (0.39, 0.41)	0.47 (0.45, 0.50)	0.35 (0.32, 0.37)	0.54 (0.53, 0.55)	0.65 (0.63, 0.68)
Reference caption	0.58 (0.55, 0.61)	0.58 (0.57, 0.59)	0.64 (0.62, 0.66)	0.56 (0.52, 0.59)	0.68 (0.68, 0.69)	0.76 (0.75, 0.78)

Table 3: Performance of different systems on Clotho and AudioCaps datasets.

3.2. Practical implementation details

The extraction of sound events from the textual captions is not possible using simple word matching, due to the unconstrained nature of the annotation process. We therefore match individual terms and their synonyms in a controlled manner, to extract what we call sound event tokens as well as possible. The processing steps are the following: First, AudioSet vocabulary and the reference captions are tokenized and lemmatized to obtain the root form for each word. Then, for each token in the caption, its first order synonym and first order hypernym are extracted from WordNet [20], based on its POS (part of speech) obtained using spaCy [21] and nltk [22]. This allows identifying words that are related to sounds, when they do not match the exact vocabulary of AudioSet. AudioSet vocabulary is used as a two-level hierarchy to standardize the depth of the vocabulary. If the child node in AudioSet does not match the extracted sound event from the caption but its synonym does, the extracted token is matched to the parent node. For example "people *talking*" is processed to "group, people, citizenry, *speak*, talk, communicate", where *speak* is a child of *speech* in AudioSet, therefore *talking* is matched to *speech*.

We expect that this matching process will result in some amount of errors, in some cases matching wrong terms. However, we consider that the vocabulary used in the captions and in general for describing sounds are a small subset of WordNet, and words are mostly used with their most common meaning, which means that wrongly matched synonyms should not affect the scores very much. There are also cases where a correspondence to the AudioSet vocabulary is not found, and therefore the process fails to identify sounds.

3.3. Numerical results and analysis

The evaluation results are presented in Table 3 for the three systems, comparing SPIDEr, FENSE and CB-score. Confidence intervals for the metrics were calculated using the jackknife resampling procedure on the system output. To understand the dynamics of the metric values and to have better insight on the expected upper bound for each metric, we selected randomly one of the five reference captions and compared it to the other four using the three metrics. The results of this evaluation are presented in the last row of Table 3.

Among the captioning systems, we observe that the baseline has the best performance on Clotho, a result which is consistent for all three metrics. On AudioCaps, the baseline and AACTransformer have similar performance, with confidence intervals of the metrics overlapping. In comparison, the evaluation of one reference caption against the others produces similar SPIDEr score on both datasets, while FENSE and CB-score are significantly higher on AudioCaps. It may be noteworthy that the datasets are of different size, with 46k and 29k captions available for training in AudioCaps and Clotho, respectively. Since usually larger datasets lead to more robust models, the size difference may explain the AudioCaps better scores.

To understand the meaning of these average values, the distribution of the scores calculated for the baseline system are illustrated in Fig. 1. We observe that SPIDEr is very concentrated close to 0, while FENSE looks normally distributed between 0 and 1. CB-score has many values at the extremes, with 0 corresponding to the case when no sounds were matched between the evaluated caption and the reference ones, and 1 for the case when the evaluated caption contained the most relevant sound(s). For Clotho 33% of the captions produced for the test split have CB-score 0, while 27% have a score of 1. On the other hand the FENSE score is more moderate, with most captions scoring between 0.4–0.6, only 13 files having a FENSE score of 0. SPIDEr scores are under 0.25 for 66% of the captions. AudioCaps data obtains more balanced scores, with only around 13% of the captions in the test split getting a CB-score of 0, while the maximal score of 1 is obtained by around 30% of the files, similar to Clotho.

For comparison, Fig. 2 shows the CB-score and FENSE distributions for evaluating one reference caption (selected randomly) against the others. Interestingly, 17% of the Clotho test split gets CB-score 0, which means that the captions produced by different people do not contain the same sound events. This information is not indicated by FENSE, which measures the general semantic similarity of the sentences. For both Clotho and AudioCaps, a CB-score of 1 is obtained by around 40% of the human-produced captions.

To analyze the content evaluated by the CB-score, we investigate the number of sound tokens extracted from the caption. The statistics are shown in Fig. 3; the prediction uses the output of the

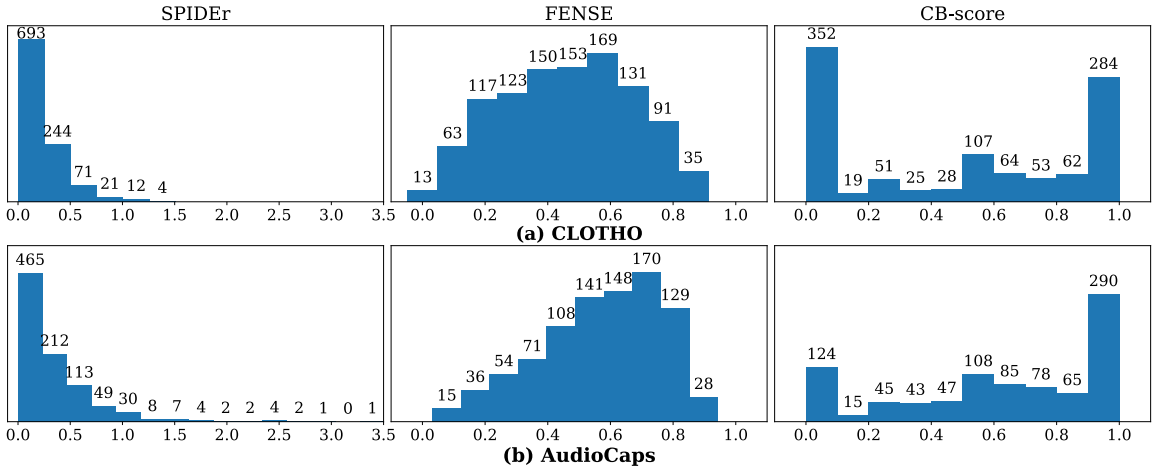


Figure 1: Distribution of SPIDEr, FENSE and CB-score on the test splits of Clotho and AudioCaps for the baseline system.

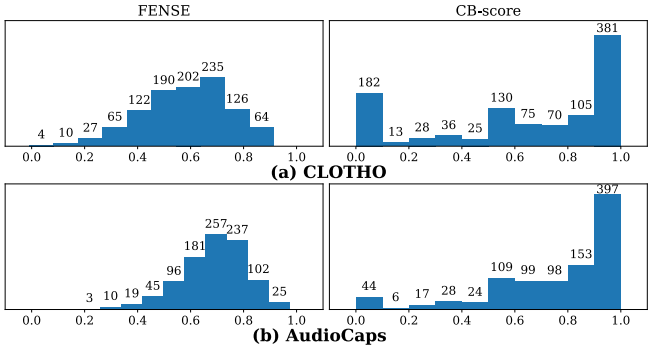


Figure 2: Distribution of FENSE and CB-score for evaluating one reference caption against the others for each clip in the test set.

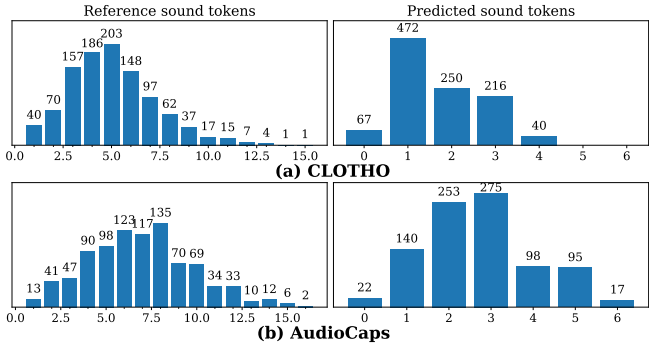


Figure 3: Statistics of the extracted tokens that refer to sound events in the reference captions and in the automatically generated ones.

baseline system. For both Clotho and AudioCaps, our token extraction process results in a large number of terms per clip. On the other hand, 45% of the captions produced for Clotho result in only one token, while 6% do not match any sounds. For AudioCaps, the same system produces captions which are richer in sound tokens, and our token extraction method results in 2 or 3 tokens for 54% of them.

3.4. Discussion and further development

Based on its formulation in Section 2, it can be noticed that this is a precision-type of metric, reflecting how many of the sound events that appear in a caption are as highly weighted as possible. This scoring penalizes the presence of sounds if there are others more highly weighted but not included in the caption. A recall-oriented score can be formulated by defining the optimal caption as containing the *average number* of sound events in the reference annotations (instead of the same number K as the candidate). It can also be observed that the formulation in eq.3 is not sensitive to extra information, which in effect means that inserted sounds are not penalized, even if they were not present in the reference captions at all.

The idea of scoring the caption using consensus is not new: CIDEr also uses consensus, but is based on n-grams, while our proposal is based on sound events. We find the use of sound events as units of information more relevant to the audio captioning than the use of n-grams, even though this ignores the lexical structure. Because the definition of the sound relevance is independent from the CB-score itself, therefore the CB-score is flexible in allowing sound relevance score formulations depending on the target application. Additionally, given the relatively short sentences available as captions, we hypothesize that a higher number of reference captions would provide more reliable relevance estimates.

4. CONCLUSIONS

We introduced CB-score, a metric to evaluate how well the sound events mentioned in automatically produced captions correspond to the sounds mentioned in the reference captions. We also introduced a simple method to estimate relevance of reference sounds based on multiple captions, which can be extended depending on the target application to give more importance to the most commonly mentioned sounds or to rare sounds that are highly specific to a clip. The proposed metric lacks the ability to measure lexical or grammar structure of the caption, therefore it is not a sufficient metric for evaluating audio captions, but it successfully summarizes the content of the acoustic description, which means that it can be used as such for evaluating captioning tasks focused on the correctness rather than syntactic richness of the produced sentences. Similar to SPIDEr being a linear combination of a semantic and a syntactic fluency score, the CB-score can further evolve into more complex compound measurements, with additional components measuring for example syntactic and semantic aspects of the produced caption.

5. REFERENCES

[1] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "Bleu: A method for automatic evaluation of machine translation," in *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ser. ACL '02. USA: Association for Computational Linguistics, 2002, p. 311–318.

[2] A. Lavie and A. Agarwal, "METEOR: An automatic metric for MT evaluation with high levels of correlation with human judgments," in *Proceedings of the Second Workshop on Statistical Machine Translation*. Prague, Czech Republic: Association for Computational Linguistics, Jun. 2007, pp. 228–231.

[3] C.-Y. Lin, "ROUGE: A package for automatic evaluation of summaries," in *Text Summarization Branches Out*. Barcelona, Spain: Association for Computational Linguistics, Jul. 2004, pp. 74–81.

[4] M. Hodosh, P. Young, and J. Hockenmaier, "Framing image description as a ranking task: Data, models and evaluation metrics," *J. Artif. Int. Res.*, vol. 47, no. 1, p. 853–899, May 2013.

[5] P. Anderson, B. Fernando, M. Johnson, and S. Gould, "Spice: Semantic propositional image caption evaluation," in *Computer Vision – ECCV 2016*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds. Cham: Springer International Publishing, 2016, pp. 382–398.

[6] R. Vedantam, C. L. Zitnick, and D. Parikh, "Cider: Consensus-based image description evaluation." in *CVPR*. IEEE Computer Society, 2015, pp. 4566–4575.

[7] S. Liu, Z. Zhu, N. Ye, S. Guadarrama, and K. Murphy, "Improved image captioning via policy gradient optimization of spider," in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 873–881.

[8] K. Drossos, S. Adavanne, and T. Virtanen, "Automated audio captioning with recurrent neural networks," in *2017 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, 2017, pp. 374–378.

[9] X. Mei, X. Liu, Q. Huang, M. D. Plumbley, and W. Wang, "Audio captioning transformer," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2021 Workshop (DCASE2021)*, Barcelona, Spain, November 2021, pp. 211–215.

[10] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the NAACL HLT, Vol. 1*. Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 4171–4186.

[11] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, "BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Jul. 2020, pp. 7871–7880.

[12] Z. Zhou, Z. Zhang, X. Xu, Z. Xie, M. Wu, and K. Q. Zhu, "Can audio captions be evaluated with image caption metrics?" in *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022, pp. 981–985.

[13] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence embeddings using siamese bert-networks," in *EMNLP/IJCNLP*, K. Inui, J. Jiang, V. Ng, and X. Wan, Eds. Association for Computational Linguistics, 2019, pp. 3980–3990.

[14] K. Drossos, S. Lipping, and T. Virtanen, "Clotho: an audio captioning dataset," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 736–740.

[15] A. Nenkova and R. Passonneau, "Evaluating content selection in summarization: The pyramid method," in *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics: HLT-NAACL 2004*. Boston, Massachusetts, USA: Association for Computational Linguistics, May 2004, pp. 145–152.

[16] J. F. Gemmeke, D. P. W. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio set: An ontology and human-labeled dataset for audio events," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017, pp. 776–780.

[17] F. Font, G. Roma, and X. Serra, "Freesound technical demo," in *ACM International Conference on Multimedia (MM'13)*, ACM. Barcelona, Spain: ACM, Oct. 2013, pp. 411–412.

[18] C. D. Kim, B. Kim, H. Lee, and G. Kim, "AudioCaps: Generating captions for audios in the wild," in *Proceedings of the 2019 Conference of the NAACL HLT, Vol. 1*. Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 119–132.

[19] S. Hershey, S. Chaudhuri, D. P. W. Ellis, J. F. Gemmeke, A. Jansen, C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold, M. Slaney, R. Weiss, and K. Wilson, "Cnn architectures for large-scale audio classification," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017, pp. 131–135. [Online]. Available: https://arxiv.org/abs/1609.09430

[20] C. Fellbaum, *WordNet: An Electronic Lexical Database*. The MIT Press, 05 1998. [Online]. Available: https://doi.org/10.7551/mitpress/7287.001.0001

[21] I. Montani, M. Honnibal, M. Honnibal, S. V. Landeghem, A. Boyd, H. Peters, P. O. McCann, M. Samsonov, J. Geovedi, J. O'Regan, D. Altinok, G. Orosz, S. L. Kristiansen, L. Miranda, D. de Kok, Roman, E. Bot, L. Fiedler, G. Howard, Edward, W. Phatthiyaphaibun, Y. Tamura, S. Bozek, murat, R. Daniels, M. Amery, B. Böing, B. Vanroy, and P. K. Tippa, "explosion/spaCy: v3.3.0: Improved speed, new trainable lemmatizer, and pipelines for Finnish, Korean and Swedish," Apr. 2022. [Online]. Available: https://doi.org/10.5281/zenodo.6504092

[22] S. Bird, E. Klein, and E. Loper, *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.", 2009.

FEW-SHOT BIOACOUSTIC EVENT DETECTION USING AN EVENT-LENGTH ADAPTED ENSEMBLE OF PROTOTYPICAL NETWORKS

John Martinsson[1,2*], *Martin Willbo*[1], *Aleksis Pirinen*[1], *Olof Mogren*[1], *Maria Sandsten*[2]

[1]Computer Science, RISE Research Institutes of Sweden, Sweden
{john.martinsson, martin.willbo, aleksis.pirinen, olof.mogren }@ri.se
[2] Centre for Mathematical Sciences, Lund University, Sweden
maria.sandsten@matstat.lu.se

ABSTRACT

In this paper we study two major challenges in few-shot bioacoustic event detection: variable event lengths and false-positives. We use prototypical networks where the embedding function is trained using a multi-label sound event detection model instead of using episodic training as the proxy task on the provided training dataset. This is motivated by polyphonic sound events being present in the base training data. We propose a method to choose the embedding function based on the average event length of the few-shot examples and show that this makes the method more robust towards variable event lengths. Further, we show that an ensemble of prototypical neural networks trained on different training and validation splits of time-frequency images with different loudness normalizations reduces false-positives. In addition, we present an analysis on the effect that the studied loudness normalization techniques have on the performance of the prototypical network ensemble. Overall, per-channel energy normalization (PCEN) outperforms the standard log transform for this task. The method uses no data augmentation and no external data. The proposed approach achieves a F-score of 48.0% when evaluated on the hidden test set of the Detection and Classification of Acoustic Scenes and Events (DCASE) task 5.

***Index Terms*—** Machine listening, bioacoustics, few-shot learning, ensemble

1. INTRODUCTION

The human-induced accelerated loss in biodiversity [1] has led to a need for automated and low-cost wildlife monitoring where machine learning is a promising way forward [2]. Passive acoustic monitoring (PAM) is becoming an important tool in ecology for monitoring animal populations through their vocalizations [3]. Annotating PAM data is costly and requires specific domain expertise which motivates research on few-shot learning for bioacoustic event detection [4]. The goal of few-shot bioacoustic event detection is to detect the onset and offset of animal vocalizations in sound recordings using only a few annotated examples.

Recent work has demonstrated that prototypical networks are a promising approach for few-shot sound event detection [5, 6, 7], but a remaining challenge is high variance in classification accuracy between models because of the small amount of training data. Recent work on audio classification and sound event detection has demonstrated promising results using ensembles [8, 9, 10]. Ensembles may be especially useful for the few-shot task due to the high variance in classification accuracy between models [11]. To the best of our knowledge, prior work on ensemble methods for few-shot sound event detection remains understudied and motivated by this we study the effect of using an ensemble of prototypical networks for few-shot bioacoustic event detection.

Another challenge in few-shot sound event detection is the high variability in event lengths for the different event classes [6]. The event lengths can range from milliseconds to multiple seconds, which necessitates methods capable of adapting to the task specific event lengths. Wang et al. [6] suggest that future work should look into adapting the context window to the few-shot task. A common approach is to use a model which can handle variable context windows, train using a fixed context window, and at test time adapt the context window to the few-shot task. In this work we propose choosing the embedding function as well as the context window based on the few-shot examples. The embedding function is chosen from a set of embedding functions trained on different context windows, thus acting as experts on certain event lengths. Another way to approach this problem is by using a proposal based method [12].

2. METHOD

In this section we present our method which is based on prototypical networks [13] and extended with an event-length adapted ensemble. We describe how each embedding function for the prototypical networks is trained and how the embedding functions are selected based on the few-shot examples to produce an ensemble prediction at test time. The full source code and instructions on how to reproduce the results can be found at: `https://github.com/johnmartinsson/few-shot-learning-bioacoustics`.

2.1. Training the embedding function

The goal is to learn an embedding function from the base training data, acting as a proxy task for the few-shot task. The base training data set consists of annotated sound recordings for 47 known event classes and one "unknown" event class. The set of sound event classes are disjoint between the base training data and the few-shot task. We are given the start and end times $\mathcal{A}_k = \{(s_i^k, e_i^k)\}_{i=1}^N$ of these classes, where (s_i^k, e_i^k) denotes the start and end time of sound event class k for annotation i. There is overlap in the annotations, i.e. two different sound events can occur (partially) simultaneously, and we therefore treat this as a multi-label problem. We model the

*Thanks to the Swedish foundation for strategic research for funding.

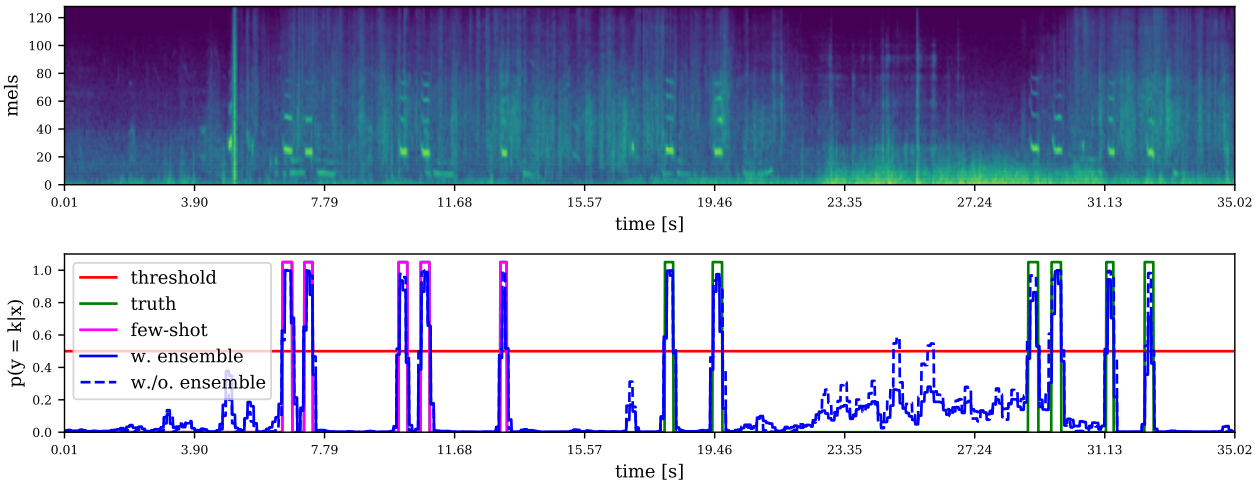


Figure 1: A log Mel spectrogram of part of a sound recording (top) and examples of predictions (bottom) from an ensemble prototypical network (solid blue line) and a prototypical network (dashed blue line) as well as the given few-shot examples (purple line) and remaining ground truth events (green line). The decision threshold τ is 0.5 (red line).

47 known sound event classes and the "unknown" sound event class identically, yielding a total of $K = 48$ classes.

We assume that a fixed length audio segment $x \in \mathbb{R}^T$, consisting of T consecutive audio samples, is fed to the embedding function $f_\theta^T : \mathbb{R}^T \to \mathbb{R}^M$ (see section 2.4 for further details), where $M \ll T$. We split the audio recordings into audio segments $x_i \in \mathbb{R}^T$ by sliding a window of size T with a hop size of $T/2$ over each recording. For each audio segment x_i, a target vector $y_i \in \{0,1\}^{K \times n}$ is derived. If $n = T$ it means that the target contains one label per audio sample. Choosing $n < T$ means that the temporal resolution for the target is reduced. The resulting dataset $\mathcal{D}_b = \{(x_i, y_i)\}_{i=1}^N$ defines the sound event detection task used to train the embedding function.

A prediction of the target classes for a given audio segment x_i is derived by $\hat{y}_i = h_\phi(f_\theta^T(x_i))$, where $h_\phi(\cdot)$ is a linear layer followed by an element-wise sigmoid activation function, and $f_\theta^T(\cdot)$ is a convolutional neural network where the first layer is a (non-learnable) time-frequency transform.

The loss function is the mean element-wise binary cross-entropy between the target y_i and the prediction $\hat{y}_i$, where the mean is taken over the class dimension K and the temporal dimension n.

For a fixed T, we train a set of C different embedding functions, parameterized as $\Theta = \{\theta_1, \ldots, \theta_C\}$, each with different randomly initialized weights of the neural network, different training and validation splits of the base training data, and different time-frequency transforms in the first layer of the embedding function.

2.2. Prototypical network at test time

At test time we are given a sound recording and the $M = 5$ first event examples of the class of interest. We denote these $A_p = \{(s_i, e_i)\}_{i=1}^M$ and call them the *positive* sound events. We assume that the gaps between the positive event annotations are background noise and let $A_n = \{(e_i, s_{i+1})\}_{i=1}^{M-1}$ denote the start and end time of the $M-1$ first *negative* sound events. We assume the likelihood of an annotator missing events to be low.

Let $l_i = e_i - s_i$ be the length of annotation i. If $l_i < T$ we "expand" the annotation with the $(T - l_i)/2$ preceding and subsequent audio samples to get an audio segment of length T, and if $l_i \geq T$ we do not expand. We then split this into segments of length T by sliding a window of size T over the signal with a hop size of $T/16$ (if expanded this will only result in one segment). Let S_p denote the set of positive audio segments derived from these annotated start and end times, and let S_n denote the set of negative audio segments. We use the embedding function f_θ^T and define the prototypes as

$$c_k = \frac{1}{|S_k|} \sum_{x \in S_k} f_\theta^T(x) \tag{1}$$

and derive a pseudo-probability of audio segment x belonging to sound class k from the prototypical network by

$$p_\theta(y = k|x) = \frac{\exp(-d(f_\theta^T(x), c_k))}{\sum_{k'} \exp(-d(f_\theta^T(x), c_{k'}))}, \tag{2}$$

where $k \in \{n, p\}$ and $d(f_\theta^T(x), c_k)$ denotes the Euclidean distance between the query $f_\theta^T(x)$ and the prototype c_k.

The query set S_q is derived by sliding a window of size T over the signal with a hop size of $T/2$. The reason for setting the hop size relative to T is that this means that we do equally many predictions for each audio sample in the validation recordings.

2.3. Our contributions

We now present the two main contributions of this paper: i) an event-length adapted embedding function for the few-shot task, and ii) using an ensemble of predictions.

Adapting the embedding function. We use the annotated positive events $A_p = \{(s_i, e_i)\}_{i=1}^M$ and compute the set of event lengths $L = \{e_i - s_i\}_{i=1}^M$. We choose $T^* \in \{T_1, 2^1 T_1, 2^2 T_1, 2^3 T_1\}$ such that $\sqrt{(T - l_{\text{min}}/2)^2}$ is minimized, where l_{min} is the shortest event length in L.

We choose $T_1 = 2048$ which is 0.09 seconds at a sampling rate of 22050 Hz so that we can plausibly detect the shortest

events in the few-shot validation set. We limit the amount of extra computation needed during training and the extra memory needed during inference by setting the maximum T to $2^3 T_1$.

Ensemble. Let $\Theta = \{\theta_i^{T^*}\}_{i=1}^{C}$ denote the set of parameters of C different prototypical network models adapted to the average event length of the few-shot task. Then we define

$$p_\Theta(y = k|x) = \frac{1}{C} \sum_{\theta \in \Theta} p_\theta(y = k|x) \quad (3)$$

as in [14], which can be viewed as a uniformly-weighted mixture of experts. We say that x belongs to the positive event class if $p_\Theta(y = p|x) > \tau$ and otherwise x belongs to the negative event class. This is done for every $x \in S_q$. Finally, if the query is classified as a positive event then the start and end time associated with that query is used as the predicted positive event timings.

2.4. Details of the embedding function

The embedding function consists of a time-frequency transform followed by a convolutional neural network, both of which are briefly described below.

Time-frequency transform. The first layer of the embedding function is a time-frequency transform. We use the Mel transform where the number of Mel bins is 128, the window size is roughly 25ms, and the hop size is half the window size. We either use the log transform as a loudness normalization or we use PCEN [15] with fixed parameters developed for speech audio or for bioacoustics as suggested in [16].

Convolutional neural network. The convolutional neural network used is an adapted version of the 10-layer residual neural network [17] used in the baseline for the challenge. Specifically, we i) add the classification head $h_\phi(\cdot)$ so that we can model the defined multi-label task, ii) use the same number of filters in every convolutional layer, and iii) reduce the max pooling along the time-dimension when audio segments are too short.

2.5. Evaluation metric

The method is evaluated by taking the harmonic mean over the F-scores for the different subsets in the evaluation sets. The F-score is computed by a bi-partite matching between the predicted and ground truth events, where the requirement for a match is an intersection-over-union (IoU) of at least 0.3 [4].

2.6. Post-processing

Since we get one prediction for each query audio segment, this limits the possible length of the prediction with this model. To solve this, we simply merge all overlapping predicted positive events into one detected event with a single start and end time.

A predicted positive event will only be considered to be a match with a true positive event during evaluation if they have an intersection-over-union (IoU) of at least 0.3. We therefore remove predictions which are shorter than $0.3 * l_{\text{avg}}$ or longer than $(1/0.3) * l_{\text{avg}}$, where l_{avg} is the average event length of the given five annotations. Since predictions of these lengths can on average not be matched with true events as the evaluation is defined.

Subset	Mean event length	Mean gap length	Mean density
HB	11.25 ± 3.11	6.12 ± 5.39	0.73 ± 0.12
ME	0.22 ± 0.03	1.40 ± 0.04	0.17 ± 0.02
PB	0.12 ± 0.08	59.89 ± 55.55	0.01 ± 0.02

Table 1: Few-shot validation data statistics.

3. DATA

We use the few-shot examples to compute the mean event length, the mean gap length, and the density of annotated sound events – see table 1. The few-shot validation set consists of three different subsets: HB, ME, and PB. The HB subset contains long events with low noise. The ME subset contains short events with low noise. The PB subset contains very short events with very high noise. The mean event length is defined as the mean length of the five annotated events; the mean gap length is defined as the mean length of the *unannotated* gaps between the five annotated events; and the density is the sum of the time of the five annotated events divided by the total time. A full description of the dataset can be found in [18].

4. EXPERIMENTS AND RESULTS

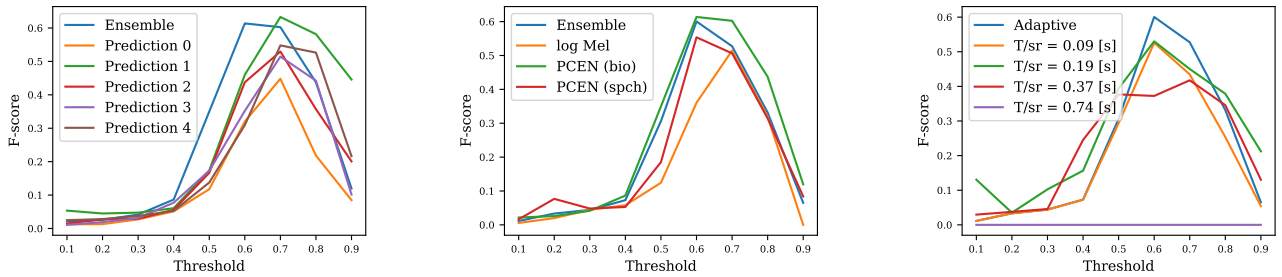
We have trained each embedding function on the described multi-label task on the base training data using the Adam [19] optimizer with a learning rate of $1\text{e}{-3}$. The network is trained on a random split with 80% training data and validated on the remaining 20%. Each network in the ensemble is trained on a different random split. The training proceeds until we have observed no reduction in validation loss for the last 10 epochs and the model with the lowest validation loss is chosen as the final model. The temporal resolution of the targets have been fixed to $n = 16$, meaning that we have 16 targets for any given audio segment.

In figure 2a we compare the F-score achieved on the few-shot validation set when using an ensemble of five predictions with using each of these predictions by themselves. The time-frequency transform used is PCEN (bioacoustics). The achieved F-score by the ensemble is higher than the best of these individual predictions for $0.4 \leq \tau \leq 0.6$, and outperforms or matches the mean of them for other τ. We also note that the optimal τ is around 0.7 for the single predictions, and moves to 0.6 for the ensemble.

In figure 2b we compare the F-score of a five prediction ensemble for each time-frequency transform and compare this to an ensemble over all three time-frequency transform ensembles. We do not observe a significant increase in F-score when comparing the time-frequency ensemble to the ensemble using the PCEN (bioacoustic) time-frequency transform, but the time-frequency ensemble outperforms the ensemble using PCEN (speech) and log Mel transform. The optimal threshold τ varies around 0.6 to 0.7 for the ensembles using a single transform, and is at 0.6 for the time-frequency ensemble.

In figure 2c we compare the F-score achieved on the few-shot validation set when using the event-length adapted embedding functions in the ensemble with using any of the fixed $T \in \{T_1, 2^1 T_1, 2^2 T_1, 2^3 T_1\}$. Adapting the embedding function increases performance from 53.0% (using best $T = 4096$) to 60.0% F-score for $\tau = 0.6$.

In table 2 we show an ablation study. Adapting the embedding function increases the F-score on average with 8.3 percentage points, and adding the ensemble increases the F-score an additional



(a) Ensemble over seeds, and training splits. (b) Ensemble over time-frequency transforms. (c) Adapted embedding function.

Figure 2: A comparison between: (a) an ensemble of five predictions using embedding functions trained on PCEN (bioacoustics) features with each of the individual predictions, (b) an ensemble of embedding functions trained and tested on log Mel, PCEN (bioacoustics), or PCEN (speech), with an ensemble of predictions over all three, and (c) the adaptive embedding function with using each of the fixed size embedding functions respectively (sr denotes the sample rate). All results in the figure are derived on the few-shot validation set.

Method	Ensemble	Adaptive	F-score
Ours	No	No	41.3 ± 3.8
Ours	No	Yes	49.6 ± 5.3
Ours	Yes	Yes	60.0

Table 2: An ablation study of our system on the few-shot validation set where we add adaptive embedding functions and ensemble.

System	External data	Augmentation	F-score
Baseline (TM)	No	No	12.3
Baseline (PN)	No	No	5.3
Ours [20]	No	No	48.0
Tang et al., [21]	No	No	62.1
Liu et al., [22]	Yes	Yes	48.2
Hertkorn [23]	No	No	44.4
Liu et al., [24]	Yes	Yes	44.3

Table 3: The final F-score evaluation on the hidden test set for the baselines provided by the challenge organizers: template matching (TM) and prototypical networks (PN), and the top five submissions for the challenge.

11.4 percentage points. We compare against a prototypical network using an embedding function (no ensemble) which has been trained on PCEN (speech) and a best performing fixed segment length of 4096. The F-score when no ensemble is performed is the average (and standard deviation) over each single network in the ensemble.

In table 3 we present the F-score from the final evaluation on the hidden test set from the challenge. We include information on whether or not the system uses data augmentation techniques or external datasets during training.

5. DISCUSSION AND CONCLUSIONS

In this section we will discuss our results and relate them to the baselines which were all developed concurrently to our work.

During development of this method we observed that random sampling of S_n, the set of negative examples does not work well for validation files with high event densities, which is why we chose to use the gaps between the first five annotated events instead. This observation was also made in concurrent work submitted to the challenge [21, 22, 24]

We further observed that a fixed audio segment size T resulted in poor predictive performance on the few-shot validation set in cases where event-lengths deviated from size T. Indicating the importance of adapting the embedding function.

We observed that the optimal threshold was different for the few-shot validation tasks and choosing a default value of $\tau = 0.5$ to be detrimental. However, finding an optimal threshold for the few-shot tasks is a difficult problem. Using an ensemble alleviates this issue by moving the optimal threshold closer to the default value.

The ensemble improves performance by correctly predicting most true positives, while reducing the number of false positives. This could intuitively be thought of as the ensemble being in agreement for true positive predictions, the average of which still yields a high pseudo-probability, while being in disagreement when predicting false positives, the average of which would be closer to 0.5. This effect can be seen in figure 1, where some of the false positives predicted when not using an ensemble (dashed blue line) are removed by using an ensemble of the predictions (solid blue line), leading to a reduction in false-positives.

The baselines in this study were all developed concurrently to our work. Tang et al., [21] propose using a frame-level cross-entropy loss function for training instead of episodic training as the proxy task. Our approach is similar when setting the temporal resolution n of the target vector to the number of frames in the time-frequency image. The effect of varying temporal resolutions n for the proxy task would be interesting to study in future work. Tang et al. [21] further propose an iterative training scheme to adapt their method to the few-shot task [21] where the unlabeled audio in the test files is iteratively classified and then used for training. Liu et al. [22] and Liu et al. [24] use transductive inference to better adapt to the evaluation set, and Hertkorn [23] studies the importance of choosing appropriate parameters for the used time-frequency transform.

In conclusion, we have shown that choosing the embedding function based on the event lengths will increase performance, and that false-positives can be reduced by an ensemble of predictions. We have also shown that out of the three time-frequency transforms we have studied, PCEN (bioacoustics) performs best, followed by PCEN (speech) and log Mel.

6. REFERENCES

[1] S. Díaz, J. Settele, E. S. Brondízio, H. T. Ngo, J. Agard, A. Arneth, P. Balvanera, K. A. Brauman, S. H. Butchart, K. M. Chan, A. G. Lucas, K. Ichii, J. Liu, S. M. Subramanian, G. F. Midgley, P. Miloslavich, Z. Molnár, D. Obura, A. Pfaff, S. Polasky, A. Purvis, J. Razzaque, B. Reyers, R. R. Chowdhury, Y. J. Shin, I. Visseren-Hamakers, K. J. Willis, and C. N. Zayas, "Pervasive human-driven decline of life on Earth points to the need for transformative change," *Science*, vol. 366, no. 6471, 2019.

[2] D. Tuia, B. Kellenberger, S. Beery, B. R. Costelloe, S. Zuffi, B. Risse, A. Mathis, M. W. Mathis, F. van Langevelde, T. Burghardt, R. Kays, H. Klinck, M. Wikelski, I. D. Couzin, G. van Horn, M. C. Crofoot, C. V. Stewart, and T. Berger-Wolf, "Perspectives in machine learning for wildlife conservation," *Nature Communications*, vol. 13, no. 1, pp. 1–15, 2022.

[3] R. Gibb, E. Browning, P. Glover-Kapfer, and K. E. Jones, "Emerging opportunities and challenges for passive acoustics in ecological assessment and monitoring," *Methods in Ecology and Evolution*, vol. 10, no. 2, pp. 169–185, 2019.

[4] V. Morfi, I. Nolasco, V. Lostanlen, S. Singh, A. Strandburg-Peshkin, L. Gill, H. Pamuła, D. Benvent, and D. Stowell, "Few-Shot Bioacoustic Event Detection: A New Task at the DCASE 2021 Challenge," *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2021 Workshop (DCASE2021)*, no. November, pp. 145–149, 2021.

[5] B. Shi, M. Sun, K. C. Puvvada, C. C. Kao, S. Matsoukas, and C. Wang, "Few-Shot Acoustic Event Detection Via Meta Learning," *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, vol. 2020-May, pp. 76–80, 2020.

[6] Y. Wang, J. Salamon, N. J. Bryan, and J. P. Bello, "Few-shot sound event detection," *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, vol. 2020-May, pp. 81–85, 2020.

[7] J. Pons, J. Serra, and X. Serra, "Training Neural Audio Classifiers with Few Data," *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, vol. 2019-May, pp. 16–20, 2019.

[8] D. Lee, S. Lee, Y. Han, and K. Lee, "Ensemble of Convolutional Neural Networks for Weakly-Supervised Sound Event Detection Using Multiple Scale Input," *DCASE 2017*, vol. 1, no. November, pp. 14–18, 2017. [Online]. Available: http://www.cs.tut.fi/sgn/arg/dcase2017/documents/challenge_technical_reports/DCASE2017_Lee_199.pdf

[9] L. Nanni, Y. M. Costa, R. L. Aguiar, R. B. Mangolin, S. Brahnam, and C. N. Silla, "Ensemble of convolutional neural networks to improve animal audio classification," *Eurasip Journal on Audio, Speech, and Music Processing*, vol. 2020, no. 1, 2020.

[10] L. Nanni, G. Maguolo, S. Brahnam, and M. Paci, "An ensemble of convolutional neural networks for audio classification," *Applied Sciences (Switzerland)*, vol. 11, no. 13, pp. 1–27, 2021.

[11] N. Dvornik, J. Mairal, and C. Schmid, "Diversity with cooperation: Ensemble methods for few-shot classification," *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2019-October, pp. 3722–3730, 2019.

[12] P. Wolters, C. Daw, B. Hutchinson, and L. Phillips, "Proposal-based Few-shot Sound Event Detection for Speech and Environmental Sounds with Perceivers," pp. 1–7, 2021. [Online]. Available: http://arxiv.org/abs/2107.13616

[13] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," *Advances in Neural Information Processing Systems*, vol. 2017-Decem, pp. 4078–4088, 2017.

[14] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," *Advances in Neural Information Processing Systems*, vol. 2017-December, pp. 6403–6414, 2017.

[15] Y. Wang, P. Getreuer, T. Hughes, R. F. Lyon, and R. A. Saurous, "Trainable frontend for robust and far-field keyword spotting," *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, no. 1, pp. 5670–5674, 2017.

[16] V. Lostanlen, J. Salamon, M. Cartwright, B. Mcfee, A. Farnsworth, S. Kelling, and J. P. Bello, "Per-Channel Energy Normalization: Why and How," *IEEE SIGNAL PROCESSING LETTERS*, no. September, pp. 1–6, 2018. [Online]. Available: http://www.justinsalamon.com/uploads/4/3/9/4/4394963/lostanlen_pcen_spl2018.pdf

[17] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," *Arxiv.Org*, vol. 7, no. 3, pp. 171–180, 2015. [Online]. Available: http://arxiv.org/pdf/1512.03385v1.pdf

[18] I. Nolasco, S. Singh, E. Vidana-Villa, E. Grout, J. Morford, M. Emmerson, F. Jensens, H. Whitehead, I. Kiskin, A. Strandburg-Peshkin, L. Gill, H. Pamula, V. Lostanlen, V. Morfi, and D. Stowell, "Few-shot bioacoustic event detection at the dcase 2022 challenge," 2022. [Online]. Available: https://arxiv.org/abs/2207.07911

[19] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," pp. 1–15, 2014. [Online]. Available: http://arxiv.org/abs/1412.6980

[20] J. Martinsson, M. Willbo, A. Pirinen, O. Mogren, and M. Sandsten, "Few-shot bioacoustic event detection using a prototypical network ensemble with adaptive embedding functions," Tech. Rep., 2022.

[21] J. Tang, X. Zhang, T. Gao, D. Liu, X. Fang, J. Pan, Q. Wang, J. Du, K. Xu, and Q. Pan, "Few-shot embedding learning and event filtering for bioacoustic event detection," iFLYTEK Research Institute, Hefei, China, Tech. Rep., 2022.

[22] H. Liu, X. Liu, X. Mei, Q. Kong, W. Wang, and M. D. Plumbley, "Surrey system for DCASE 2022 task 5 : few-shot bioacosutic event detection with segment-level metric learning," University of Surrey, Surrey, United Kingdom, Tech. Rep., 2022.

[23] M. Hertkorn, "Few-shot bioacoustic event detection : don't waste information," ZF Friedrichshafen AG, Friedrichshafen, Germany, Tech. Rep., 2022.

[24] M. Liu, J. Zhang, L. Wang, J. Peng, and C. Hu, "Bit SRCB teams 's submission for DCASE2022 task5 - few-shot bioacoustic event detection," Beijing Institute of Technology, Beijing, China, Tech. Rep., 2022.

QUANTITY OVER QUALITY? INVESTIGATING THE EFFECTS OF VOLUME AND STRENGTH OF TRAINING DATA IN MARINE BIOACOUSTICS

Andrea Napoli[*]*, Paul R. White, Thomas Blumensath*

Institute of Sound and Vibration Research
University of Southampton, UK
{an1g18, P.R.White, Thomas.Blumensath}@soton.ac.uk

ABSTRACT

The trade-off between the quality and quantity of training data is considered for the detection of minke whale (*Balaenoptera acutorostrata*) vocalisations. The performance of two different detectors is measured across a range of label strengths using training sets of different sizes. A detector based on spectrogram correlation and a convolutional neural network (CNN) are considered. The results show that increasing label strength does not benefit either detector past a certain point, corresponding here to a label density of 60 to 70%. Performance is found to be good even when labels are extremely weak (4% label density). Additionally, it is noted that performance of the spectrogram correlation plateaus beyond the use of 5 training calls, whereas the CNN's performance continues to increase up to the maximum training set size tested. Finally, interaction effects are observed between label strength and quantity, indicating that larger training sets are more robust to weaker labels. Overall, these findings suggest that there is indeed a benefit to collecting more, lower quality data when training a CNN, but that for a correlation-based detector this is not the case.

Index Terms— Weak labels, marine bioacoustics, CNN, spectrogram correlation, sound event detection

1. INTRODUCTION

Passive acoustic monitoring (PAM) forms a major part of marine mammal conservation. Acoustic surveys are an effective and non-invasive means to further our understanding of species-wise geographic distributions, migration patterns and feeding grounds, monitor ecosystem health, and help to mitigate the impacts of human activity. Automated analysis of survey data can improve our ability to achieve these goals, whilst substantially reducing the manual effort required [1].

It is generally accepted that larger training sets allow deep neural networks to build richer representations and improve performance across many classification tasks [2]–[4]. However, labelling large-scale datasets is expensive and time-consuming, so weak labels are commonly used to allow more training data to be collected [4]–[7]. This has resulted in a "quantity over quality" mantra which is at risk of being rashly applied without giving due consideration to the data distributions, labelling techniques, and algorithms pertinent to a particular application. So, work is needed to empirically determine whether "quantity over quality" retains its relevance in marine bioacoustics.

Weak data in PAM can take many forms [1], but this study will consider the effects of label noise – when some training instances do not represent the label they are assigned. This is common when audio is labelled without exact temporal localisation of the signal of interest, which is the scenario presented here. The longer the label, the more irrelevant or even confounding information is likely to be present. The question is then how long the labels need to be in order to make best use of the analysts' time – accepting some label noise allows the analyst to work faster and thus collect more samples.

Label noise can also be introduced through computer-assisted labelling. In PAM, manual labelling is often combined with generic energy detection and unsupervised clustering to reduce annotation effort, at the cost of increased label noise [8]–[10]. Here, the relevant parameter is the sensitivity threshold for the energy detectors. A lower threshold yields more training samples, but also results in more erroneous labels.

Regardless of the origin of the label noise, the recurring theme is that a direct trade-off exists between the strength and quantity of the training data created. Determining the best labelling strategy therefore requires a quantification of exactly how classification performance is affected by these two opposing variables. The impact of label noise has been already been investigated for a range of audio tasks [11]–[13]. Additionally, the effect of training set size is often reported when new models are developed [4]. However, varying both strength and quantity concurrently, and in a controlled manner, has not been previously considered. Conducting a two-way study is important, since the impacts of the two are not necessarily independent (termed *factor interaction*). Thus, the description of a method to determine the presence of any interaction effects has broader relevance to other audio domains as well.

Once the training data has been collected, several weak learning techniques can be applied. Strong label assumption training (SLAT) is a basic option that splits the audio into frames and then assigns the same parent label to each [4]. This creates more, smaller training instances, both of which benefit smaller datasets, but a portion of these may be incorrectly labelled. In this case, prior estimates of the size of this portion can be used to improve performance [14]. Other options include multiple-instance learning [15] and scalable variants [16], or the use of attention or recurrence mechanisms [17].

For many years, template matching techniques such as matched filtering [18], [19] and spectrogram correlation [19], [20] have been the PAM algorithms of choice for detecting call types with limited variation (known as stereotyped vocalisations). Following widespread adoption in many other disciplines, CNNs have also recently found success in this field [1], [10], [21], [22]. Since

[*] Thanks to BAE Systems and the Engineering and Physical Sciences Research Council for funding.

these methods differ significantly, but are both extensively used, an implementation of each will be tested. We stress that the objective of this is not to directly compare the performance of the two detection algorithms, but to identify any differences in how gradient-based learning and traditional signal processing methods are affected by the quality and quantity of data used.

In summary, this paper contributes the following:

- Design of two detectors for minke whale vocalisations, one based on spectrogram correlation, and the other a CNN.
- A quantification of how label strength affects the performance of these detectors. This is the first such study in marine bioacoustics, and also tests more datapoints than similar studies in other audio domains.
- The addition of a second dimension to the problem space, so as to explore the impact of both quantity and quality of labels simultaneously. Comments are also provided regarding interaction effects between the two factors.

2. DATA

The scenario considered is the detection of minke whales in towed-array data from the 2017 Hawaiian Islands Cetacean and Ecosystem Assessment Survey [23]. The dataset comprises over 23,000 60-second audio files, sampled at 500 kHz on 6 channels. The channels have varying signal-to-noise ratios (SNRs) based on the distances between the hydrophones and the ship. Like other marine mammals, minke whales are under threat from human activities including vessel strikes, fishing gear entanglement, noise pollution, and ingestion of debris, so collecting regular and accurate abundance estimates is important [24].

The minke whales in this area produce stereotyped "boing" calls. A boing comprises a brief pulse followed by a longer, frequency and amplitude modulated component, and has a peak frequency of 1.4 kHz, harmonics up to 9 kHz, and source levels around 150 dB re 1 µPa·m [25], Figure 1. Note that this image is thresholded to improve clarity for illustrative purposes, but the spectrograms used for the experiments are unthresholded to maximise detectability and for better generalisability of the results.

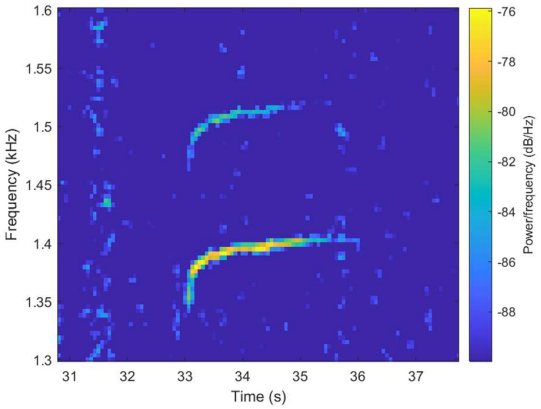


Figure 1: A typical minke whale boing.

For this experiment, 40 calls are identified, and selected such that no audio file contains more than one call. These are strongly labelled by hand, with the start and end times determined to 0.1 s precision. Call durations range from 1.4 to 4.3 s, with a mean of 2.7 s. For the non-target class, 40 files are manually verified to contain only ambient noise. All audio is taken from the same survey day to minimise data leakage. The files are split into 4 validation folds, each containing 30 training calls and 10 test calls. Hereinafter, audio will be referred to as *positive* if it contains a call, and *negative* otherwise.

In addition to the strong labels, 7 sets of weaker labels are generated by increasing the length of the label. Thus, weakly-labelled *positive* audio will contain varying amounts of ambient noise in addition to a call, depending on the label length. For the first 3 sets, the label length is variable, and is equal to the call length plus a fixed time quantity of 0.5, 1 or 1.5 s. Since no call is longer than 5 s, for weaker sets, the label length is fixed, and varies from 5 to 60 s. The extra time added is split randomly between the start and end of the strong label, so the calls can occur at any point in the audio. The strength of each set can be quantitively measured by considering the call duration as a percentage of the overall label length, referred to as the *label density* [11]. Table 1 shows the label density for each training case.

Table 1: Average label density for each of the label strengths tested.

Training Set	*Label Density*
Strong	100%
Strong +0.5 s	84%
Strong +1 s	72%
Strong +1.5 s	63%
5 s	52%
10 s	26%
20 s	13%
60 s	4%

Spectrogram representations of the audio are generated using 250 ms Hamming windows with 75% overlap. The spectrograms are cropped to a narrow band between 1.3 and 1.6 kHz, containing only the peak frequency and one additional harmonic, as per Figure 1. The spectrograms are then divided into 1 s frames with 50% overlap, discarding any excess beyond the length of the label. The resulting frames measure 76 by 16 pixels. Each audio channel is treated independently, resulting in 6 times as many samples. Since the channels have varying SNRs, this acts as a form of data augmentation and helps to regularise the data. SLAT is then applied, so all the frames are assigned the label from their parent audio segment. Note that:

- The "Strong" set contains only samples where the calls entirely fill the frames.
- The "Strong +0.5 s" and "Strong +1 s" sets also include samples that only partially contain calls.
- The weaker sets include partial samples as well as samples that do not contain calls at all (but are still labelled *positive*).

For each label strength, the number of calls in the training sets is varied from 30 down to 1. Training on a single channel of one call is also considered. The test data is the same for every training case in the fold, and is always strongly labelled.

In every case, the *negative* audio contains only ambient sounds, which is not unrealistic for PAM data. Thus, the label noise is entirely one-sided, effectively rendering this a positive-unlabelled (PU) learning problem [26]. The classes are kept balanced by randomly undersampling from the *negative* audio.

Table 2: Average detection accuracies and standard deviations for different sized training sets and label strengths.

№ Calls	*Strong*	*Strong +0.5 s*	*Strong +1 s*	*Strong +1.5 s*	*5 s*	*10 s*	*20 s*	*60 s*
				Spectrogram Correlation				
30	99.0 (0.8)	98.6 (0.7)	99.0 (0.8)	98.4 (1.2)	98.0 (1.2)	97.5 (1.4)	97.6 (1.7)	95.9 (2.1)
20	99.0 (0.8)	98.8 (0.9)	99.0 (0.8)	98.4 (1.1)	98.1 (1.2)	97.5 (1.5)	97.3 (1.7)	96.1 (2.1)
10	98.9 (0.9)	99.1 (0.7)	98.8 (0.7)	98.5 (0.8)	98.7 (1.4)	97.4 (1.5)	97.2 (1.4)	95.7 (2.0)
5	99.1 (0.7)	98.9 (1.0)	98.9 (0.5)	98.5 (0.8)	98.2 (1.5)	97.1 (1.5)	97.0 (1.6)	94.9 (1.7)
1	96.7 (1.7)	96.5 (2.1)	96.8 (1.6)	96.6 (2.2)	96.7 (2.2)	95.4 (2.9)	93.7 (3.7)	90.9 (4.6)
1/6	96.4 (2.2)	96.6 (2.1)	96.6 (2.1)	97.3 (2.2)	96.8 (2.1)	96.3 (2.6)	96.2 (2.3)	93.1 (3.8)
				CNN				
30	98.4 (1.5)	98.0 (2.0)	98.1 (1.1)	96.0 (1.0)	94.1 (0.6)	86.9 (1.2)	82.0 (8.2)	83.5 (3.9)
20	97.5 (2.3)	98.0 (2.0)	97.4 (1.7)	95.7 (1.7)	93.2 (2.0)	83.8 (6.4)	83.1 (5.3)	79.4 (5.7)
10	94.4 (5.7)	96.6 (3.0)	96.7 (2.2)	93.7 (1.8)	92.1 (1.8)	81.6 (1.4)	80.4 (4.0)	78.4 (8.6)
5	94.8 (5.8)	95.9 (3.8)	96.1 (2.7)	93.6 (1.9)	88.9 (1.9)	82.5 (3.5)	79.5 (1.7)	76.0 (6.4)
1	85.6 (2.8)	88.4 (6.1)	85.1 (2.5)	90.9 (5.1)	81.9 (3.7)	75.2 (1.9)	73.2 (5.0)	61.9 (11.3)
1/6	76.2 (8.9)	62.6 (15.0)	66.2 (11.9)	61.4 (20.5)	60.9 (15.4)	54.5 (10.9)	51.7 (10.8)	47.8 (5.6)

3. DETECTORS

Design of the spectrogram correlation detector follows Mellinger and Clark [20]. For each test sample, a 2D correlation is performed with each *positive* training sample and the highest correlation value is taken as the recognition score. The *negative* training samples are unused. The decision threshold is then set as the median recognition score across the test samples. Thus, this implementation implicitly assumes that half of the test samples are *positive*. However, this can easily be modified to use a fixed threshold or a threshold-moving algorithm for imbalanced data [27].

A simple CNN is designed with three convolutional layers and one dense layer. The convolutional layers have 3 by 3 kernels, [2, 2] stride, 8, 16 and 32 filters, and are followed by batch normalisation [28] and RELU activations. The network has 7,170 trainable parameters in total. Spectrogram values are rescaled to the range [0, 1] before input. Training is performed using the Adam optimiser [29] with an initial learning rate of 0.003, a batch size of 50, and early stopping.

4. RESULTS

The average detection accuracies and standard deviations across the 4 validation folds are shown in Table 2. The results are also given graphically, also called interaction plots, in Figure 2.

As expected, the overall trend is that the detectors perform better when trained with stronger labels, and higher quantities of data. However, the results show that both methods can tolerate some label noise (corresponding to a label density of 60 to 70%) without a meaningful reduction in performance. This has an important corollary: increasing the strength of the labels does not improve performance beyond a certain point.

Since the analysis frames are 1 s long, "Strong +1.5 s" is the first dataset for which samples can be labelled *positive* but not contain even a partial call, and this coincides with the accuracy beginning to drop. This suggests a possible interaction between the length of the analysis frames and the strength of the labels, especially for the CNN. However, further work is needed to establish to what extent, if at all, this is the case, and determine whether frame length should also be considered when choosing a label strength.

The spectrogram correlation responds similarly to increasing label quantity, with performance plateauing beyond the use of 5 training calls. Performance of the CNN, however, continues to increase up to the maximum training set size tested. Thus, the CNN is shown to scale better with the quantity of training data available. On the other hand, the spectrogram correlation is more robust to fewer training samples, with the accuracy dropping at most only a few percent between the highest and lowest values tested.

It is observed that using all 6 audio channels instead of only one actually makes the spectrogram correlation perform worse. This is because channels for the same call are highly correlated, so comparing additional channels provides insufficient new information to compensate for the reduced SNR, which only serves to confuse the detector. On the other hand, the varying SNR is shown to be an effective regulariser for the CNN, improving accuracy by an average of 20%. When only a single channel is available, it is likely that similar gains can be achieved by varying the SNR artificially (i.e., standard data augmentation).

Both methods perform well even when labels are extremely weak. In the weakest case, only 4% of the *positive* audio distinguishes it from the non-target class. The spectrogram correlation has far better robustness to label noise, with its accuracy dropping by at most only 6%. This is likely because samples of ambient noise correlate poorly with each other, as well as with samples of minke whale call. The CNN, on the other hand, does not hold this bias, making it susceptible to overfitting, and learning to erroneously discriminate the two classes based on spurious information in the background noise. With only 7,000 parameters, the extremely small capacity of the model may have helped to avoid this.

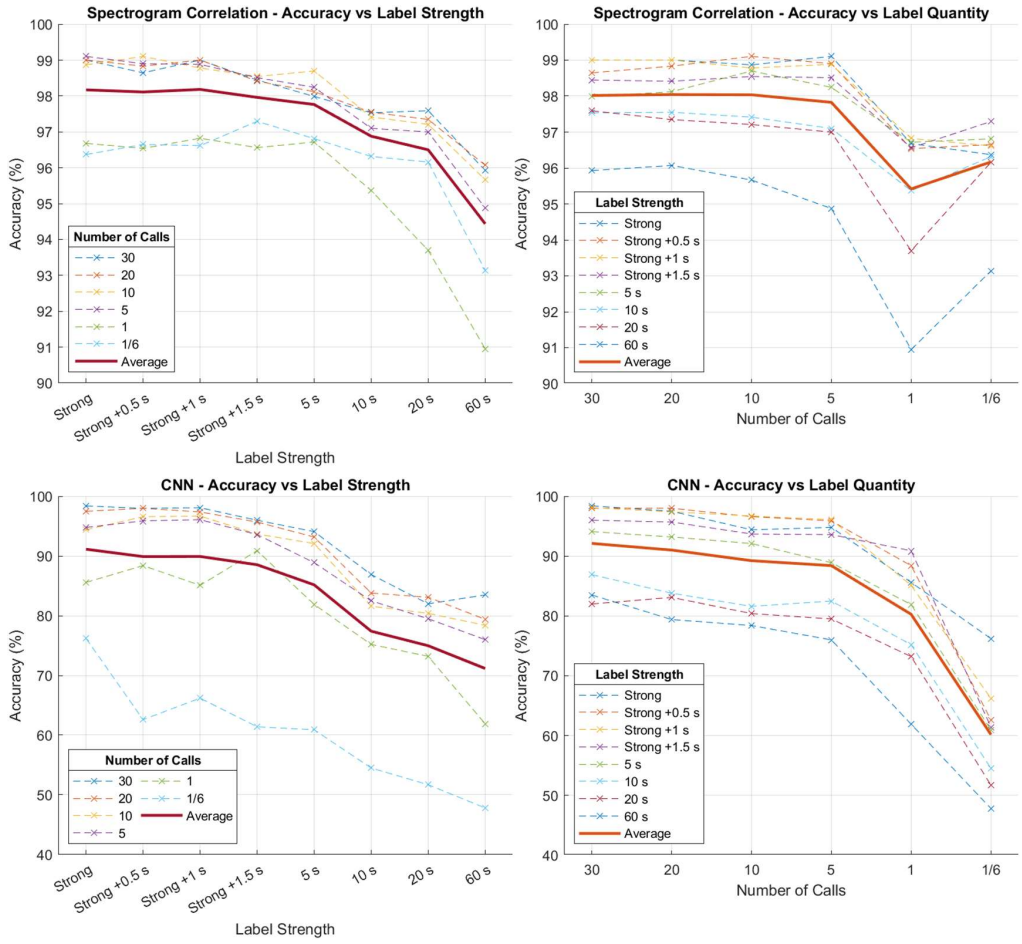


Figure 2: Interaction plots for label quantity and strength, for the spectrogram correlation (top) and CNN (bottom).

The presence of factor interaction is indicated by the lines in an interaction plot being nonparallel. In general, the lines in each plot of Figure 2 can be seen to spread out from left to right, showing that some interaction between label strength and quantity is occurring. Specifically, the results indicate that stronger labels are more robust to smaller training sets, and larger training sets are more robust to weaker labels. The presence of interaction also demonstrates the importance of conducting multi-factor studies.

5. CONCLUSION

This paper studied the effects of volume and strength of training data on the performance of two detectors for minke whale calls. The aim was to determine the relevance to marine bioacoustics of the oft-quoted principle of "quantity over quality". Given the CNN's good scalability to larger training sets, and the performance saturation that occurs when enhancing label strength, the study concludes that "quantity over quality" does indeed hold for CNNs for the call detection scenario presented. Consistency in other machine learning domains suggests that this conclusion is likely to be valid for other gradient-based models as well. However, this is not the case for the spectrogram correlation, which is found to be incapable of exploiting additional training data. On the other hand, the spectrogram correlation demonstrates greater robustness to noisy labels and smaller training sets, making it more appropriate for few-shot learning scenarios.

Future work includes investigating the effects of frame length, mixing labels of different strengths, and using larger training sets to find the saturation points of the CNN's performance. Extensions can also be made to include more advanced weak learning methods, and classification of multiple marine mammal species. Finally, statistical significance tests such as two-way analysis of variance can be used to provide a more objective measure of factor interaction.

6. REFERENCES

[1] P. Nguyen, "Development of artificial intelligence methods for marine mammal detection and classification of underwater sounds in a weak supervision (but) Big Data-Expert context," Doctoral Thesis, Sorbonne University, 2020.

[2] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2016-December, pp. 770–778, Dec. 2015, doi: 10.48550/arxiv.1512.03385.

[3] H. Wei, L. Tao, R. Xie, and B. An, "Open-set Label Noise Can Improve Robustness Against Inherent Label Noise," *Adv. Neural Inf. Process. Syst.*, vol. 10, pp. 7978–7992, Jun. 2021, doi: 10.48550/arxiv.2106.10891.

[4] S. Hershey *et al.*, "CNN Architectures for Large-Scale Audio Classification," *ICASSP, IEEE Int. Conf. Acoust. Speech Signal Process. - Proc.*, pp. 131–135, Sep. 2016, doi: 10.1109/ICASSP.2017.7952132.

[5] J. F. Gemmeke *et al.*, "Audio Set: An ontology and human-labeled dataset for audio events," *ICASSP, IEEE Int. Conf. Acoust. Speech Signal Process. - Proc.*, pp. 776–780, Jun. 2017, doi: 10.1109/ICASSP.2017.7952261.

[6] K. J. Piczak, "ESC: Dataset for Environmental Sound Classification," *MM 2015 - Proc. 2015 ACM Multimed. Conf.*, pp. 1015–1018, Oct. 2015, doi: 10.1145/2733373.2806390.

[7] E. Fonseca, X. Favory, J. Pons, F. Font, and X. Serra, "FSD50K: An Open Dataset of Human-Labeled Sound Events," *IEEE/ACM Trans. Audio Speech Lang. Process.*, vol. 30, pp. 829–852, Oct. 2020, doi: 10.48550/arxiv.2010.00475.

[8] K. E. Frasier, "A machine learning pipeline for classification of cetacean echolocation clicks in large underwater acoustic datasets," *PLoS Comput. Biol.*, vol. 17, no. 12, pp. 1–26, 2021, doi: 10.1371/JOURNAL.PCBI.1009613.

[9] K. E. Frasier, M. A. Roch, M. S. Soldevilla, S. M. Wiggins, L. P. Garrison, and J. A. Hildebrand, "Automated classification of dolphin echolocation click types from the Gulf of Mexico," *PLOS Comput. Biol.*, vol. 13, no. 12, p. e1005823, Dec. 2017, doi: 10.1371/JOURNAL.PCBI.1005823.

[10] M. Ferrari, H. Glotin, R. Marxer, and M. Asch, "DOCC10: Open access dataset of marine mammal transient studies and end-to-end CNN classification," *Proc. Int. Jt. Conf. Neural Networks*, Jul. 2020, doi: 10.1109/IJCNN48605.2020.9207085.

[11] A. Shah, A. Kumar, A. G. Hauptmann, and B. R. Fellow, "A Closer Look at Weak Label Learning for Audio Events," *CoRR*, Apr. 2018, doi: 10.48550/arxiv.1804.09288.

[12] N. Turpault, R. Serizel, and E. Vincent, "Limitations of Weak Labels for Embedding and Tagging," *ICASSP, IEEE Int. Conf. Acoust. Speech Signal Process. - Proc.*, vol. 2020-May, pp. 131–135, May 2020, doi: 10.1109/ICASSP40776.2020.9053160.

[13] S. Hershey *et al.*, "The Benefit Of Temporally-Strong Labels In Audio Event Classification," *ICASSP, IEEE Int. Conf. Acoust. Speech Signal Process. - Proc.*, vol. 2021-June, pp. 366–370, May 2021, doi: 10.48550/arxiv.2105.07031.

[14] N. Quadrianto, A. J. Smola, T. S. Caetano, and Q. V. Le, "Estimating Labels from Label Proportions," *J. Mach. Learn. Res.*, pp. 2349–2374, 2009, Accessed: Jun. 09, 2022. [Online]. Available: https://www.jmlr.org/papers/v10/quadrianto09a.html

[15] A. Kumar and B. Raj, "Audio Event Detection using Weakly Labeled Data," *MM 2016 - Proc. 2016 ACM Multimed. Conf.*, pp. 1038–1047, May 2016, doi: 10.1145/2964284.2964310.

[16] A. Kumar and B. Raj, "Weakly Supervised Scalable Audio Content Analysis," *Proc. - IEEE Int. Conf. Multimed. Expo*, vol. 2016-August, Jun. 2016, doi: 10.48550/arxiv.1606.03664.

[17] Y. Xu, Q. Kong, Q. Huang, W. Wang, and M. D. Plumbley, "Attention and Localization based on a Deep Convolutional Recurrent Model for Weakly Supervised Audio Tagging," *Proc. Annu. Conf. Int. Speech Commun. Assoc. INTERSPEECH*, vol. 2017-August, pp. 3083–3087, Mar. 2017, doi: 10.48550/arxiv.1703.06052.

[18] G. L. Turin, "An introduction to matched filters," *IRE Trans. Inf. Theory*, vol. 6, no. 3, pp. 311–329, 1960, doi: 10.1109/TIT.1960.1057571.

[19] J. R. Potter, D. K. Mellinger, and C. W. Clark, "Marine mammal call discrimination using artificial neural networks," *J. Acoust. Soc. Am.*, vol. 96, p. 2636, 1994, doi: 10.1121/1.410274.

[20] D. K. Mellinger and C. W. Clark, "Recognizing transient low-frequency whale sounds by spectrogram correlation," *J. Acoust. Soc. Am.*, vol. 107, p. 3518, 2000, doi: 10.1121/1.429434.

[21] T. Lu, B. Han, and F. Yu, "Detection and classification of marine mammal sounds using AlexNet with transfer learning," *Ecol. Inform.*, vol. 62, p. 101277, May 2021, doi: 10.1016/J.ECOINF.2021.101277.

[22] Y. Shiu *et al.*, "Deep neural networks for automated detection of marine mammal species," *Sci. Reports 2020 101*, vol. 10, no. 1, pp. 1–12, Jan. 2020, doi: 10.1038/s41598-020-57549-y.

[23] NOAA Pacific Islands Fisheries Science Center, "Hawaiian Islands Cetacean and Ecosystem Assessment Survey (HICEAS) towed array data. Edited and annotated for DCLDE 2022," *NOAA National Centers for Environmental Information*. 2022. doi: https://doi.org/10.25921/e12p-gj65.

[24] D. Risch, T. Norris, M. Curnock, and A. Friedlaender, "Common and Antarctic Minke Whales: Conservation Status and Future Research Directions," *Front. Mar. Sci.*, vol. 0, p. 247, May 2019, doi: 10.3389/FMARS.2019.00247.

[25] J. N. Oswald, W. W. L. Au, and F. Duennebier, "Minke whale (Balaenoptera acutorostrata) boings detected at the Station ALOHA Cabled Observatory," *J. Acoust. Soc. Am.*, vol. 129, no. 5, p. 3353, May 2011, doi: 10.1121/1.3575555.

[26] G. Blanchard, M. Flaska, G. Handy, S. Pozzi, and C. Scott, "Classification with Asymmetric Label Noise: Consistency and Maximal Denoising," *Electron. J. Stat.*, vol. 10, no. 2, pp. 2780–2824, Mar. 2013, doi: 10.48550/arxiv.1303.1208.

[27] C. Esposito, G. A. Landrum, N. Schneider, N. Stiefl, and S. Riniker, "GHOST: Adjusting the Decision Threshold to Handle Imbalanced Data in Machine Learning," *J. Chem. Inf. Model.*, vol. 61, no. 6, pp. 2623–2640, Jun. 2021, doi: 10.1021/ACS.JCIM.1C00160/ASSET/IMAGES/LARGE/CI1C00160_0015.JPEG.

[28] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," *32nd Int. Conf. Mach. Learn. ICML 2015*, vol. 1, pp. 448–456, Feb. 2015, doi: 10.48550/arxiv.1502.03167.

[29] D. P. Kingma and J. L. Ba, "Adam: A Method for Stochastic Optimization," *3rd Int. Conf. Learn. Represent. ICLR 2015 - Conf. Track Proc.*, Dec. 2014, doi: 10.48550/arxiv.1412.6980.

DG-MIX: DOMAIN GENERALIZATION FOR ANOMALOUS SOUND DETECTION BASED ON SELF-SUPERVISED LEARNING

Ismail Nejjar[1,2]*, Jean Meunier-Pion*[1,3]*, Gaetan Frusque*[1]*, Olga Fink*[1]*,*

[1] EPFL, IMOS, Lausanne, Switerzland, {ismail.nejjar, gaetan.frusque, olga.fink}@epfl.ch
[2] ETH Zürich, Chair of Intelligent Maintenance Systems, Zürich, Switerzland, {inejjar}@ethz.ch
[3] CentraleSupélec, Gif-sur-Yvette, France, {jean.meunier-pion}@student-cs.fr

ABSTRACT

Detecting anomalies in sound data has recently received significant attention due to the increasing number of implementations of sound condition monitoring solutions for critical assets. In this context, changing operating conditions impose significant domain shifts resulting in performance drops if a model trained on a set of operating conditions is applied to a new operating condition. An essential challenge is distinguishing between anomalies due to faults and new operating conditions. Therefore, the high variability of operating conditions or even the emergence of new operating conditions requires algorithms that can be applied under all conditions. Therefore, domain generalization approaches need to be developed to tackle this challenge. In this paper, we propose a novel framework that leads to a representation that separates the health state from changes in operating conditions in the latent space. This research introduces DG-Mix (Domain Generalization Mixup), an algorithm inspired by the recent Variance-Invariance-Covariance Regularization (VICReg) framework. Extending the original VICReg algorithm, we propose to use Mixup between two samples of the same machine type as a transformation and apply a geometric constraint instead of an invariance loss. This approach allows us to learn a representation that distinguishes between the operating conditions in an unsupervised way. The proposed DG-Mix enables the generalization between different machine types and diverse operating conditions without an additional adaptation of the hyperparameters or an ensemble method. DG-Mix provides superior performance and outperforms the baselines on the development dataset of DCASE 2022 challenge task 2. We also demonstrate that training using DG-Mix and then fine-tuning the model to a specific task significantly improves the model's performance.

Index Terms— Unsupervised Learning, Mixup, Domain Generalization, Self-Supervised Learning, Anomalous Sound Detection

1. INTRODUCTION

Anomalous sound detection (ASD) is the task of identifying whether the sound emitted by a machine is normal or abnormal [1, 2, 3]. One of the main challenges of this task is to distinguish between novel operating conditions or novel background noise and real anomalies caused by a machine fault or malfunction. Moreover, the sound emitted by machines of the same type but operated differently or installed under different conditions may differ significantly. Deep learning models have recently demonstrated excellent performance in detecting abnormal sounds under different scenarios. The directions pursued to tackle these challenges range from Unsupervised Anomalous sound detection [4, 5, 6], where only normal sound samples are used for training, to domain adaptation techniques for bridging domain shifts [7]. Domain shifts are discrepancies in the acoustic signals between a source and a target domain mainly caused by differences in machine operating conditions or ambient noise. The shifts result in performance drops if a model trained on a set of operating conditions is applied to a new operating condition.

In real-world applications, the background noise of the machine can be affected by various sound sources surrounding the machine. Therefore, it is difficult to identify the distinct causes of the changes and attribute them to the domain shift. Consequently, it is necessary to develop a method that can be generalized to different changes in operating conditions without relying on the detection of domain shifts.

In this paper, we propose a novel algorithm DG-Mix (Domain Generalization Mixup), which aims to learn representations that distinguish between the different operating and recording conditions in an unsupervised manner. Three objectives are thereby pursued: (1) reveal the impact of attributes on the data by enforcing embeddings in the same batch to be different, (2) obtain uncorrelated embedding features containing specific information, (3) respect defined geometrical constraints between the different domains. We also investigate how self-supervised learning pre-training helps our model to learn more robust and more general representations that generalize across various operating conditions for different machine types, and are robust to different noise levels and noise types. To this end, our proposed algorithm is compared on the one hand to another popular self-supervised approach, VICReg, and on the other hand to our proposed model trained from scratch. We evaluated the proposed approach and submitted it to task 2, "Unsupervised anomalous sound detection (ASD) for machine condition monitoring applying domain generalization techniques" of the DCASE challenge 2022 [8]. In experimental evaluations, it is shown that the proposed technique significantly outperforms both baseline approaches but also the originally proposed VICReg on the source and target domains of the development set.

2. DATASET

The dataset of this task was generated from the MIMII DG [9] (Malfunctioning Industrial Machine Investigation and Inspection for Domain Generalization) and ToyADMOS2 [10] (Anomaly Detection in Machine Operating Sounds) datasets consisting of normal and anomalous operating sounds of seven types of toy/real machines. ToyCar and ToyTrain machine types are extracted from ToyADMOS2 dataset while fan, gearbox, bearing, slide rail, and valve are extracted from MIMII DG dataset. Each recording is a single-

channel 10-second audio signal sampled at 16 kHz. The signals originate from a mixture between machine sounds and environmental noise samples of several real-world factories. Each machine type contains six sections. The dataset is divided into a development set of three sections and an evaluation set consisting of the other three sections. For each section, there are 1000 training samples, including 990 source samples and ten target samples. In this research, we use all the training data in the development dataset and the additional training dataset for training the models.

3. METHOD

We propose a new framework inspired by the Variance-Invariance-Covariance Regularization (VICReg), a self-supervised algorithm proposed in [11]. We give an overview of our proposed approach in Figure 1. The framework is composed of (1) a self-supervised learning algorithm, (2) a subsequent fine-tuning step and (3) an anomaly detection phase based on k-Nearest Neighbors (k-NN) [12].

The objective of the self-supervised task is to learn an encoder that provides meaningful representations of audio samples. This pre-trained encoder is then used and fine-tuned to perform supervised classification of the section ID. Finally, we use the embeddings from the encoder to compute an anomaly score with k-NN.

3.1. Audio Prepocessing

To simplify the task, all audio samples are provided with only one channel. Each audio sample is then transformed into a log-mel spectrogram. The input given to our model is a two-dimensional image-like feature $X \in \mathcal{R}^{P \times F}$. The frame size of the Short-Time Fourier Transform (STFT) is 64 ms, and the hop size is 32 ms. We also set the number of Mel bins F to 128. The number of frames of the context window P is fixed to 64. The context window is shifted by L frames resulting in B extracted images, with $B = \left\lceil \frac{T-P}{L} \right\rceil$ with $L = 8$. Given the previous parameters, the total spectrogram size T is equal to 313.

3.2. CNN Architectures

We used the MobileNetV2 [13] backbone trained from scratch in this work. The off-the-shelf Pytorch [14] implementation of MobileNetV2 is used. The width multiplier parameter is set to 0.5, and the last layers are adapted to obtain a 320-dimensional vector per input image. Table 1 provides a detailed summary of the applied CNN architecture, with a total parameter number of 1.45M.

3.3. DG-Mix : Self-Supervised Pre-training

The first step of our approach is based on a self-supervised learning algorithm inspired by VICReg [11]. This framework provides a good feature representation for image classification problems.

Background: VICReg is an algorithm based on Siamese networks. VICReg aims to prevent a collapse by regularising the variance and covariance of the network outputs. The objective function of VICReg contains three main terms: a variance term, an invariance term, and a covariance term.

1. **Variance**: Regularization term that prevents mode collapse
2. **Covariance**: Regularization term that prevents dimensional collapse.
3. **Invariance**: Similarity metric to be minimized between two augmented views of the same source image.

Input	Operator	t	c	n	s
$128 \times 64 \times 3$	conv2d 3×3	-	16	1	2
$64 \times 32 \times 16$	bottleneck	1	8	1	1
$64 \times 32 \times 8$	bottleneck	6	16	2	2
$32 \times 16 \times 16$	bottleneck	6	16	3	2
$16 \times 8 \times 16$	bottleneck	6	32	4	2
$8 \times 4 \times 32$	bottleneck	6	48	3	1
$8 \times 4 \times 48$	bottleneck	6	80	3	2
$4 \times 2 \times 80$	bottleneck	6	160	1	1
$4 \times 2 \times 160$	conv2d 1×1	-	320	1	1
$4 \times 2 \times 320$	conv2d 4×2	-	320	1	1
$1 \times 1 \times 320$	conv2d 1×1	-	320	1	

Table 1: Modified MobileNetV2 architecture used for all experiments. Each row represents the sequence of layers, repeated n times, with c channels, and stride s

Proposed Approach: Figure 1 provides an overview of DG-MIX. The proposed loss comprises three parts. The last two terms correspond to the variance and covariance losses presented in the VICReg [11] implementation, while we propose to substitute the third part (representing the invariance criterion in VICReg) with a term that enforces the embedding features of the source and target domains to be distinguishable. Thereby, we are able to learn a specific representation for both domains. For each machine type, we train a Siamese architecture where the three branches are similar and share the same weights. Each branch is composed of an encoder f_θ which corresponds to the modified MobileNetV2 presented in Table 1, followed by an expander h_ϕ. The expander is composed of three fully-connected layers of size 1280. Each of the layers is followed by a batch normalization layer [15] and a ReLU [16] activation function.

In order to mitigate the gap between the source and target domains, we propose to extend the VICReg framework [11] by using Mixup [17] to augment the target domain. Furthermore, a novel loss is proposed to take into account the added Mixup branch, acting as a regularization term and improving domain generalization.

While in the original VICReg approach, a data augmentation approach is applied, we propose to impose similarity between the sample representations. Given all the S log Mel-spectrograms from both the source and target domains of all sections for each machine type, two different samples, X and X' are selected. For each such pair of samples, a linear combination with respect to λ is obtained. This combination gives rise to a new sample denoted as X_λ. Formally, λ is a realization of a beta distribution $Beta(\alpha, \beta)$ and represents the mixup rate. In our case we set $\alpha = \beta = 0.5$.

First X, X' and X_λ are encoded by f_θ resulting in Y, Y' and Y_λ, and then mapped by the expander on the embeddings, Z, Z' and Z_λ. The loss is composed of three terms and computed at the embedding level on Z, Z' and Z_λ. For a batch of size N, we denote $Z = [z_1, ..., z_N]$, with $z \in \mathcal{R}^D$ and D the expander dimension.

The proposed consistency term seeks to generate new virtual domains. A linear interpolations of feature vectors should lead to linear interpolations of their corresponding domain. Therefore the proposed loss aims at minimzing the distance between the embedding vector of X_λ and the linear combination of embeddings of X and X':

$$s(Z_\lambda, Z, Z', \lambda) = \frac{1}{N} \sum_{i=1}^{N} \| z_{\lambda,i} - (\lambda z_i + (1-\lambda) z'_i) \|_2^2 \quad (1)$$

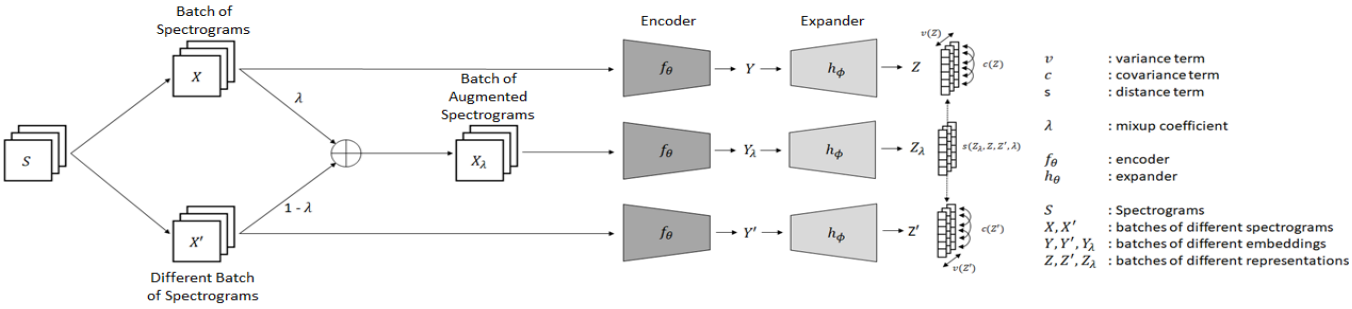


Figure 1: DG-Mix : Self-Supervised Framework for Domain Generalization

The second loss term forces the variance inside each batch to be equal to 1, preventing a mode collapse:

$$v(Z) = \frac{1}{D}\sum_{i=1}^{D} max(0, 1 - S(z^j)) \quad (2)$$

where z^j is the jth row of the matrix Z in the batch, and $S(z) = \sqrt{Var(z)}$ is the standard deviation.

Finally, the last loss term aims to learn uncorrelated features for each embedding, by forcing the off-diagonal elements to be zero, resulting in a rich embedding. The covariance matrix is defined as:

$$C(Z) = \frac{1}{N-1}\sum_{i=1}^{N}(z_i - \bar{z})(z_i - \bar{z})^T \quad (3)$$

with $\bar{z} = \frac{1}{N}\sum_{i=1}^{N} z_i$ representing the mean embedding over a mini-batch.

$$c(z) = \frac{1}{d}\sum_{i \neq j}[C(Z)]_{i,j}^2 \quad (4)$$

The final loss used to train the model is:

$$\gamma s(Z_\lambda, Z, Z', \lambda) + \mu(v(Z) + v(Z')) + \nu(c(Z) + c(Z')) \quad (5)$$

γ, μ, ν are hyper-parameters that we set in this report to $25, 25, 1$ respectively. The networks are trained using the Large Batch Training of Convolutional Networks (LARS) [18] optimizer, with a learning rate of 0.8, weight decay of 10^{-4} and a batch size of 1024 for 100 epochs. In addition, ten warmup epochs were used, and the learning rate followed a cosine decay schedule starting from 0 and finishing at 0.002. The expander size and the hyperparameters were not finetuned on the task but rather taken as reported in the VICReg paper. After the pre-training, only the encoder model was used for the downstream classification task presented in the next section.

3.4. Fine-tuning on Section ID Classification

Similar to the baseline method, our proposed model is fine-tuned to identify the section ID of an audio sample.

The pre-trained encoder is used, and a classifier composed of two fully connected layers (320-128-6) is added. To improve the robustness of the model, a mixup strategy on the source and target data for each section is used to generate augmented data of intra-domain and inter-domain samples.

The KL-divergence loss between the classifier output and the mixed section ID is used for this task along with the geometrical constraint presented in equation 1 as a regularization term. However, this time it is directly applied to the encoder. Finally, the networks are fine-tuned using AdamW [19] optimizer, with a learning rate of 10^{-4}, weight decay of 10^{-4}, and a batch size of 64.

3.5. Anomaly Detection

After the fine-tuning step, we apply a k-NN algorithm [12] to compute the anomaly score. We use the mean embedding vector from the 10-s audio recording as input feature to the k-NN algorithm.

We used the Euclidean metric as the anomaly score, and the number of nearest neighbors was set to 1. In other words, the larger the distance from the training embeddings, the more abnormal the sample is.

4. RESULTS

We tested our model on the development set of the DCASE 2022 Task 2 described in section 2. We first present how our proposed method without pre-training already outperforms the baseline. The main differences between the baseline and our proposed method are (1) change of the CNN, (2) use of Mixup for data augmentation, (3) addition of a regularization term from the Mixup transformation, and (4) use of k-NN for anomaly score computation. Table 2 displays the harmonic means of the AUC Source, AUC Target, and pAUC computed over all three sections for each machine type using the baselines and the proposed method without pre-training. The harmonic means of the AUC Source, AUC Target, and pAUC are also reported for each method.An absolute improvement of more than 10% of the baseline approach is obtained.

The importance of pre-training becomes apparent from the results reported in Table 3 where the performance of two self-supervised approaches, VICReg using SpecAugment [20] and the proposed DG-Mix method is compared. Moreover, we use the SpecAugment procedure for VICReg because Mixup is not adapted for this method. VICReg provides an improvement of almost 1% in terms of the overall harmonic mean and DG-Mix has a performance gain of 4%, suggesting that they both successfully helped to get better and more robust representations.

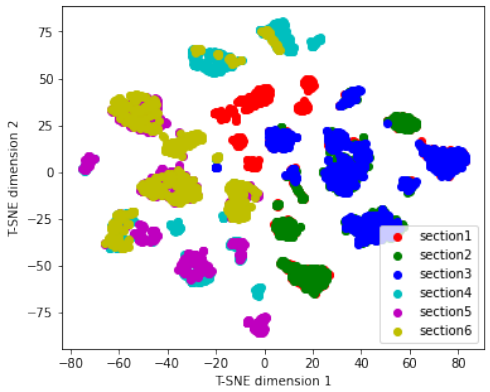
Pre-training with DG-Mix outperforms by 3% the pre-training obtained using VICReg. This is due to Mixup and its regularization term that provide a richer representation of the dataset. This is illustrated in Figures 2a and 2b presenting the t-SNE [21] plots of the embeddings obtained from a ToyCar training sample with DG-Mix and VICReg. There are more clusters with DG-Mix, showing more granularity and a better separation of the dataset into different modalities.

	Baseline MobileNet-V2				Proposed anomaly detection method without pre-training			
Machine	AUC Source	AUC Target	pAUC	Harmonic Mean	AUC Source	AUC Target	pAUC	Harmonic Mean
ToyCar	58,97	52,26	52,39	54,37	**79,82**	**73,62**	**60,11**	**70,18**
ToyTrain	**58,59**	46,07	**51,56**	51,57	46,45	**61,25**	51,43	**52,36**
Bearing	**62,88**	61,81	**57,35**	**60,58**	59,24	**69,06**	50,13	58,47
Fan	71,35	48,53	57,10	57,54	**88,85**	**70,27**	**69,45**	**75,22**
Gearbox	69,98	56,60	56,18	60,29	**79,83**	**70,32**	**60,78**	**69,44**
Slider	66,03	40,72	54,77	51,76	**90,70**	**66,02**	**64,89**	**72,15**
Valve	67,75	58,01	**62,70**	62,57	**71,74**	**65,99**	59,05	**65,18**
Overall	64.73	51.06	55.80	56.65	**72.71**	**70.93**	**62.27**	**68.32**

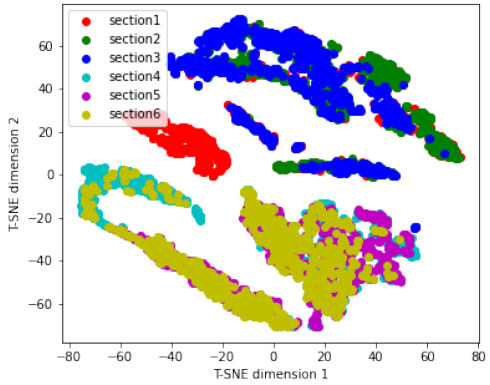
Table 2: Results for the MobileNet-V2 baseline and our proposed anomaly detection method without pre-training (in %).

	VICReg with SpecAugment				DG-Mix			
Machine	AUC Source	AUC Target	pAUC	Harmonic Mean	AUC Source	AUC Target	pAUC	Harmonic Mean
ToyCar	85.09	78,68	59,05	72.47	**93.28**	**81.73**	**68.32**	**79.80**
ToyTrain	48,34	**56,20**	**53,69**	52,54	**52.50**	55.48	53.20	**53.70**
Bearing	**77,99**	70,49	**73,22**	**73,77**	66.99	**82.90**	64.75	70.70
Fan	**83,83**	**77,17**	65,93	74,9	83.37	76.73	**70.74**	**76.60**
Gearbox	**94,99**	68,77	68,30	75,55	88.96	**82.74**	**71.22**	**80.28**
Slider	**94,99**	**68,74**	68,25	**75,51**	94.52	67.35	**68.91**	75.11
Valve	75,02	66,48	60,60	66,86	**85.09**	**81.93**	**71.65**	**79.13**
Overall	76.44	68.78	63.53	69.18	**77.55**	**74.08**	**66.33**	**72.34**

Table 3: Results obtained when pre-training with VICReg compared to DG-Mix (in %).



(a) t-SNE plot of the embedding result when Pre-training with DG-Mix



(b) t-SNE plot of the embedding result when Pre-training with VICReg

However, SpecAugment is less suited than Mixup for anomaly detection because it uses time warping, frequency masking, and time masking transformations that can interfere with the anomaly patterns. In contrast, Mixup does not assume anything about possible anomalies but only mixes samples. Using Mixup between and across multiple domains [22] allows us to sample the augmented training data from the heterogeneous Mixup distribution and get a more robust feature extractor at the end, which improves the results.

5. CONCLUSION

In this paper, we proposed a novel sound anomaly detection framework for domain generalization composed of a self-supervised algorithm followed by a supervised task using Mixup on the input log-Mel spectrograms. In this work, our goal was to develop a unified framework that is robust and performs well across all machine types. We used the same hyperparameters for each machine type to achieve this goal. Experimental evaluation shows that the proposed approach significantly outperforms all baseline approaches. In addition, we demonstrated that pre-training an encoder improved the generalization ability of this encoder. Extending the framework to other tasks and datasets is left for future research.

6. ACKNOWLEDGMENTS

This work was supported by the Swiss National Science Foundation under Grant PP00P2_176878

7. REFERENCES

[1] Y. Koizumi, S. Saito, H. Uematsu, Y. Kawachi, and N. Harada, "Unsupervised detection of anomalous sound based on deep learning and the neyman–pearson lemma," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 27, no. 1, pp. 212–224, 2018.

[2] Y. Kawaguchi, R. Tanabe, T. Endo, K. Ichige, and K. Hamada, "Anomaly detection based on an ensemble of dereverberation and anomalous sound extraction," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 865–869.

[3] K. Suefusa, T. Nishida, H. Purohit, R. Tanabe, T. Endo, and Y. Kawaguchi, "Anomalous sound detection based on interpolation deep neural network," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 271–275.

[4] Y. Koizumi, Y. Kawaguchi, K. Imoto, T. Nakamura, Y. Nikaido, R. Tanabe, H. Purohit, K. Suefusa, T. Endo, M. Yasuda, *et al.*, "Description and discussion on dcase2020 challenge task2: Unsupervised anomalous sound detection for machine condition monitoring," *arXiv preprint arXiv:2006.05822*, 2020.

[5] R. Giri, F. Cheng, K. Helwani, S. V. Tenneti, U. Isik, and A. Krishnaswamy, "Group masked autoencoder based density estimator for audio anomaly detection," in *Detection and Classification of Acoustic Scenes and Events Workshop 2020*, 2020.

[6] K. Wilkinghoff, "Sub-cluster adacos: Learning representations for anomalous sound detection," in *2021 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2021, pp. 1–8.

[7] Y. Kawaguchi, K. Imoto, Y. Koizumi, N. Harada, D. Niizumi, K. Dohi, R. Tanabe, H. Purohit, and T. Endo, "Description and discussion on dcase 2021 challenge task 2: Unsupervised anomalous sound detection for machine condition monitoring under domain shifted conditions," *arXiv preprint arXiv:2106.04492*, 2021.

[8] K. Dohi, K. Imoto, N. Harada, D. Niizumi, Y. Koizumi, T. Nishida, H. Purohit, T. Endo, M. Yamamoto, and Y. Kawaguchi, "Description and discussion on DCASE 2022 challenge task 2: Unsupervised anomalous sound detection for machine condition monitoring applying domain generalization techniques," *In arXiv e-prints: 2206.05876*, 2022.

[9] K. Dohi, T. Nishida, H. Purohit, R. Tanabe, T. Endo, M. Yamamoto, Y. Nikaido, and Y. Kawaguchi, "MIMII DG: Sound dataset for malfunctioning industrial machine investigation and inspection for domain generalization task," *In arXiv e-prints: 2205.13879*, 2022.

[10] N. Harada, D. Niizumi, D. Takeuchi, Y. Ohishi, M. Yasuda, and S. Saito, "ToyADMOS2: Another dataset of miniature-machine operating sounds for anomalous sound detection under domain shift conditions," in *Proceedings of the 6th Detection and Classification of Acoustic Scenes and Events 2021 Workshop (DCASE2021)*, Barcelona, Spain, November 2021, pp. 1–5.

[11] A. Bardes, J. Ponce, and Y. LeCun, "Vicreg: Variance-invariance-covariance regularization for self-supervised learning," 2022. [Online]. Available: https://hal.inria.fr/hal-03541297/file/vicreg_iclr_2022.pdf

[12] E. Fix and J. L. Hodges, "Discriminatory analysis. nonparametric discrimination: Consistency properties," *International Statistical Review/Revue Internationale de Statistique*, vol. 57, no. 3, pp. 238–247, 1989.

[13] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," 2018. [Online]. Available: https://arxiv.org/abs/1801.04381

[14] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," in *NIPS-W*, 2017.

[15] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," *CoRR*, vol. abs/1502.03167, 2015. [Online]. Available: http://arxiv.org/abs/1502.03167

[16] A. F. Agarap, "Deep Learning using Rectified Linear Units (ReLU)," *arXiv preprint arXiv:1803.08375*, 2018.

[17] H. Zhang, M. Cissé, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond Empirical Risk Minimization," *CoRR*, vol. abs/1710.09412, 2017. [Online]. Available: http://arxiv.org/abs/1710.09412

[18] Y. You, I. Gitman, and B. Ginsburg, "Scaling SGD Batch Size to 32k for ImageNet Training," *CoRR*, vol. abs/1708.03888, 2017. [Online]. Available: http://arxiv.org/abs/1708.03888

[19] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," 2017. [Online]. Available: https://arxiv.org/abs/1711.05101

[20] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "Specaugment: A simple data augmentation method for automatic speech recognition," *arXiv preprint arXiv:1904.08779*, 2019.

[21] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne." *Journal of machine learning research*, vol. 9, no. 11, 2008.

[22] Y. Wang, H. Li, and A. C. Kot, "Heterogeneous domain generalization via domain mixup," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 3622–3626.

FEW-SHOT BIOACOUSTIC EVENT DETECTION AT THE DCASE 2022 CHALLENGE

I. Nolasco[1], *S. Singh*[1], *E. Vidaña-Vila*[2], *E. Grout*[3,4], *J. Morford*[5], *M.G. Emmerson*[6] *F. H. Jensen*[7], *I. Kiskin*[8] *H. Whitehead*[9], *A. Strandburg-Peshkin*[3,4], *L. Gill*[10], *H. Pamuła*[11], *V. Lostanlen*[12], *V. Morfi*[1,13], *D. Stowell*[14]

[1] Centre for Digital Music (C4DM), Queen Mary University of London, London, UK
[2] La Salle, University Ramon Llull, Barcelona, ES
[3] Dept. of Biology & Centre for the Advanced Study of Collective Behaviour, University of Konstanz, DE
[4] Dept. for the Ecology of Animal Societies, Max Planck Institute of Animal Behavior, DE
[5] The Oxford Navigation group, Dept. of Zoology, Oxford University, Oxford, UK
[6] School of Biological and Behavioural Sciences, Queen Mary University of London, London, UK
[7] Biology Dept, Syracuse University, NY, USA
[8] Institute for People-Centred AI, FHMS, University of Surrey, Surrey, UK
[9] School of Science, Engineering and Environment, University of Salford, Manchester, UK
[10] BIOTOPIA Naturkundemuseum Bayern, Munich, DE
[11] AGH University of Science and Technology, Kraków, PL
[12] Nantes Université, École Centrale Nantes, CNRS, LS2N, UMR 6004, F-44000 Nantes, FR
[13] Sonantic Limited, London, UK
[14] Tilburg University, Tilburg, The Netherlands; Naturalis Biodiversity Centre, Leiden, NL

ABSTRACT

Few-shot sound event detection is the task of detecting sound events, despite having only a few labelled examples of the class of interest. This framework is particularly useful in bioacoustics, where often there is a need to annotate very long recordings but the expert annotator time is limited. This paper presents an overview of the second edition of the few-shot bioacoustic sound event detection task included in the DCASE 2022 challenge. A detailed description of the task objectives, dataset, and baselines is presented, together with the main results obtained and characteristics of the submitted systems. This task received submissions from 15 different teams from which 13 scored higher than the baselines. The highest F-score was of 60.2% on the evaluation set, which leads to a huge improvement over last year's edition. Highly-performing methods made use of prototypical networks, transductive learning, and addressed the variable length of events from all target classes. Furthermore, by analysing results on each of the subsets we can identify the main difficulties that the systems face, and conclude that few-show bioacoustic sound event detection remains an open challenge.

Index Terms— Few-shot learning, bioacoustics, sound event detection, DCASE challenge

1. INTRODUCTION

The task of bioacoustic sound event detection refers to the retrieval of animal vocalisations from audio recordings in terms of onset and offset times. It shares a common methodology with other sound event detection (SED) contexts, yet, the application domain of bioacoustics is particularly challenging for SED. Deep learning contributed to overcome some of these difficulties in bioacoustic SED, however it also established strong requirements regarding the amount of annotated data needed [1]. Collecting and annotating a large dataset of animal vocalisations is often not feasible given that species are unequally abundant [2] and may be rarely observed; and audio annotation is costly and time-consuming [3]. In contrast to traditional deep learning approaches that use a large amount of data to train models, few-shot learning tries to build accurate models with very few training data [4]. Few-shot learning is usually studied using N-way-k-shot classification, where N denotes the number of classes and k the number of known examples for each class.

This problem was first evaluated as a task on the DCASE 2021 challenge [5]. This year, the setup and goal remain the same: Given the first 5 events of a target class, can systems detect the subsequent events of the same class in the remaining of the audio recording? Diverse approaches have been used to address the few-shot learning problem for classification. Some use prior knowledge about similarity between sounds by computing embeddings (learnt representation spaces) designed to help discriminate between unseen classes [4], while others exploit prior knowledge about the structure of the data by using augmentation to synthesise new data [6]. Finally, some approaches can learn models with parameters that can be fine-tuned to smaller datasets [7]. More recent works use meta-learning and/or prototypical networks for acoustic few-shot learning [8], [9]. All of the above approaches deal with classification tasks rather than detection. Indeed, SED in a few shot setup is commonly approximated as an audio tagging task and few works have addressed the actual detection of onsets and offsets of events [10]. At last year's task edition, the best ranked system improved over the baseline prototypical approach by applying a transductive inference method and a mutual learning framework designed to make the feature extraction network more task dependent [11]. The overall best results were just below 40% F-score which indicates the difficulty of this task. This year, we added more and diverse datasets, and

increased the task difficulty (dataset diversity); yet the task doubled the amount of participants and the best overall F-score in the evaluation set reached the 60% level. This paper is structured as follows. Section 2 presents the bioacoustic datasets used for developing and evaluating submitted systems. Section 3 presents the two baseline methods proposed for the task, followed by the evaluation procedure. Finally, section 4 presents the results of the submitted systems and a discussion about the overall task and future steps in the field of few-shot bioacoustic event detection.

2. DATASETS

A *development dataset* consists of predefined training and validation sets to be used for system development[1]. The training set contains multi-class temporal annotations, provided for each recording as: positive (POS), negative (NEG) and unknown (UNK). For the validation set only single-class temporal annotations (POS/UNK) were provided for each recording. A separate *evaluation set* was kept for evaluating the performance of the systems[2]. During the challenge, only the first five POS events of the class of interest were provided for each of the recordings. Table 1 presents an overview of all the datasets in the development and evaluation sets.

BirdVox-DCASE-10h (BV): This dataset contains audio files recorded in 2015, during the fall migration season by four different autonomous recording units located in Tompkins County, NY, USA. An expert ornithologist, Andrew Farnsworth, has annotated flight calls from four families of passerines, namely: American sparrows, cardinals, thrushes, and New World warblers. These flight calls have a duration in the range 50–150 ms and a fundamental frequency in the range 2–10 kHz.

Hyenas (HT): Hyenas use a variety of types of vocalizations to coordinate with one another over both short and long distances. Spotted hyenas were recorded on custom-developed audio tags designed by Mark Johnson and integrated into combined GPS/acoustic collars (Followit Sweden AB) by Frants Jensen and Mark Johnson. Collars were deployed on female hyenas of the Talek West hyena clan at the MSU-Mara Hyena Project (directed by Kay Holekamp) in the Masai Mara, Kenya as part of a multi-species study on communication and collective behaviour. The recordings contain 5 different vocalisations which were identified and classified into types based on the established hyena vocal repertoire [12]. Field work was carried out by Kay Holekamp, Andrew Gersick, Frants Jensen, Ariana Strandburg-Peshkin, and Benson Pion; labelling was done by Kenna Lehmann and colleagues.

Meerkats (MT, ME): Meerkats are a highly social mongoose species that live in stable social groups and use a variety of distinct vocalisations to communicate and coordinate with one another. Recordings of meerkats were acquired at the Kalahari Meerkat Project (Kuruman River Reserve, South Africa; directed by Marta Manser and Tim Clutton-Brock), as part of a multi-species study on communication and collective behaviour. Recordings of the training set (MT) were recorded on small audio devices (TS Market, Edic Mini Tiny+ A77, 8 kHz) integrated into combined GPS/audio collars which were deployed on multiple members of meerkat groups. Recordings of the validation set (ME) were recorded by an observer following a focal meerkat with a Sennheiser ME66 directional microphone (44.1 kHz) from a distance of less than 1m. Recordings were carried out during daytime hours while meerkats were primarily foraging and include several different call types. Field work was carried out by Ariana Strandburg-Peshkin, Baptiste Averly, Vlad Demartsev, Gabriella Gall, Rebecca Schaefer and Marta Manser. Audio recordings were labelled by Baptiste Averly, Vlad Demartsev, Ariana Strandburg-Peshkin, and colleagues.

Jackdaws (JD): Jackdaws are corvid songbirds that usually breed, forage and sleep in large groups.In a multi-year field study (Max-Planck-Institute for Ornithology, Seewiesen, Germany), wild jackdaws were equipped with small backpacks containing miniature voice recorders (Edic Mini Tiny A31, TS-Market Ltd., Russia) to investigate the vocal behaviour of individuals interacting with their group and behaving freely in their natural environment. Field work was conducted by Lisa Gill, Magdalena Pelayo van Buuren and Magdalena Maier. Sound files were annotated by Lisa Gill.

Western Mediterranean Wetlands Bird Dataset (WMW): Contains bird sounds from 20 endemic species that are typically inhabitants of the "Aiguamolls de l'Empordà" natural park in Girona, Spain. The audio files that compose this dataset were originally retrieved from the Xeno-Canto portaland were manually cleaned and labelled by Juan Gómez-Gómez, Ester Vidaña-Vila and Xavier Sevillano using the Audacity software [13].

HumBug (HB): Mosquitoes produce sound both as a by-product of their flight and as a means for communication and mating. Fundamental frequencies vary in the range of 150 to 750 Hz [14]. As part of the HumBug project, acoustic data was recorded with a high specification field microphone (Telinga EM-23) coupled with an Olympus LS-14. The recordings used in this challenge are a subset of the datasets marked as *'OxZoology'* and *'Thailand'* from HumBugDB [15][3]. The recordings contain the sound of lab-cultured *Culex quinquefasciatus* mosquitoes from Oxford, UK, and various species captured in the wild in Thailand, placed into plastic cups.[16].

Polish Baltic Sea bird flight calls (PB): The PB dataset consists of bird flight calls recordings from Hanna Pamuła's project focused on the acoustic monitoring of birds migrating at night along the Polish Baltic Sea coast. Three autonomous recording units (Song Meters SM2, Wildlife Acoustics, Inc) were deployed close to each other ($<$100m) near a lake, on the dune, and on a forest clearing. The passerines night flight calls were annotated by Hanna Pamuła. Target classes belong to: song thrush, *Turdus philomelos* and blackbird, *Turdus merula*. Event lengths vary between 8 to 400 milliseconds and the usual fundamental frequency range for calls is 5 to 9 kHz.

Transfer-Exposure-Effects dataset (CHE): This dataset comes from the Transfer Exposure-Effects (TREE) research project[4], Data were collected using unattended acoustic recorders (Songmeter 3) in the Chornobyl Exclusion Zone (CEZ) to capture the Chornobyl soundscape and investigate the longterm effects of the nuclear plower plant accident on the local ecology. The fieldwork was designed and undertaken by Mike Wood (University of Salford), Nick Beresford (UK Centre for Ecology & Hydrology) and Sergey Gashchak (Chornobyl Center). Common Chiffchaff (Phylloscopus collybita) and Common Cuckoo (Cuculus canorus) vocalisations were manually annotated and labelled from these recordings by Helen Whitehead.

BIOTOPIA Dawn Chorus (DC): The Dawn Chorus project is a worldwide citizen science and arts project bringing together amateurs and experts to experience and record the dawn chorus at their doorstep.The DC dataset stems from dawn chorus recordings,

[1]Dev set: https://doi.org/10.5281/zenodo.6012309
[2]Eval set: https://doi.org/10.5281/zenodo.6517413
[3]https://github.com/HumBug-Mosquito/HumBugDB/
[4]https://tree.ceh.ac.uk/

	Dataset	mic type	# audio files	total duration	# labels (excl. UNK)	# events
Development Set: Training	BV	fixed	5	10 hours	11	9026
	HT	various	5	5 hours	5	611
	MT	animal mounted	2	70 mins	4	1294
	JD	mobile	1	10 mins	1	357
	WMW	various	161	5 hours	26	2941
Development Set: Validation	HB	handheld	10	2.38 hours	1	712
	PB	fixed	6	3 hours	2	292
	ME	animal mounted	2	20 mins	2	73
Evaluation Set	CHE	fixed	18	3 hours	3	2550
	DC	fixed	10	95 mins	3	967
	CT	handheld	3	48 mins	3	365
	MS	fixed	4	40 mins	1	1087
	QU	animal mounted	8	74 mins	1	3441
	MGE	fixed	3	32 mins	2	1195

Table 1: Information on each dataset.

made using Zoom H2 recorders at 44100 Hz, at three different locations in Southern Germany (Haspelmoor, Munich's Nymphenburg Schlosspark, and Nantesbuch), by Moritz Hertel and Rudi Schleich. The vocalisations of three target species were annotated by Lisa Gill (Common cuckoo, *Cuculus canorus*; European robin, *Erithacus rubecula*; Eurasian wren, *Troglodytes troglodytes*). A challenging aspect of this data is related to recordings being very busy with various other birds vocalising at the same time.

Coati (CT): Coatis are omnivorous diurnal mammals that live in stable social groups ranging from 5 to 30 individuals. The target calls used in this dataset are growls, chitters and chirp-grunts. Several other call types that might be confused with the targets were captured in the recordings which configures the main challenging aspect of this data. Audio recordings were collected from two adult females from the same group on Barro Colorado Island, Panama in March 2020. These individuals wore collars which recorded high resolution GPS data with an external attachment of a small audio recording device (TS Market, Edic Mini Tiny+ A77, 22050 Hz). Audio data were recorded during their active foraging period in daytime hours.Fieldwork was carried out by Emily Grout, Josué Ortega and Ben Hirsch.

Manx Shearwater (MS): Manx shearwaters are procellariform seabirds that breed in dense island colonies in the North Atlantic and winter in the South Atlantic off the South American coast. Adult Manx shearwaters make loud, distinctive vocalisations while present at their breeding colony in various contexts.The target class is Chick begging vocalisations which typically comprise bouts of short, high-pitched 'peeps'. In a multi-year study, Audiomoth recorders were placed in burrows during the breeding season. Fieldwork on Skomer Island was undertaken by various members of the Oxford Navigation Group (OxNav) and annotation was carried out by Joe Morford.

Dolphin Quacks (QU): Bottlenose dolphins are highly acoustic animals with an expansive repertoire of acoustic signals used for social interactions. The target class is Quacks which are short signals (around 100 ms), low-frequencyand emitted at relatively high rates, often with 100s of quacks in a single short vocal bout. The recordings were obtained using sound-and-movement recording DTAGs (Johnson and Tyack 2003) attached with suction cups to bottlenose dolphins by F.Jensen in collaboration with Profs. Peter Tyack, Vincent Janik, and Laela Sayigh. All tags were deployed during routine health assessments conducted by the Sarasota dolphin research project and under a National Marine Fisheries Service research permit to Dr. Randall Wells of Chicago Zoological Society. Individual quacks were labelled by Austin Dziki and validated by F. Jensen.

Chick calls (MGE): Chickens are a precocial bird and upon hatching undergo a process of 'filial imprinting' whereby they establish a strong attachment to their mother. Chicks are active participants in this filial imprinting process and use their calls to signal they are in close proximity to their mother and other family members (i.e. pleasure calls) and to signal distress during social separation in order to solicit maternal contact (i.e. contact calls). The dataset includes three chicks with each chick recorded for 10 minutes; pleasure calls were annotated in recordings from chicks one and two, contact calls were annotated in recordings from chick three. All data was collected by Elisabetta Versace, Shu Wang, and Laura Freeland as part of a project from the Prepared Minds Lab from Queen Mary University of London[5]. All data were annotated by Shu Wang, Laura Freeland, and Michael Emmerson.

3. BASELINE METHODS AND EVALUATION

The baseline systems proposed did not change considerably from last year's edition [5]. Template Matching is based on spectrogram cross-correlation and still commonly used in bioacoustics. This approach scored surprisingly well on last edition evaluation set and thus it remains relevant as a baseline for this task. The second system proposed is based on prototypical networks which remain the state of the art for few-shot learning [4]. The changes from last year's system are the use of a ResNet, and adapting segment size depending on the target class in the query set. These changes mainly address the problem of high variation of event lengths and create a more adaptive system.

The evaluation of this task is based on an event-level F-measure with macro-averaged metric across all classes [5]. A positive match between predicted events and reference is found by applying the Intersection over Union (IoU) with 30% minimum overlap, followed by Hopcroft-Karp-Karzanov algorithm [18] for bipartite graph matching. True Positives (TP), False Positives (FP), and False Negatives (FN) can be computed after the matching step. These are defined as: TP - predicted events that match ground truth events; FP - predicted events that do not match any ground truth events;

[5] https://www.preparedmindslab.org/home

Team code	Code	Eval set: F-score % (95% CI)	Val set F-score %	Main characteristics
Du_NERCSLIP	(A)	60.22 (59.66-60.70)	74.4	CNN+ProtoNet; Frame-level embeddings; PCEN;
Liu_Surrey	(B)	48.52 (48.18-48.85)	50.03	CNN+ProtoNet; extra data; PCEN+$\triangle MFCC$; various post-process.
Martinsson_RISE	(C)	47.97 (47.48-48.40)	60	ResNet+ProtoNet; Ensemble(15) based input size; logMel+PCEN
Hertkorn_ZF	(D)	44.98 (44.44-45.42)	61.76	CNN; Frequency resolution preserving pooling; various post-process
Liu_BIT-SRCB	(E)	44.26 (43.85-44.62)	64.77	CNN+ProtoNet; Transductive inference
Wu_SHNU	(F)	40.93 (40.48-41.30)	53.88	ResNet+ProtoNet; Continual-learning; spectrogram
Zgorzynski_SRPOL	(G)	33.24 (32.69-33.69)	57.2	CNN+Siamese Networks; Emsemble (3) average event-length;
Mariajohn_DSPC	(H)	25.66 (25.40-25.91)	43.89	CNN+ProtoNet; logMel; augmentation with time-shifting and mirroring
Wilbo_RISE	(I)	21.67 (21.32-21.97)	47.94	ResNet+ProtoNEt; Semi-supervised; Melspect+PCEN; various post-process
Zou_PKU	(J)	19.20 (18.88-19.51)	51.99	CNN+protoNet; mutual information loss; time frequency masking + mixup
Huang_SCUT	(K)	18.29 (18.01-18.56)	54.63	Transductive inference + Adapted central difference convolution
Tan_WHU	(L)	17.22 (16.82-17.55)	54.53	CNN+ProtoNet pretrained; transductive inference; task adaptive features
Li_QMUL	(M)	15.49 (15.16-15.77)	47.88	CNN+protoNet; PCEN; time, frequency masking + time warping
baseline-TempMatch	[5]	12.35 (11.52-12.75)	3.37	Spectrogram Cross correlation
baseline-ProtoNet	[5]	5.3 (5.1-5.2)	28.45	ResNet+ProtoNet
Zhang_CQU	(N)	4.34 (3.74-4.56)	44.17	CNN+protoNet; Fine tunning with MIMI; PCEN
Kang_ET	(O)	2.82 (2.76-2.87)	-	CNN+ProtoNEt; pretrained ECAPA-TDNN; Fine-tuning; Specaugment

Table 2: F-score results per team (best scoring system) on evaluation and validation sets, and summary of system characteristics. Systems are ordered by higher scoring rank on the evaluation set. These results and technical reports for the submitted systems can be found on task 5 results page [17].

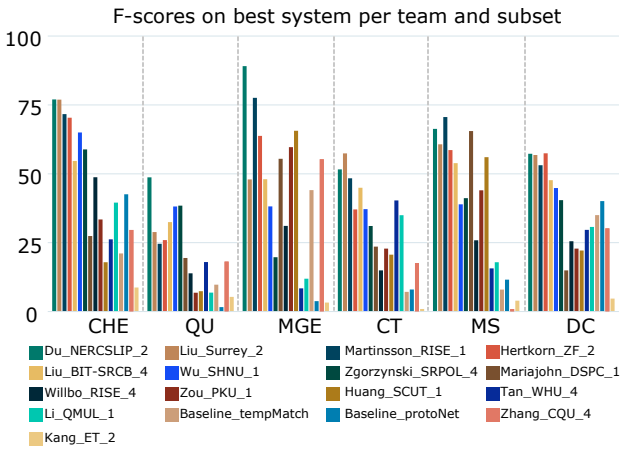


Figure 1: F-Score results by dataset. Systems are ordered by highest scoring rank on the evaluation set.

FN - ground truth events that are not predicted. Matches to UNK events are ignored from these counts as to not negatively impact the systems that predict these events. Finally, the F-score metric is computed per dataset in the evaluation set and the harmonic mean over all is reported.

4. RESULTS

For the 2022 edition, 15 teams participated submitting a total of 46 systems. The results for the highest scoring submission for each team are presented in Table 2, together with the reported F-scores on the validation set and summary of the system characteristics. Fig. 1 presents the F-scores obtained by each team on each subset of the evaluation data. The majority of systems adopted a prototypical network approach. Similar to last year's results, simple improvements over the baselines were achieved by applying data augmentation techniques and intelligent post-processing. Better ways to construct the negative prototype were also explored by some teams who report improved results (B, C, F, I). Transductive inference, the method used by the past edition's winning team, was also applied here by several participants (B, M, L, J). The highest scoring system implements a frame-level embedding learning approach which confers to the system a high time resolution capability (A). The system was particularly effective on the QU and MGE dataset (Fig.1). This confirms that good time precision is fundamental, particularly for classes with events of very short duration as the ones in these datasets. The system ranked in second place implements a novel approach designed to optimise the contrast between positive events and negative prototypes (B). This, together with an adaptive segment length dependent on each target class, works well across all the evaluation sets. The problem of very different lengths of events across target classes was also directly addressed by other submissions. Both (C) and (G) implemented an ensemble approach where each individual model focuses on a different input size range. In (E) this is explored through a multi-scale ResNet, and in (I) with a wide ResNet containing many channels. Finally, it is worth mentioning the system in (D). Their few-shot adaptation was based on fine-tuning alone. The innovation here is related to simple modifications to a CNN-based architecture in order to optimise the use of information, particularly in the frequency axis. Furthermore, by allowing the network to overfit (up to a degree) to the 5 shots, the system achieves surprisingly good performance across all the datasets of the evaluation set.

Overall, this edition saw some novel ideas being implemented that tried to address previously identified challenges related to this task: how to deal with very different event lengths; how to construct a negative class when no explicit labels are given for this; and how to bridge the gap between classification and detection for few-shot sound event detection. We believe these remain relevant questions for our goal and for SED in general, and that the collective work developed here helped pushing few-shot bioacoustic sound event detection into DCASE central stage.

5. REFERENCES

[1] D. Stowell, "Computational bioacoustics with deep learning: a review and roadmap," *PeerJ*, vol. 10, 2022.

[2] W.-P. Vellinga and R. Planqué, "The xeno-canto collection and its relation to sound recognition and classification," in *CLEF (Working Notes)*, 2015.

[3] A. E. Méndez Méndez, M. Cartwright, and J. P. Bello, "Machine–crowd–expert model for increasing user engagement and annotation quality," in *Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems*, 2019, pp. 1–6.

[4] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," *Advances in neural information processing systems*, vol. 30, 2017.

[5] V. Morfi, I. Nolasco, V. Lostanlen, S. Singh, A. Strandburg-Peshkin, L. F. Gill, H. Pamula, D. Benvent, and D. Stowell, "Few-shot bioacoustic event detection: A new task at the dcase 2021 challenge." in *DCASE*, 2021, pp. 145–149.

[6] Y.-X. Wang, R. B. Girshick, M. Hebert, and B. Hariharan, "Low-shot learning from imaginary data," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7278–7286, 2018.

[7] A. Nichol, J. Achiam, and J. Schulman, "On first-order meta-learning algorithms," *ArXiv*, vol. abs/1803.02999, 2018.

[8] Y. Wang, J. Salamon, M. Cartwright, N. J. Bryan, and J. P. Bello, "Few-shot drum transcription in polyphonic music," *CoRR*, vol. abs/2008.02791, 2020.

[9] Y. Shiu, K. Palmer, M. A. Roch, E. Fleishman, X. Liu, E.-M. Nosal, T. Helble, D. Cholewiak, D. Gillespie, and H. Klinck, "Deep neural networks for automated detection of marine mammal species," *Scientific reports*, vol. 10, no. 1, pp. 1–12, 2020.

[10] P. Wolters, C. Daw, B. Hutchinson, and L. Phillips, "Proposal-based few-shot sound event detection for speech and environmental sounds with perceivers," *arXiv preprint arXiv:2107.13616*, 2021.

[11] D. Yang, H. Wang, Z. Ye, and Y. Zou, "Few-shot bioacoustic event detection = a good transductive inference is all you need," DCASE2021 Challenge, Tech. Rep., June 2021.

[12] K. D. S. Lehmann, "Communication and cooperation in silico and nature," Ph.D. dissertation, Michigan State University, 2020.

[13] J. Gómez-Gómez, E. Vidaña-Vila, and X. Sevillano, "Western mediterranean wetlands bird species classification: evaluating small-footprint deep learning approaches on a new annotated dataset," 2022. [Online]. Available: https://arxiv.org/abs/2207.05393

[14] I. Kiskin, D. Zilli, Y. Li, M. Sinka, K. Willis, and S. Roberts, "Bioacoustic detection with wavelet-conditioned convolutional neural networks," *Neural Computing and Applications*, vol. 32, no. 4, pp. 915–927, 2020.

[15] I. Kiskin, M. Sinka, A. D. Cobb, W. Rafique, L. Wang, D. Zilli, B. Gutteridge, R. Dam, T. Marinos, Y. Li, *et al.*, "Humbugdb: a large-scale acoustic mosquito dataset," *arXiv preprint arXiv:2110.07607*, 2021.

[16] Y. Li, I. Kiskin, M. Sinka, D. Zilli, H. Chan, E. Herreros-Moya, T. Chareonviriyaphap, R. Tisgratog, K. Willis, and S. Roberts, "Fast mosquito acoustic detection with field cup recordings: an initial investigation." in *DCASE*, 2018, pp. 153–157.

[17] "DCASE challenge 2022 Few-shot bioacoustic event detection task - results page," accessed: 2022-09-27. [Online]. Available: https://dcase.community/challenge2022/task-few-shot-bioacoustic-event-detection-results

[18] J. E. Hopcroft and R. M. Karp, "An n5/2 algorithm for maximum matchings in bipartite graphs," *SIAM J. Comput.*, vol. 2, pp. 225–231, 1973.

EXPLORING ECO-ACOUSTIC DATA WITH K-DETERMINANTAL POINT PROCESSES

Mohamed Outidrarine[1]*,Pierre Baudet*[1]*,Vincent Lostanlen*[1*]*,*
Mathieu Lagrange[1]*, Juan Sebastián Ulloa*[2]

[1] Nantes Université, École Centrale Nantes, CNRS, LS2N, UMR 6004, F-44000 Nantes, France
[2] Instituto de Investigación de Recursos Biológicos Alexander von Humboldt, Bogotá, Colombia
vincent.lostanlen@ls2n.fr

ABSTRACT

The deployment of acoustic sensor networks in a natural environment contributes to the understanding and the conservation of biodiversity. Yet, the sheer size of audio data which result from these recordings prevents listening them in full. In order to skim through an eco-acoustic corpus, one may typically draw K snippets uniformly at random. In this article, we present an alternative method, based on K-determinantal point processes (K-DPP). This method weights the sampling of K-tuples according to a two-fold criterion of relevance and diversity. To study the eco-acoustics of a tropical dry forest in Colombia, we define relevance in terms of time–frequency second derivative (TFSD) and diversity in terms of scattering transform. Hence, we show that K-DPP offers a better tradeoff than K-means clustering. Furthermore, we estimate the species richness of the K selected snippets by means of the BirdNET birdsong classifier, which is based on a deep neural network. We find that, for $K > 10$, K-DPP and K-means tend to produce a species checklist that is richer than sampling K snippets independently without replacement.

Index Terms— Acoustic sensor networks, Bird species classification, Scattering transform, Time-frequency second derivative

1. INTRODUCTION

The science of eco-acoustics aims to analyze outdoor acoustic scenes so as to characterize certain ecological processes, such as population dynamics, species assemblages, and the emergence of a sonic landscape or "soundscape" [1]. This young field of research contributes to the overarching goal of biodiversity conservation at the planetary scale. Yet, a supervised classifier of eco-acoustic events may only be deployed once defined a research hypothesis, a taxonomy of sounds of interest, and a training set with expert or crowdsourced annotation.

Hence, the mass collection of eco-acoustic signals raises a problem in exploratory data analysis: how to quickly skim through a given corpus of N audio snippets without listening to it in full? In this context, the naive approach consists in drawing a subcorpus $\mathcal{X}$ of $K \ll N$ equiprobable samples. One may refine this method by weighting the probability of drawing each signal $\boldsymbol{x}_i$ in the corpus by a "relevance" prior q_i. Intuitively, the audio feature q_i is designed so as to quantify the saliency of spectrotemporal modulations in the time–frequency domain, such as animal vocalizations. However, this reweighted approach incurs a form of selection bias in terms of species richness: the top-K values of the saliency measure q_i typically belong to much fewer than K distinct species. This is because few species will exhibit calls with high saliency while most species will only exhibit calls with low-to-moderate saliency. The former tend to be over-represented in $\mathcal{X}$, at the detriment of the latter.

In this article, we propose a method for sampling audio signals according to a probabilistic tradeoff between relevance and diversity. The key idea is to represent each $\boldsymbol{x}_i$ by a vector $\boldsymbol{\Phi}_i$ of norm $\sqrt{q_i}$ and such that the angles $\angle(\boldsymbol{\Phi}_i, \boldsymbol{\Phi}_j)$ approximate the auditory dissimilarity of the pair $(\boldsymbol{x}_i, \boldsymbol{x}_j)$. Thus, we draw the subcorpus $\mathcal{X}$ with a probability which is proportional to the determinant of the indexed family $\boldsymbol{\Phi}_i$ of vectors $\boldsymbol{x}_i \in \mathcal{X}$. Because of this proportionality, the proposed method is known as *K-determinantal point process*, or K-DPP for short.

There is a growing body of literature on the topic of applying (K-)DPP to various machine learning problems, such as image search and graph threading in a document collection: we refer to [2] for an introduction. However, no application of K-DPP to machine listening has been published until today. Hence, the novel contribution of our paper is to offer a first proof of concept which demonstrates its interest for diverse sampling in eco-acoustics. A recent article [3] has employed a K-DPP in order to extract a subcorpus of urban acoustic scenes as part of a human annotation campaign; but the authors used a constant relevance term $q_i = 1$ and did not evaluate their approach.

As a first illustration, Figure 1 illustrates different sampling methods with $K = 3$ for eco-acousticc signals from a dry tropical forest. Qualitatively speaking, we notice that uniform independent draws (row 1) tend to lack in relevance, with some samples containing only high-frequency insect stridulations. Relevance weighting (row 2) mitigates this problem at the cost of a reduction in diversity: the most salient species tend to recur disproportionately. Clustering with K-means (row 3) restores diversity at the detriment of relevance. Lastly, K-DPP (row 4) seems to offer an interesting tradeoff between relevance and diversity.

2. DETERMINANTAL POINT PROCESS

2.1. Time–frequency second derivative (TFSD)

We decompose each signal $\boldsymbol{x}_i$, of same duration T, by means of a constant-Q wavelet filters $\boldsymbol{\psi}_{\lambda_1}$, with $\lambda_1 \in \Lambda$. We apply pointwise complex modulus and denote by $\mathbf{U}_1\boldsymbol{x}_i(t, \lambda_1) = \left|\boldsymbol{x}_i * \boldsymbol{\psi}_{\lambda_1}\right|(t)$ the resulting scalogram. representation. Then, we compute the first-order difference of $\mathbf{U}_1\boldsymbol{x}_i$ over both variables t and λ_1. Hence:

$$\begin{aligned}\mathrm{TFSD}(\boldsymbol{x}_i)(t, \lambda_1) = \mathbf{U}_1\boldsymbol{x}_i(t + \tau, \lambda_1 + \delta) + \mathbf{U}_1\boldsymbol{x}_i(t, \lambda_1) \\ - \mathbf{U}_1\boldsymbol{x}_i(t + \tau, \lambda_1) - \mathbf{U}_1\boldsymbol{x}_i(t, \lambda_1 + \delta), \quad (1)\end{aligned}$$

*This work is supported by an Atlanstic 2020 award on "Trainable Acoustic Sensors" (TrAcS).

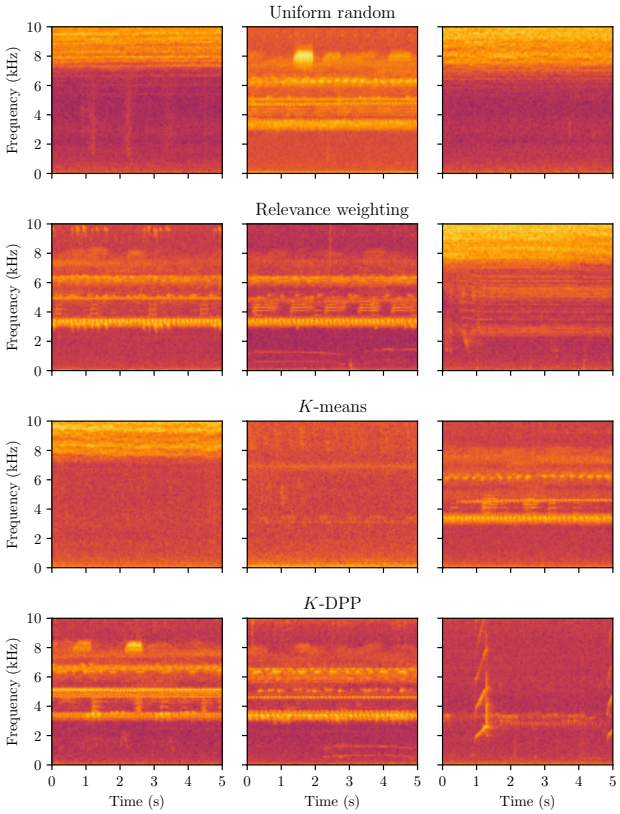


Figure 1: Short-term Fourier spectrograms of three samples from the dataset under study, drawn from four different random sampling methods: uniform, relevance-based, K-means, and K-DPP.

where the acronym TFSD stands for Time–Frequency Second Derivative. In practice, we set the hop size τ to 23 ms, the quality factor to $Q = 3$, the relative frequency interval δ to one third of an octave. We define a region of interest $\Lambda' \subset \Lambda$, corresponding to the third-octave bands λ_1 in $\mathbf{U}\boldsymbol{x}_i$ which range between 2 and 8 kHz. This region corresponds to the vocal range of most singing species of birds. Lastly, we sum the absolute values of the $\mathrm{TFSD}(\boldsymbol{x}_i)$ matrix over both regions Λ and Λ'. Their ratio provides a measure of relevance

$$q_i = \frac{\int_0^T \int_{\Lambda'} \left|\mathrm{TFSD}(\boldsymbol{x}_i)(t,\lambda_1)\right| \,\mathrm{d}t\,\mathrm{d}\lambda_1}{\int_0^T \int_{\Lambda} \left|\mathrm{TFSD}(\boldsymbol{x}_i)(t,\lambda_1)\right| \,\mathrm{d}t\,\mathrm{d}\lambda_1} \tag{2}$$

between zero and one. A recent study in an urban environment [4] has shown that the descriptor q_i (defined with a slight variation in spectrotemporal parameters) significantly correlates with the perceived time of presence of bird vocalizations.

2.2. Time scattering transform

The time scattering transform, also known as deep scattering spectrum [5], is a nonlinear convolutional operator in the time–frequency domain whose role is to extract amplitude modulations at various time scales in each wavelet subband. First, we integrate the scalogram $\mathbf{U}_1\boldsymbol{x}_i$ along time t, which gives the averaged scalogram:

$$\mathbf{S}_1\boldsymbol{x}_i(\lambda_1) = \int_0^T \mathbf{U}_1\boldsymbol{x}_i(t,\lambda_1)\,\mathrm{d}t, \tag{3}$$

also known as first-order scattering. The transformation from $\mathbf{U}_1$ to $\mathbf{S}_1$ guarantees a property of global invariance, which comes at the cost of a loss in discriminability: $\mathbf{S}_1\boldsymbol{x}_i(\lambda_1)$ ignores the amplitude modulations in each wavelet subband $\mathbf{U}_1\boldsymbol{x}_i(t,\lambda_1)$ around its average value, and so for every fixed λ_1. The key idea behind the scattering transform is to recover these amplitude modulations by means of a second filter bank of constant-Q wavelet filters $\boldsymbol{\psi}_{\lambda_2}$, except with a quality factor of $Q = 1$ now. We obtain the so-called amplitude modulation spectrum of $\boldsymbol{x}_i$:

$$\mathbf{U}_2\boldsymbol{x}_i(t,\lambda_1,\lambda_2) = \left|\mathbf{U}_1\boldsymbol{x}_1 * \boldsymbol{\psi}_{\lambda_2}\right|(t,\lambda_1), \tag{4}$$

where the convolution between scalogram and second-order wavelet is performed over time t, and broadcasted across all frequencies λ_1. Symmetrically to $\mathbf{U}_1$ et $\mathbf{S}_1$, we integrate $\mathbf{U}_2$ along time to obtain the matrix:

$$\mathbf{S}_2\boldsymbol{x}_i(\lambda_1,\lambda_2) = \int_0^T \mathbf{U}_2\boldsymbol{x}_i(t,\lambda_1,\lambda_2)\,\mathrm{d}t. \tag{5}$$

We concatenate the first-order coefficients $\mathbf{S}_1\boldsymbol{x}_i$ with the flattened matrix $\mathbf{S}_2\boldsymbol{x}_i$ so as to obtain a high-dimensional vector $\mathbf{S}\boldsymbol{x}_i$, which is generically indexed by the "scattering path" multiindex p, i.e., either the singleton λ_1 or to the pair (λ_1,λ_2) depending on order.

Time scattering approximately verifies a property of energy conservation, similarly to the Parseval identity for the Fourier transform. Therefore, dividing the vector $\mathbf{S}\boldsymbol{x}_i$ by its ℓ^2 norm is tantamount to normalizing the waveform $\boldsymbol{x}_i$ itself.

2.3. Likelihood kernel

We define $\phi_i = \mathbf{S}\boldsymbol{x}_i/\|\mathbf{S}\boldsymbol{x}_i\|_2$ the renormalized vector, and $\boldsymbol{\Phi}_i = \sqrt{q_i}\phi_i$ the vector that is parallel to ϕ_i and has ℓ^2 norm equal to $\sqrt{q_i}$. Our working hypothesis is that constructing a K-DPP with time scattering as the descriptor of choice will preserve the auditory diversity across eco-acoustic samples. We repeat the same operation for every signal $\boldsymbol{x}_i \in \mathcal{X}$.

The likelihood kernel of the associated K-DPP is defined as

$$\mathbf{L}_{i,j} = \langle \boldsymbol{\Phi}_i | \boldsymbol{\Phi}_j \rangle = \sqrt{q_i q_j}\,\langle \phi_i | \phi_j \rangle. \tag{6}$$

Given a set of distinct indices $\boldsymbol{\sigma} = \{\sigma_1 \dots \sigma_K\}$ between 1 and N, we denote by $\mathbf{L}_{\boldsymbol{\sigma}}$ the restriction of the matrix $\mathbf{L}$ to the rows and columns whose indices belong to $\boldsymbol{\sigma}$. Thus, the K-DPP with kernel $\mathbf{L}$ is a random variable over the K-uplets of $1 \dots N$, in which the probability of drawing a specific K-uplet $\boldsymbol{\sigma}$ is proportional to the determinant of the matrix $\mathbf{L}_{\boldsymbol{\sigma}}$:

$$\mathbb{P}[\mathcal{X} = (\boldsymbol{x}_{\sigma_1} \dots \boldsymbol{x}_{\sigma_K})] \propto \det \mathbf{L}_{\boldsymbol{\sigma}}. \tag{7}$$

3. APPLICATION TO ECO-ACOUSTICS

3.1. Protocol

We apply our protocol to a dataset of $N = 432$ audio snippets, which we recorded between February 14th and February 16th in the dry tropical forest of San Jacinto (Bolívar, Colombia) by means

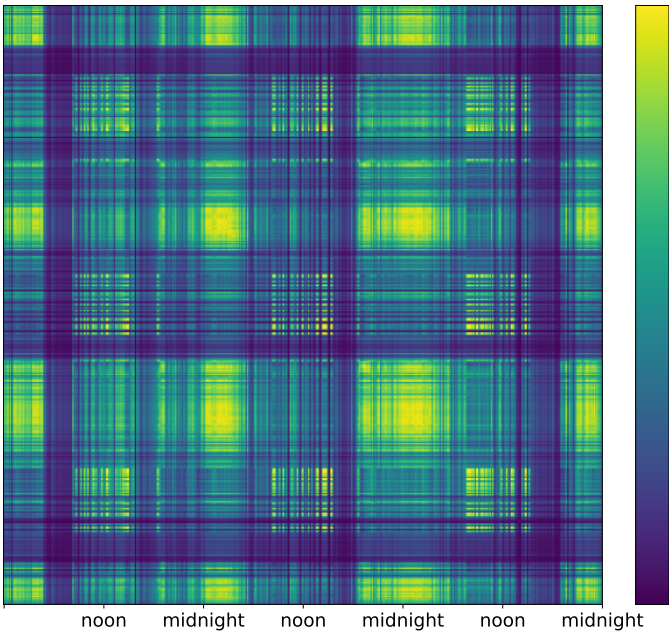


Figure 2: Likelihood kernel $\mathbf{L}$ for the audio snippets under study, arranged in a chronological order over three days. Darker colors denote a greater joint probability of sampling the row snippet and the column snippet as part of the same DPP draw.

of a Wildlife Acoustics "Song Meter 2" acoustic sensor. This sensor is equipped with an omnidirectional microphone and records intermittently at a rate of one five-second snippet every ten minutes, twenty-four hours a day. Such data acquisition campaign belongs to a larger endeavor for biodiversity coordination, which is being coordinated by the Alexander von Humboldt Biological Resources Research Institute [6]. For our experiment, we rely on a Python implementation of K-DPP via an open-source library named DPPy [7] by using the aforementioned criteria of relevance and diversity. We extract the TFSD via the scikit-maad package [8] and the time scattering transform via the Kymatio package [9].

Figure 2 presents the likelihood kernel $\mathbf{L}$ as computed over this eco-acoustic dataset. Note that the diagonal of the matrix $\mathbf{L}$ gives the relevance of observations: for every i, $\mathbf{L}_{i,i} = q_i$. We find that the relevance term, as described by the TFSD eco-acoustic indicator, roughly follows a daily periodicity: it is highest at dawn and dusk, lowest at midnight, and takes irregular values around noon. Future research is necessary to indicate whether this temporal pattern aligns with the chronobiology of vocalizing animals in the site under study.

In Figure 2, we also observe that $\mathbf{L}$ has a block-wise structure, which again, aligns with the daily cycle of animal vocalizations in the forest. Relatedly, we find that the soundscape under study seems to exhibit greater acoustical diversity, as measured by the scattering transform, during the night than during the day.

Mathematically speaking: for snippets $(\boldsymbol{x}_i, \boldsymbol{x}_j)$ which are acoustically similar (e.g., because they are one day apart), their ℓ^2-normalized scattering transforms ϕ_i and ϕ_j are almost colinear. As a result, the inner product $\langle \phi_i | \phi_j \rangle$ is close to one, and the determinant associated to $\sigma = (i, j)$ is:

$$\begin{aligned} \det \mathbf{L}_\sigma &= q_i q_j - \sqrt{q_i q_j} \langle \phi_i | \phi_j \rangle \sqrt{q_j q_i} \langle \phi_j | \phi_i \rangle \\ &= q_i q_j \left(1 - \langle \phi_i | \phi_j \rangle^2\right), \end{aligned} \tag{8}$$

which is nonnegative but close to zero. The subtractive term $-\langle \phi_i | \phi_j \rangle^2$ has a repulsive effect over the pair of snippets $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, of the order of the cosine similarity between features ϕ_i and ϕ_j.

We compare our K-DPP to a naive baseline, which we name "uniform random". The baseline verifies, for each snippet i, the proportionality rule: $\mathbb{P}[\boldsymbol{x}_i \in \mathcal{X}] \propto 1/N$, where N is the total number of audio snippets. In addition, we refine the naive baseline so that the probability of drawing the snippet $\boldsymbol{x}_i$ is proportional to its relevance q_i: $\mathbb{P}[\boldsymbol{x}_i \in \mathcal{X}] \propto q_i$. Under both "uniform random" and "relevance weighting" methods, the probabilistic sampling is made K times independent without replacement. Thirdly, we evaluate a well-known method for unsupervised clustering: namely, K-means. This method produces a partition of the full corpus into K disjoint clusters $\mathcal{C}_1, \ldots, \mathcal{C}_K$ so as to minimize the intra-class variance:

$$\sum_{k=1}^{K} \sum_{\boldsymbol{x}_i \in \mathcal{C}_k} \left\| \mathbf{S}\boldsymbol{x}_i - \frac{1}{\operatorname{card} \mathcal{C}_k} \sum_{\boldsymbol{x}_j \in \mathcal{C}_k} \mathbf{S}\boldsymbol{x}_j \right\|^2. \tag{9}$$

We build the subcorpus $\mathcal{X}$ by selecting, for each cluster $\mathcal{C}_k$, the snippet $\boldsymbol{x}$ whose scattering transform $\mathbf{S}\boldsymbol{x}$ is closest to the Euclidean centroid of all points in $\mathcal{C}_k$. Unlike the first two naive baselines, K-means does not perform independent sampling without replacement: instead, the cluster assignment of every snippet affects the global cost function in Equation 9, and thus indirectly conditions the probability of sampling every other snippet. At the same time, we note that K-means clustering becomes impractical if repeated draws of the subcorpus $\mathcal{X}$ are needed, or if the number of elements K needs to be adjusted dynamically.

3.2. Relevance–diversity tradeoff

We run all four methods with $K = 3$ and repeat them 200 times independently. Figure 3 illustrates our findings. As expected, the uniform random method fares poorly on both metrics of relevance and diversity. Relevance weighting improves relevance but still lacks in diversity. Clustering with K-means is outperformed by relevance weighting on both metrics. Last but not least, we observe that the K-DPP reaches a favorable tradeoff between relevance and diversity: it offers a greater diversity than relevance-weighted sampling while guaranteeing a better relevance than K-means. More precisely, K-DPP triples the diversity of relevance-weighted sampling while retaining about 90% of its relevance on average.

3.3. Species inventory

In the previous subsection, we have verified that the K-DPP method yields diverse and relevant subcorpora, by our predefined measure of relevance (TFDS) and diversity (scattering transform). It remains to be seen whether these definitions are useful in practice for conservation science. For this purpose, we run every snippet $\boldsymbol{x}_i$ through BirdNET, a pretrained convolutional network for species identification [10]. In this way, our evaluation metric is the total number of distinct species in the K-uplet $\mathcal{X}$, also known as "species richness" in ecology. Moreover, we measure the "precision at K"; that is, the proportion of snippets in $\mathcal{X}$ which contain at least one bird vocalization, whatever be its species.

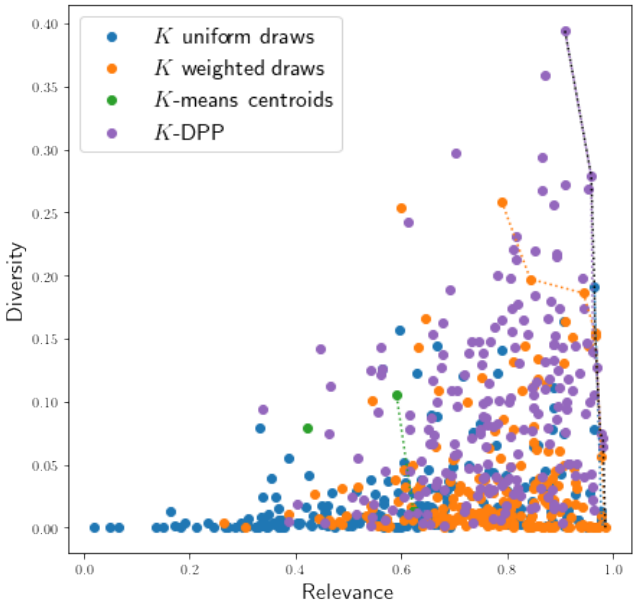


Figure 3: Scatter plot of relevance (x-axis) and diversity (y-axis) of K-uplets from our eco-acoustic dataset, as drawn from various sampling techniques (see legend). Dashed lines indicate the Pareto front for each method. The best K-uplets are in the top-right corner.

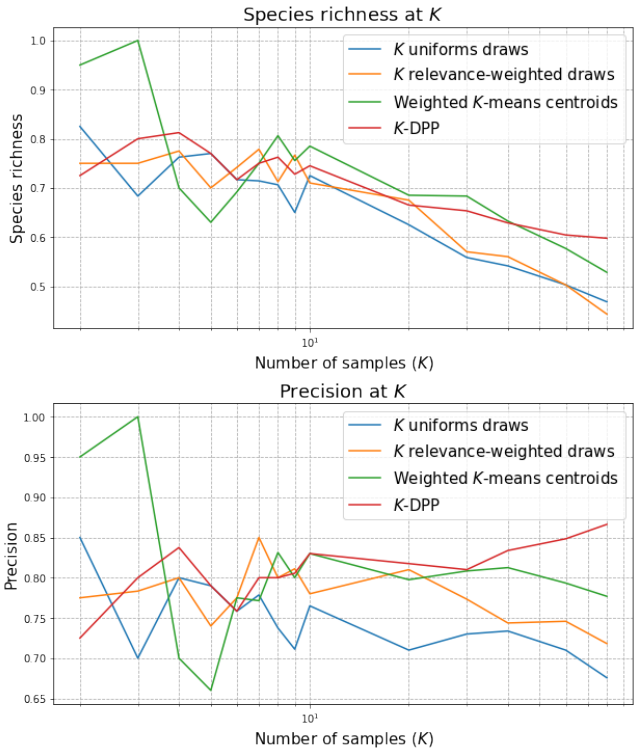


Figure 4: Species richness (top) and precision (bottom) of K-uplets drawn from all four presented methods, for a set cardinal ranging between $K = 3$ and $K = 80$. Higher is better.

Figure 4 illustrates our findings. We see that the naive baseline has a poor species richness and a poor precision. Meanwhile, sampling with K-DPP leads to better results: its K-uplets contain a richer species inventory and fewer false positives. Yet, a surprising result, deserving of further inquiry, is that K-means clustering matches K-DPP in richness, and even outperforms it for very small values of K.

4. CONCLUSION

We have shown that the use of K-determinantal point processes (K-DPP) in eco-acoustics allows to explore and summarize large volumes of audio data while satisfying an interesting tradeoff between relevance and diversity. By means of an off-the-shelf classifier of bird species (BirdNET), we have shown that K-DPP tend to enrich the species inventory of the subcorpus compared to random uniform sampling; and so, particularly for $K > 10$, when there a substantial risk of accidentally drawing near-duplicate snippets.

We note that the choice of a likelihood kernel plays a large role in the success of K-DPP. In our article, this choice was motivated by domain-specific knowledge in eco-acoustics and psycho-acoustics, and was later confirmed by means of a species classifier. Our paper calls attention on the risks underlying the random subsampling of a dataset, especially in the early phase of forming a research hypothesis. Relevance weighting reduces the risk of sampling false positives, yet at the cost of biasing the subcorpus $\mathcal{X}$ towards a narrow range of extremely salient events. Hence, we have advocated for a balanced approach, which takes both relevance and diversity into account. We have presented a first application of K-DPP for exploring biodiversity in a Colombian dry forest, with the hope to encourage more applications of this tool in the future.

5. ACKNOWLEDGMENT

We thank Guillaume Gautier and Rémi Bardenet for their help with the DPPy package.

6. REFERENCES

[1] J. Sueur and A. Farina, "Ecoacoustics: the ecological investigation and interpretation of environmental sound," *Biosemiotics*, vol. 8, no. 3, pp. 493–502, 2015.

[2] A. Kulesza, B. Taskar, *et al.*, "Determinantal point processes for machine learning," *Foundations and Trends in Machine Learning*, vol. 5, no. 2–3, pp. 123–286, 2012.

[3] M. Cartwright, J. Cramer, A. E. M. Mendez, Y. Wang, H.-H. Wu, V. Lostanlen, M. Fuentes, G. Dove, C. Mydlarz, J. Salamon, *et al.*, "SONYC-UST-V2: An urban sound tagging dataset with spatiotemporal context," in *Proc. DCASE*, 2020.

[4] F. Gontier, C. Lavandier, P. Aumond, M. Lagrange, and J.-F. Petiot, "Estimation of the perceived time of presence of sources in urban acoustic environments using deep learning techniques," *Acta Acustica united with Acustica*, vol. 105, no. 6, pp. 1053–1066, 2019.

[5] J. Andén and S. Mallat, "Deep scattering spectrum," *IEEE Transactions on Signal Processing*, vol. 62, no. 16, pp. 4114–4128, 2014.

[6] C. Pizano and H. García, Eds., *El bosque seco tropical en Colombia*. Bogotá, D.C., Colombia.: Instituto de Investigación de Recursos Biológicos Alexander von Humboldt (IAvH), 2014.

[7] G. Gautier, G. Polito, R. Bardenet, and M. Valko, "DPPy: DPP Sampling with Python," *Journal of Machine Learning Research - Machine Learning Open Source Software (JMLR-MLOSS)*, vol. 20, no. 180, pp. 1–7, 2019.

[8] J. S. Ulloa, S. Haupert, J. F. Latorre, T. Aubin, and J. Sueur, "scikit-maad: An open-source and modular toolbox for quantitative soundscape analysis in Python," *Methods in Ecology and Evolution*, vol. 12, no. 12, pp. 2334–2340, 2021.

[9] M. Andreux, T. Angles, G. Exarchakis, R. Leonarduzzi, G. Rochette, L. Thiry, J. Zarka, S. Mallat, J. andén, E. Belilovsky, J. Bruna, V. Lostanlen, M. Chaudhary, M. J. Hirn, E. Oyallon, S. Zhang, C. Cella, and M. Eickenberg, "title=Kymatio: Scattering transforms in Python," *Journal of Machine Learning Research*, vol. 21, no. 60, pp. 1–6, 2020.

[10] S. Kahl, C. M. Wood, M. Eibl, and H. Klinck, "Birdnet: A deep learning solution for avian diversity monitoring," *Ecological Informatics*, vol. 61, p. 101236, 2021.

POLYPHONIC SOUND EVENT DETECTION FOR HIGHLY DENSE BIRDSONG SCENES

Alberto García Arroba Parrilla[1]

a.g.a.p.albertogarciaarrobaparrilla@tilburguniversity.edu

1. Tilburg University, the Netherlands

Dan Stowell[1,2]

d.stowell@tilburguniversity.edu

2. Naturalis Biodiversity Center, Leiden, the Netherlands

ABSTRACT

One hour before sunrise, one can experience the dawn chorus where birds from different species sing together. In this scenario, high levels of polyphony, as in the number of overlapping sound sources, are prone to happen resulting in a complex acoustic outcome. Sound Event Detection (SED) tasks analyze acoustic scenarios in order to identify the occurring events and their respective temporal information. However, highly dense scenarios can be hard to process and have not been studied in depth. Here we show, using a Convolutional Recurrent Neural Network (CRNN), how birdsong polyphonic scenarios can be detected when dealing with higher polyphony and how effectively this type of model can face a very dense scene with up to 10 overlapping birds. We found that models trained with denser examples (i.e., higher polyphony) learn at a similar rate as models that used simpler samples in their training set. Additionally, the model trained with the densest samples maintained a consistent score for all polyphonies, while the model trained with the least dense samples degraded as the polyphony increased. Our results demonstrate that highly dense acoustic scenarios can be dealt with using CRNNs. We expect that this study serves as a starting point for working on highly populated bird scenarios such as dawn chorus or other dense acoustic problems.

Index points - Sound Event Detection, Polyphony, Birdsong, Dense scene

1. INTRODUCTION

Among other domains, acoustic biodiversity studies have flourished as improvements in data science have helped researchers understand, monitor and study many species in detail [1] [2] [3] [4]. Sound event detection tasks have covered many domains where relevant information can be extracted from acoustic scenarios [5] [6] [7]. SED problems can increase in difficulty when many classes are considered and when several events overlap at the same time [8] [9], also known as polyphony. Sound Event Detection (SED) models that can detect overlapping sound events can be addressed as polyphonic SED.

Polyphony is generally defined as the number of sound sources occurring at the same time (i.e., overlapping). In this work we restrict our attention to the number of simultaneous species (i.e., sound types). Polyphony is very common in real-life birdsong scenarios and requires polyphonic SED models to detect all overlapping events. To solve this problem, it requires a model capable of detecting, despite high levels of overlap, all existing species in the studied area. This addresses scenarios like dawn chorus, defined by [10] as a phenomenon around one hour before sunrise with a high activity of birdsong from many individuals and species. This could reach up to 30+ species [11] presenting a very dense scenario for studying and monitoring wildlife.

Our work aims to build a polyphonic SED model capable of dealing with birdsong real-life scenarios. We focus on two research questions (RQs):

- RQ1: What level of polyphony can be successfully detected on a dense audio scene and dense species list?
- RQ2: How accurately can this model detect very dense scenes with up to 10 overlapping sounds?

To address these research questions, several datasets with different polyphony were created and used to train SED models. This work demonstrates that a single neural network can provide, considering the complexity of the task and conditions of the dataset, encouraging results on low-polyphonic scenes and high-polyphonic ones.

2. RELATED WORK

Both convolutional networks and recurrent networks have been developed to deal with SED tasks due to the ability to extract useful features from audio signals (convolution part or CNN) [1] [2] and detect temporal context in those features (recurrence part or RNN) [5] [12], however, a consensus on architecture is clear, as very good results have been obtained applying a commonly used design of CRNN. This consensus has also been reached regarding the input features where spectrograms tend to be the most popular ones [2] [5] [6] [9] [13] [14] [15] [12]. Differences in birdsong are expected to arise from both pitch and timbral variations [16] and thus the spectrogram is generally appropriate for multi-species birdsong recognition.

Reviewing related papers, it can surely be stated that not much research has been carried out in studying the effect of high polyphony in the birdsong audio domain. Some papers study sound event detection addressing polyphony in other kinds of scenes, mostly urban sounds, [6] [9] [14] [17] or using non-specific datasets while others employ avian datasets but without giving much focus on dense polyphonic scenes [1] [3] [4] [18].

The most closely related project is the study of polyphony using a convolutional recurrent network of [15]. The author presents a model that can address birdsong detection with good results for recordings with up to 3 overlapping sounds. However, it has limitations as only a total of 5 species are considered making this approach not very close to a real-life dawn chorus scenario. Also related to birdsong, [19] proposed an HMM-based method

to determine the temporal information of bird sounds without establishing any intrinsic limit on the polyphony. However, this study does not reflect how neural networks work on these problems.

To study dense polyphonic scenes, datasets should include a considerable number of classes, but most of the datasets used in the literature do not get over 16. Many papers have employed data synthesis to force a certain level of polyphony in the samples but if the number of classes is limited, then the polyphony will be too. Meaning that, if a dataset only contains *k* possible classes, the maximum overlapping sounds will be *k*, therefore limiting the polyphonic scenario.

Polyphony approaching a level of 30 or more [11] appears to be beyond the current state of the art, both in datasets and SED systems. Out work builds towards this goal, by training and evaluating an SED system for high-density polyphonic scenes typical in birdsong.

3. METHODS

3.1. Dataset selection

The chosen dataset [20], contains 77 recordings of dawn chorus from the USA contemplating 48 species and a total of 16,052 event activations. Every audio file has a fixed sample rate of 32kHz, fixed length of 5 minutes and has an associated annotation file with detailed information about every event (exact onset, offset and label). Annotation files also contain information about the frequency range (higher and lower values) the audio file is limited to, however, this information is not used in the present experiment as every file is processed without any upper or lower boundaries, and only temporal extents are predicted (start/end time).

A 48-species multi-class multi-label model presents a very complex SED problem, so after studying the distribution of the data, it could be seen that there were unbalanced class sizes, where many have less than 10 annotations in the entire dataset. In this work, we use the data in their empirical (unbalanced) distribution of sound events.

Only those species with at least 100 activations were considered for the detection model, resulting in 20 species. This process reduced the complexity while still allowing to increase the polyphony up to 10. Species with fewer than 100 activations are acoustically present but considered here as background sound.

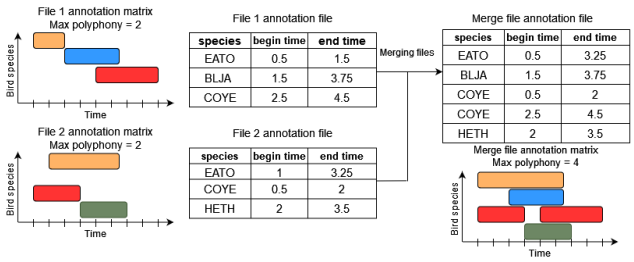


Figure 1: Annotation files merge process. Tables in the figure show how the data is displayed in the files (removing several decimals for visual clarity) and the colored diagram represents the reduced annotation matrix of presence/absence of the species stated in their respective files. It can be observed in those diagrams that the polyphony increases when merging annotations.

3.2. Preprocessing and subsets creation

To feed the recordings to the model, frame blocking was carried out where the audio signal was split into analysis frames of 5-second-fixed length, shifted with a fixed hop size of 2.5 seconds. Additionally, data augmentation methods such as time stretch, pitch shift and time shift were applied to help with generalization. However, the final set would have the same unbalanced proportion of classes mentioned in Section 3.1.

Three subsets were created for polyphony study of 3, 6 and 10 using data synthesis where several 5-second clips were selected and merged. Every created subset contains audio scenes where each clip has the same number of overlapping events (or less) as the determined polyphony. The maximum polyphonies for each subset were chosen by considering previous related work: 3 is the maximum polyphonic scenario successfully studied in an avian SED task; the maximum polyphonic scene reached successfully in other audio domains is 6 and 10 was selected as an approximation of a real birdsong acoustic scenario (which coincides with half the number of classes in the studied dataset).

Annotations were also merged resulting in a new list of all events of the species appearing in the involved files. This process can be observed in the tables found in Figure 1.

As an input feature, a NxM Mel spectrogram was calculated with a window size of 1024, a hop length half that size and 128 Mel bands where N is the number of frames and M the Mel bands. The annotation is transformed into a SxN binary (presence-absence) matrix, as shown in the diagrams of Figure 1, where S denotes the number of classes.

3.3. Model

The model used is a Convolutional Recurrent Neural Network (CRNN) inspired by [9]. Firstly, the convolutional part consists of 5 layers following an evolution of increasing features in the order of: (64, 128, 128, 128, 264). After every convolutional step: layer normalization, leaky *ReLu* activation function and 2D max pooling on the frequency axis. Then, two bidirectional gated recurrent units (GRU) with 128 neurons each, with tanh activation and dropout of 0.5. And, finally, a time-distributed fully connected part of 2 dense layers of 128 neurons each using Leaky *ReLu* as activation function and a follow-up dropout of 0.5. The architecture of the model is depicted in Figure 2 as well as the previous and following steps of the workflow.

Adam was used as the model's optimizer with a fixed learning rate of 0.001 and binary cross-entropy was calculated as loss function.

Every different polyphonic model was trained with 10,000 training samples and 200 epochs, using a training batch size of 128 and train-test split of 90-10. This split was carried out using random items without taking into consideration the unbalanced classes in the dataset.

Three models were created and fitted using subsets of different polyphony as training. From this point, the models are referred to as O3, O6 and O10, whose numeric IDs correspond to the maximum polyphony found in their training subsets.

Table 1: F score and error rate of the three models O3, O6 and O10 using their testing sets and 3 additional subsets with exactly 3, 6 and 10 overlapping events

	Matched testing set, with polyphony 1 to O_		Samples with polyphony of 3		Samples with polyphony of 6		Samples with polyphony of 10	
	F Score	ER	F Score	ER	F Score	ER	F Score	ER
Model O3	0.49±0.17	0.61±0.19	0.45±0.18	0.65±0.21	0.32±0.14	0.77±0.11	0.30±0.13	0.79±0.12
Model O6	**0.51±0.15**	**0.60±0.15**	**0.52±0.18**	**0.57±0.20**	**0.47±0.13**	0.64±0.12	0.41±0.09	0.72±0.08
Model O10	0.47±0.11	0.65±0.18	0.46±0.14	0.88±0.40	0.46±0.12	**0.62±0.11**	**0.45±0.09**	**0.68±0.07**

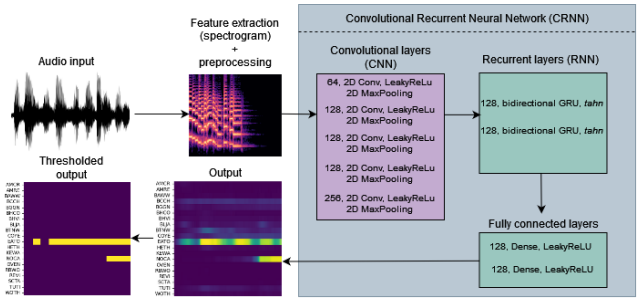


Figure 2: Data workflow of the Convolutional Recurrent model (CRNN). Input transformation and feature selection steps are stated as well as the main architecture design (layers, activations, number of neurons) of the model. Output of the model before thresholding is also shown as part of the workflow process.

Every section of the model and processing of the data was carried out using *Python*, and libraries such as *Keras* and *sklearn*. The code for the preprocessing, model and evaluation is available in github.com/SrArroba/hd_sed.

3.4. Evaluation metrics

An evaluation metric system designed for SED tasks called *sed_eval* [21] is used to measure the performance of the models. F-score and error rate segment-based metrics were used to evaluate the models with a segment size (*t_collar*) of 0.1 seconds.
For every model, 4 subsets are used for testing: 3 subsets with fixed polyphony of 3, 6 and 10, respectively, and a testing set matching each model's training session conditions (i.e., files with up to the specified value of polyphony).

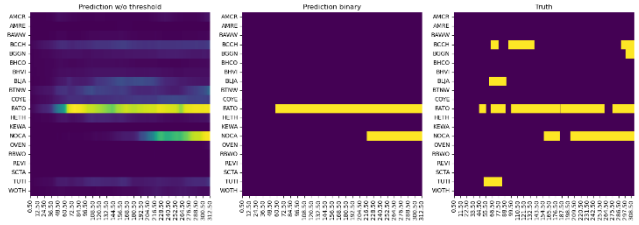


Figure 3: Output after a sample is being run through the model. From left to right: prediction obtained with values between 0 and 1; the same prediction after applying a threshold of 0.5; ground truth.

After performing a prediction, the output is binarized using 0.5 as thresholding value, as seen in Figure 3, where the x-axis represents the time frames (up to 313 frames in 5 seconds) while the y-frame contains the 20 classes/bird species alphabetically sorted. Thresholding can be used to have a better understanding of the models' behavior and pick out further anomalies or faulty tendencies. After this task, the predicted and ground truth can be compared to evaluate the model.

4. RESULTS

While training the models, the loss was measured to study their evolution and how efficiently the models learnt. The first 100 epochs were recorded and shown together in Figure 4.

Analyzing Table 1, it can be observed that O6 performs generally better than the other two models except on higher levels of polyphony than it was trained for, while the O10 model provides consistent results for all types of samples. Despite its complexity, the CRNN model performs adequately matching the conclusions reached by [12] regarding the better fitness of CRNN to deal with sound event detection compared to other networks, such as simple convolutional or recurrent networks.

Additionally, F score and error rate were calculated for each bird species individually to check if there is a correlation between the model's performance on detecting them and their number of activations in the original dataset. As is commonly observed, the items with higher representation in the dataset also gives the strongest performance. Table 2 shows the number of annotations and evaluation metrics for the 3 most common and 3 least common species.

5. DISCUSSION

The goal of this study was to find a model that distinguishes bird species from a large list and temporally detects them when dealing with a dense scene of birdsong. Additionally, it aims to study whether a model can detect the species of a scene with up to 10 overlapping sounds (i.e., very dense scenario).

It could be hypothesized that models that trained with denser samples, learn faster than those who work with simpler scenarios. However, as seen in Figure 4, every model learns in a similar way while the range of their loss evolutions depends on how complex the samples are: the more complex, the higher the range of loss evolution due to the difficulty to learn denser scenes. Generally, models trained with denser scenes are very consistent for all densities while the model trained with simpler samples shows a degradation as the complexity increases. Additionally, models trained with denser scenes perform similar or better in simpler scenes. On the other hand, O10 gets worse results than the O6 model when dealing with emptier scenes. This could be because of the distribution of the training data regarding their polyphony.

Table 2: Individual F score and ER for the 3 most common and 3 least common bird species in the dataset.

Species	Number of annotations	% of annotations	Mean F Score	Mean ER
Eastern towhee (EATO)	5238	36.63 %	0.73±0.23	0.53±0.25
Wood thrush (WOTH)	1826	11.38 %	0.42±0.38	0.51±0.36
Northern cardinal (NOCA)	884	5.51 %	0.24±0.37	0.58±0.67
Hermit thrush (HETH)	162	1.01 %	0	0.12±0.33
Red-bellied woodpecker (RBWO)	145	0.90 %	0.07±0.08	0.07±0.25
Blue-winged warbler (BAWW)	142	0.88 %	0	0.12±0.32

Considering how every synthetic file is generated for each subset, it can denote that the higher the polyphony, the denser the scene and, therefore, more examples per class can be obtained in a single sample. This means that the O3 subset will have fewer representing features of the classes in a single input compared to O6 or O10. However, this does not mean that the models with denser samples learn faster as can be seen in Figure 4, where all models provide a similar evolution in their loss but at different values range. This difference can be explained as the more complex samples are, the more difficult to learn, and therefore, the higher the loss.

On the other hand, as shown in Table 1, models with denser scenes (such as O10) are very consistent for all densities while the model that learns with simpler samples, O3, shows a degradation as the complexity increases, having very poor performance on samples with 10 overlapping sounds. On the other hand, denser scenes mean more complexity, which slows the learning process of the model in that type of scenario. This can be observed in the O10 model where more complex samples are used for training but gets worse results than the O6 model when dealing with emptier scenes. Generally, models trained with denser scenes perform in a similar or better way on simpler samples.

While calculating the performance for each species, it could be hypothesized that it holds a correlation with the total number of activations found in the original dataset. Performing an evaluation using specific species it was demonstrated in Table 2 that species with an elevated number of activations provide better evaluation metrics when studied individually while species with very low appearance in the dataset gives significantly worse performance.

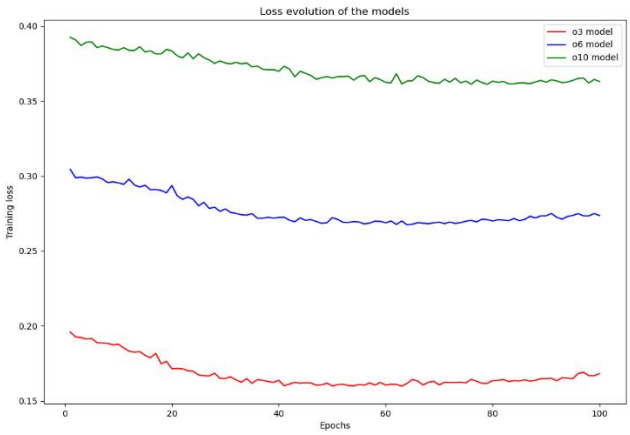


Figure 4: Training loss for the first 100 epochs. The evolution is shown together for the three established models: O3, O6, O10.

Further work could be carried out investigating deeper architectures or more innovative feature selection to achieve better results on denser scenarios. When dealing with higher levels of polyphonies, CNN filters and GRU units should also increase, as suggested by [9].

6. CONCLUSION

In this project, a birdsong dataset of 20 classes was used to train several SED models capable of labeling and temporally allocating the species of birds present on highly dense scenes. Addressing the research questions initially stated, the three models created: O3, O6 and O10 performed suitably considering the complexity of the task.

It could be observed that every model displayed a similar learning curve, nevertheless, using too dense samples could affect the performance of the model when dealing with less crowded scenes but maintaining consistent results in all scenarios. Models' accuracy decayed when dealing with samples more complex than the ones used in training. Additionally, species that have more activations in the original dataset obtained the best results when studying them separately

This work serves to prove the feasibility of elements of modern design, in terms of architecture or features, when facing a task of detecting dense acoustic scenarios. Moreover, it also serves as motivation for other researchers to try new creative ways of dealing with this problem and as a challenge to complete it by adding a location task (SELD).

7. REFERENCES

[1] V. Lostanlen, J. Salamon, A. Farnsworth, S. Kelling & J. Bello, «Robust sound event detection in bioacoustic sensor networks,» *PLoS ONE*, pp. volume 14, issue 10, Pe0214168 2019 DOI: 10.1371/journal.pone.0214168, 2019.

[2] S. Madhusudhana, Y. Shiu, H. Klinck, E. Fleishman, X. Liu, E. Nosal, T. Helble, D. Cholewiak, D. Gillespie, A. Sirovi & et al., «Improve automatic detection of animal call sequences with temporal context,» *Journal of the Royal Society Interface, 18(180):20210297*, 2021.

[3] D. Stowell, M. D. Wood, H. Pamuła, Y. Stylianou & H. Glotin, «Automatic acoustic detection of birds through deep learning: The first bird audio detection

challenge.,» *Methods in Ecology and Evolution, vol. 10, no. 3, pp. 368–380,* 2019.

[4] S. Kahl, T. Wilhelm-Stein, H. Klinch, D. Kowerko & M. Eibl, «Recognizing birds from sound - the 2018 birdclef baseline system,» *CoRR, vol. arXiv: 1804.07177,* 2018.

[5] E. Cakir, G. Parascandolo, T. Heittola, H. Huttunen & T. Virtanen, «Convolutional recurrent neural networks for polyphonic sound event detection,» *IEEE/ACM Transactions on Audio, Speech, and Language Processing, 25(6), 1291-1303,* 2017.

[6] Y. Cao, Q. Kong, T. Iqbal & F. An, «Polyphonic sound event detection and localization using a two-stage strategy,» *arXiv:1905.00268,* 2019.

[7] T. Heittola, «Computational Audio Content Analysis in Everyday Environments,» *Tampere University Dissertations - Tampereen yliopiston väitöskirjat; Vol. 434,* 2021.

[8] J. Abeßer, «USM-SED-A Dataset for Polyphonic Sound Event Detection in Urban Sound Monitoring Scenarios,» *arXiv:2105.02592,* 2021.

[9] S. Adavanne, A. Politis & T. Virtanen, «Multichannel sound event detection using 3D convolutional neural networks for learning inter-channel features,» *international joint conference on neural networks (IJCNN) (pp. 1-7),* 2018.

[10] P. R. Marler & H. Slabbekoorn, «Nature's music: the science of birdsong,» *Elsevier,* 2004.

[11] A. Farina, M. Ceraulo, C. Bobryk, N. Pieretti, E. Quinci & E. Lattanzi, «Spatial and temporal variation of bird dawn chorus and successive acoustic morning activity in a Mediterranean landscape,» *Bioacoustics, 24(3), 269-288,* 2015.

[12] R. Lu & Z. Duan, «Bidirectional GRU for sound event detection,» *DCASE,* 2017.

[13] S. Adavanne, A. Politis & T. Virtanen, «Localization, detection and tracking of multiple moving sound sources with a convolutional recurrent neural network,» *arXiv:1904.12769,* 2019.

[14] E. Cakir & T. Virtanen, «Convolutional recurrent neural networks for rare sound event detection,» *Deep Neural Networks for Sound Event Detection, 12.,* 2019.

[15] M. Crous, «Polyphonic Bird Sound Event Detection With Convolutional Recurrent Neural Networks,» *10.13140/RG.2.2.11943.09126,* 2019.

[16] M. Faiß, T. Riede & F. Goller, «Tonality over a broad frequency range is linked to vocal learning in birds,» *The Royal Society,* 2022.

[17] K. Drossos, S. I. Mimilakis, S. Gharib, Y. Li & T. Virtanen, «Sound event detection with depthwise separable and dilated convolutions,» *International Joint Conference on Neural Networks (IJCNN),* 2020.

[18] T. Pellegrini, «Densely connected CNNs for bird audio detection,» *25th European Signal Processing Conference (EUSIPCO),* 2017.

[19] D. Stowell & D. Clayton, «Acoustic event detection for multiple overlapping similar sources,» *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA),* 2015.

[20] L. M. Chronister, T. A. Rhinehart, A. Place & J. Kitzes, «An annotated set of audio recordings of Eastern North American birds containing frequency, time, and species information,» 2021.

[21] A. Mesaros, T. Heittola & T. Virtanen, «Metrics for polyphonic sound event detection,» *Applied Sciences, vol. 6, no. 6, pp. 162–178,* 2016.

LANGUAGE-BASED AUDIO RETRIEVAL WITH TEXTUAL EMBEDDINGS OF TAG NAMES

Thomas Pellegrini

IRIT, Université de Toulouse, CNRS, Toulouse INP, UT3, Toulouse, France
Artificial and Natural Intelligence Toulouse Institute (ANITI)
thomas.pellegrini@irit.fr

ABSTRACT

Language-based audio retrieval aims to retrieve audio recordings based on a queried caption, formulated as a free-form sentence written in natural language. To perform this task, a system is expected to project both modalities (text and audio) onto the same subspace, where they can be compared in terms of a distance. In this work, we propose a first system based on large scale pretrained models to extract audio and text embeddings. As audio embeddings, we use logits predicted over the set of 527 AudioSet tag categories, instead of the most commonly used 2-d feature maps extracted from earlier layers in a deep neural network. We improved this system by adding information from audio tag text embeddings. Experiments were conducted on Clotho v2. A 0.234 mean average precision at top 10 (mAP@10) was obtained on the development-testing split when using the tags, compared to 0.229 without. We also present experiments to justify our architectural design choices[1].

Index Terms— Language-based audio retrieval, Pre-trained representations, PaSST, MPnet, Transformers, AudioSet tags

1. INTRODUCTION

The ability to retrieve audio recordings from a text query can be very useful when searching for recordings in a large audio database. Language-based audio retrieval systems rank audio samples according to their match with a queried caption.

Recent systems are usually comprised of two neural networks: a text encoder, responsible for encoding a concise and dense representation of a textual query, and an audio encoder that encodes the audio content of the candidate recordings [1, 2]. Both encoders are expected to provide representations embedded in the same subspace, where the textual and audio samples can be compared in terms of a distance.

In this paper, we present a baseline system that follows this architecture, based on two open-sourced pre-trained models to extract text and audio embeddings. Our main innovation is two-fold: i) we use logits as basic audio embeddings [3], instead of the more usual 2-d feature maps extracted from a hidden layer of a pretrained deep neural network [4, 5, 6], ii) we improve our baseline system by incorporating information from tags predicted on the audio recordings. We propose to combine the basic audio logit embeddings with the textual embeddings of the tag names.

Thanks for funding by the French ANR agency (LUDAU, ANR-18-CE23-0005-01) and "Investing for the Future — PIA3" AI Interdisciplinary Institute ANITI (ANR-19-PI3A-0004).

[1]PyTorch code available: `https://github.com/topel/my-audio-retrieval-dcase2022`

After describing our system architectures and experimental setup, we report results obtained on Clotho v2 [7]. We then discuss our architecture design choices in depth.

2. SYSTEM DESCRIPTION

Our system architecture is shown in Fig. 1, with the audio encoder on the right part of the figure, and the caption encoder on the left part. We used two open-sourced pretrained transformers as encoders. Both were "frozen": the embeddings were extracted only once offline and used as input to our systems. The learnable part of the model corresponds to a single linear layer that projects the audio embeddings onto the caption embedding subspace.

2.1. Text encoder: sentence embeddings with MPnet

The textual queries (captions) are encoded as sentence embeddings, obtained by averaging the word embeddings outputted by a sentence transformer. The all-mpnet-base-v2 model [8], further referred to as MPnet, was used to extract these embeddings $\vec{e}_s$, which are 768-dimensional ℓ_2-normalized vectors. MPnet is a transformer of more than 109 M parameters, trained on over a billion pairs of sentences. It was chosen since it was reported as the best model for sentence embedding against a selected list of other models [9]. We apply Layer normalization [10], followed by ℓ_2-normalization is applied to the embeddings to obtain the final ones, named $\vec{e}_q$:

$$\vec{e}_q = \ell_2\text{-norm}\left(\text{LayerNorm}\left(\vec{e}_s\right)\right) \quad (1)$$

Layer normalization (*LayerNorm*, LN) is used to normalize the samples of a minibatch independently, with adaptive per-dimension bias and gain (scale). We found that using those led to slight overfitting, thus, we removed this option. Therefore, our text encoder does not have any learnable parameter.

2.2. Audio encoder: AudioSet logit embeddings with PaSST

In the proposed system, an audio recording is encoded as a single vector, a so-called scene embedding, using the Patchout transformer named PaSST [11], pretrained on AudioSet [12]. We use the "logits" embedding outputted by PaSST, which is a 527-dimensional dense vector, comprised of the logits predicted for the 527 AudioSet event tag classes. We denote this embedding $\vec{e}_l$.

Layer-norm is then used, in this case with adaptive bias and gain, followed by a linear layer and a final ℓ_2-normalization. The final embedding is denoted $\vec{e}_a$. We can summarize these steps in Eq. 2:

$$\vec{e}_a = \ell_2\text{-norm}\left(\text{Linear}\left(\text{LayerNorm}\left(\vec{e}_l\right)\right)\right) \quad (2)$$

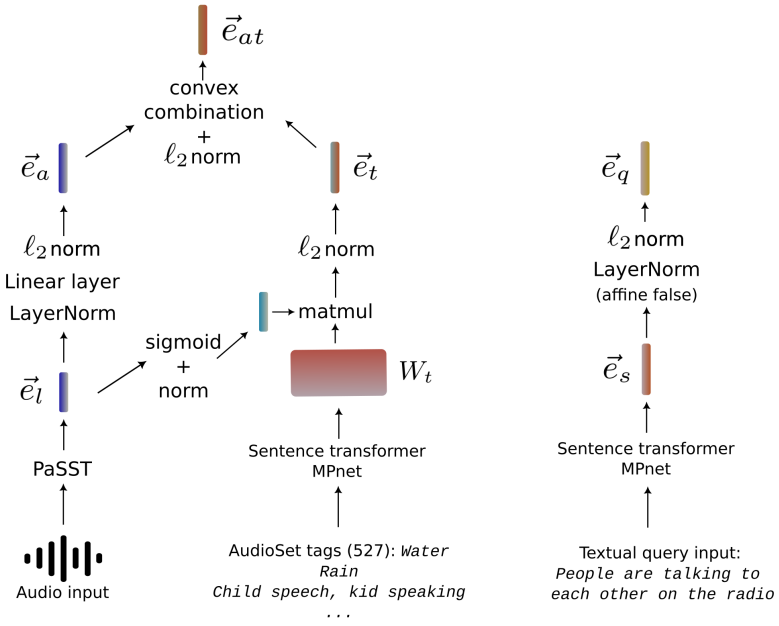


Figure 1: System architecture. Left: audio encoder (with and without tag embeddings), right: text encoder. $\vec{e}_l$: logit embedding, $\vec{e}_a$: audio embedding, $\vec{e}_t$: weighted AudioSet tag embedding, $\vec{e}_{at}$: combined audio and tag embedding, $\vec{e}_s$: sentence embedding, $\vec{e}_q$: textual query embedding, W_t: matrix of the AudioSet tag name embeddings.

We will, here-after, refer to our baseline system using PaSST and MPnet as PaSST-MPnet.

2.3. Augmented audio encoder: tag embeddings with MPnet

Adding information from tags and audio event predictions was shown to help audio captioning systems [13, 14, 15]. It should also help in our retrieval task. In this work, we tried to add such information within the audio encoder, by using MPnet to encode tag names as text embeddings. This tag name encoder is shown in Fig. 1, on the left of the audio encoding branch.

AudioSet includes 527 audio event classes, with names that can be single words, *e.g.*, "Water", "Rain" or a sequence of words, such as "Child speech, kid speaking". We precomputed 527 tag name embeddings, extracted with MPnet.

For a given audio file, we use the PaSST logits $\vec{e}_l$ to weight each of the 527 embeddings, sum them to obtain a single tag embedding $\vec{e}_t$, according the following equations. First, the PaSST logit embedding is passed through a sigmoid function to obtain multilabel probabilities $\vec{p}$, then we compute the weighted sum of the precomputed tag embeddings:

$$\vec{p} = \text{sigmoid}\,(\vec{e}_l) \tag{3}$$

$$\vec{e}_t = \ell_2\text{-norm}\left(\frac{\vec{p}}{\sum_c p_c} W_t\right) \tag{4}$$

where $W_t \in \mathbb{R}^{527\times d}$ is the matrix containing the 527 embeddings of the AudioSet tag names. d is the dimension of the embeddings provided by MPnet ($d = 768$). c is an index over the 527 tag categories.

After an ℓ_2-normalization, $\vec{e}_t$ is combined to $\vec{e}_a$ with a convex linear summation:

$$\vec{e}_{at} = \lambda\,\vec{e}_a + (1-\lambda)\,\vec{e}_t \tag{5}$$

where $\lambda \in [0, 1]$ is a weight tuned on a validation subset. The final audio embedding $\vec{e}_{at}$ is ℓ_2-normalized before being compared to the caption query embedding:

$$\vec{e}_{at} = \ell_2\text{-norm}\,(\vec{e}_{at}) \tag{6}$$

The learnable parameters of the proposed system correspond to the audio encoder ones: in the linear layer and in the element-wise adaptive biases and gains of LN, with a total of 407 k parameters. If we take into account the large-scale transformers, this number increases up to 196M parameters, but as previously mentioned, we did not fine-tune the transformers.

2.4. Training objective

We used the same loss and similarity scoring functions than in the challenge baseline system [2]. Similarity between audio and caption embeddings is estimated by taking the dot product between their embeddings, and all our systems were trained with a sampling-based triplet loss [16]. We would like a model to provide a similarity score S^p high for positive audio-query pairs, and low for either an audio embedding not paired with the caption query (audio impostor score S^n_a), or for a caption not corresponding to the audio embedding (query impostor score S^n_q). The audio and query impostors are randomly selected from the minibatch of samples being processed. The loss function as a function of the network learnable parameters θ is the sum of the contribution of these three scores:

System	# params	Development-testing split mAP@10	R@1	R@5	R@10	Evaluation dataset mAP@10	R@1	R@5	R@10
Challenge baseline	732k	0.068	0.032	0.109	0.188	0.061	0.026	0.102	0.176
CNN-transformer [5]	195M	0.260	0.150	0.400	0.530	0.251	0.153	0.387	0.504
PaSST-MPnet	196M	0.229	0.134	0.355	0.482	0.212	0.124	0.319	0.448
PaSST-MPnet-tags	196M	0.234	0.138	0.364	0.485	0.214	**0.128**	**0.332**	0.445
PaSST-MPnet-tags-AC	196M	0.240	0.145	0.365	**0.500**	0.213	0.125	0.322	0.454
Ensemble (2 models)	196M	**0.243**	**0.148**	**0.369**	0.498	**0.216**	0.127	0.321	**0.463**

Table 1: Results on Clotho v2.

$$\mathcal{L}(\theta) = \frac{1}{N} \sum_{\text{batch}} \max(0, S_a^n - S^p + \eta) + \max(0, S_q^n - S^p + \eta) \quad (7)$$

where N is the number of samples in the minibatch, η a margin.

2.5. Evaluation

Once a system is trained, retrieval is performed by scoring the similarity between the caption embedding and the audio embeddings of all the candidate audio files. The same scoring function as during training is used: the dot product. Following the DCASE 2022 challenge requirements, our systems retrieve ten audio files for each queried caption, sorted according to their match with the query.

Standard information retrieval metrics are used to evaluate the audio retrieval models: mean average precision at top-10 (mAP@10) [17] and recall at k, the number of retrieved audio items considered for scoring (R@k with $k \in \{1, 5, 10\}$) [18]. To estimate these metrics, only a correct audio-caption pair is considered as positive, all the other audio recordings retrieved are considered negative. To rank the systems submitted to the DCASE 2022 challenge, mAP@10 was the main metric [2]. It is a rank-aware metric. For a given queried caption, the precision values are averaged at the positions where the relevant items are in the retrieved rank list. The R@k metric is rank-unaware, and corresponds to the proportion of relevant items among the top-k retrieved results. For both mAP@10 and R@k, the higher the value, the better the system.

3. EXPERIMENTAL RESULTS

3.1. Datasets

We conducted our experiments on Clotho v2 [7], with the splits defined in the DCASE 2022 challenge [19]. The development set is comprised of 3839 audio clips with 19195 reference captions for training, and 1045 audio recordings with 5225 captions for testing. We refer to this test subset as development-testing, dev-test in short. We did not use the validation set. A Clotho evaluation set was provided in the challenge, used to rank the submitted systems. It contains 1000 audio files with 1000 associated captions [2]. We also used for pretraining the training subset of AudioCaps [20], containing 46231 audio files with one reference caption per file.

3.2. Experimental setup

All our models were trained on Clotho v2 for 50 epochs with the same setup: minibatches of 128 samples, a 1e-3 initial learning rate, and the Adam optimizer. We used a reduce-on-plateau scheduler based on the Clotho validation split loss, with a 0.5 ratio and a 5-epoch patience. Smaller and larger batch sizes were tested, but 128 was found to be the best one. We used $\eta = 0.4$, and $\lambda = 0.8$ when using the tags, as will be discussed here-after. When pretraining a model on AudioCaps, the number of epochs was 100 and no learning rate scheduler was used.

3.3. Results

Table 1 shows the results of our systems, together with the ones of the baseline proposed by the DCASE 2022 Task 6b challenge organizers (based on a convolutional recurrent neural network and pretrained word embeddings), and of a CNN-transformer [5], which reached the second rank out of ten teams in the challenge, with models comprised of a number of parameters similar to ours.

First, our systems based on pretrained transformers for both audio and text encoding largely outperformed the challenge baseline, which was trained from scratch on the Clotho v2 development dataset. Relying on pretrained models is efficient in this case, where labeled training data is scarce. Second, our results are worse by about 0.02 on the evaluation dataset (eval) than on the development-testing subset (dev-test). This is probably due to the fact that model selection was performed on this subset, so overfitting might have happened. It can also be that eval is more difficult than dev-test.

Compared to our baseline model PaSST-MPnet, adding information from the tag embeddings was beneficial: PaSST-MPnet-tags reached 0.234 mAP@10 compared to 0.229, on dev-test, and 0.214 compared to 0.212 on eval.

Pretraining on AudioCaps (PaSST-MPnet-tags-AC) brought a 0.06 absolute mAP@10 improvement on dev-test, compared to PaSST-MPnet-tags. Interestingly, this gain was not observed on eval, so we cannot conclude that pretraining was beneficial or not. PaSST-MPnet-tags models trained on AudioCaps, before finetuning on Clotho, reached about 0.18 mAP@10 on Clotho dev-test.

Finally, an ensemble of two PaSST-MPnet-tags-AC models trained with different seeds, led to our best results. By ensemble, we mean that the distance scores outputted by the two models were averaged. This setting allowed us to obtain the fourth place of the challenge [19].

4. ARCHITECTURE DESIGN CHOICES

In the following, all reported experiments were conducted with PaSST-MPnet-tags models.

4.1. Influence of *LayerNorm* in the audio encoder

As described in Section 2, in our models, we use *LayerNorm* (LN) with adaptive gain and bias applied to the PaSST embeddings in the

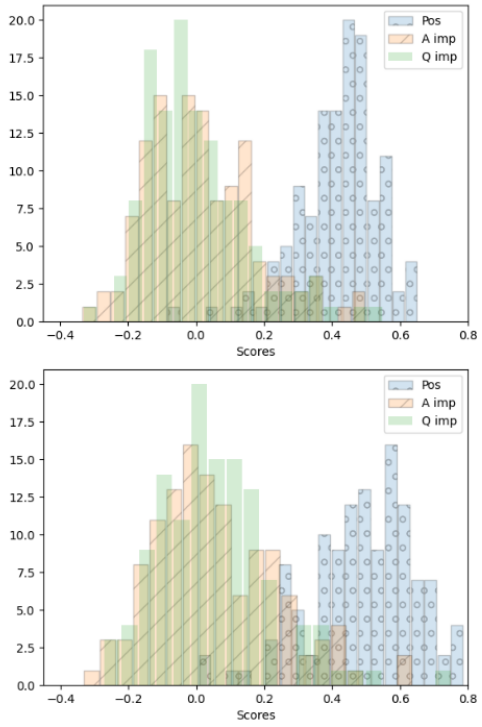


Figure 2: Similarity score distributions on the training subset when using either a 0.4 (top) or a 1.0 (bottom) value for the margin η, when training a PaSST-MPnet-tags model. Pos: positive audio-caption pair, A imp: negative audio-query pair (audio impostor), Q imp: negative query-audio pair (caption impostor).

audio encoder.

First, if we do not use LN in the audio encoder, mAP@10 and the other metrics are much worse: 0.217 mAP@10, compared to 0.234 reported in Table 1.

Second, we tried to use batch normalization (*BatchNorm*, BN) in the audio encoder, instead of LN. Using BN and keeping all the rest of the architecture unchanged (and for a given random seed), led to a 0.229 mAP@10 value, again worse than 0.234 when using LN. By observing the learning curves, BN seems to overfit a little more than LN. When using adaptive bias and scale, the number of learnable parameters is the same for both types of normalization, *i.e.* twice the dimension of the embeddings. The only possible reason why BN overfits more is because of the running means and standard deviations estimated on the training subset of data, that may not generalize so well on the test subset. These are necessary for BN but not for LN, since LN performs a per sample normalization.

4.2. Impact of the margin η

In [16], the authors used a default value of 1.0 for the margin η, used in the loss function, given in Eq. 7. In the DCASE challenge baseline, this default value was also used. In our case, the audio and queried caption embeddings are of unit norm. The results of the dot products, *i.e.,* the similarity scores, are theoretically between -1.0 and 1.0. A margin of 1.0 might be to difficult to obtain when training a model. Indeed, $\eta = 1.0$ led to a 0.225 mAP@10 on dev-test, worse than 0.234 obtained with $\eta = 0.4$. We tried several values lower than 1.0, and $\eta = 0.4$ was the best value.

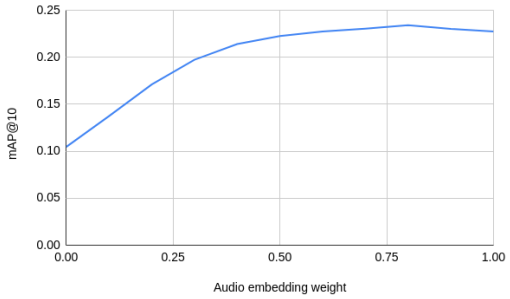


Figure 3: mAP@10 values on dev-test according to the audio embedding weight λ.

In Fig. 2, we plotted the score distributions of the positive pairs (*Pos*, a positive caption-audio pair) and negative pairs (*A imp* and *Q imp* for audio and query impostors, resp.), on the training subset. The distributions on the dev-test subset are very similar. The difference between the top and the bottom plots is the value of the margin: $\eta = 0.4$ (top figure) and $\eta = 1.0$ (bottom figure). As can be seen, the support interval of the scores is smaller with $\eta = 0.4$ than with $\eta = 1.0$: the positive pair scores reach about 0.62 maximum with $\eta = 0.4$, and about 0.8 with $\eta = 1.0$. Nevertheless, we can see more overlap with impostor scores in the case of $\eta = 1.0$, which mean that more mistakes are made by this model.

4.3. Influence of the audio/tag weight λ

Figure 3 shows the mAP@10 values on dev-test, when varying λ from Eq. 5. When $\lambda = 0$, the audio encoder provides as output the tag embeddings $\vec{e}_t$, and the model has no learnable parameters at all. In this setting, a 0.105 mAP@10 is obtained. This shows that using the tag name embeddings weighted with their probabilities already brings some information. When $\lambda = 1$, the audio encoder provides as output the projected PaSST embeddings $\vec{e}_a$, and the model corresponds to PaSST-MPnet, which reached 0.229 mAP@10. In between, the score varies, and the best value was obtained with $\lambda = 0.8$.

5. CONCLUSION

We reported text-based audio retrieval experiments, with systems based on pretrained audio and sentence large scale transformers. In our baseline system, we used logits as audio embeddings, instead of the most commonly used 2-d feature maps extracted from earlier layers in a deep neural network. We improved this system by adding information from AudioSet tags, encoded as sentence embeddings. Finally, we discussed some of our architecture design choices: the use of layer normalization, a value smaller than 1.0 for the margin in the contrastive loss function for learning, and the 0.8 value for weighting the audio embeddings when adding the tag contribution within the audio encoder.

As a short-term perspective, we would like to use logit embeddings together with more standard 2-d feature maps in the audio encoder, to see whether those are complementary [5]. We also would like to investigate the use of external data, either for pretraining models or in a semi-supervised learning setting, where existing algorithms could be adapted to the present task [21].

6. REFERENCES

[1] H. Xie, O. Räsänen, K. Drossos, and T. Virtanen, "Unsupervised audio-caption aligning learns correspondences between individual sound events and textual phrases," in *Proc. ICASSP*. IEEE, 2022, pp. 8867–8871.

[2] H. Xie, S. Lipping, and T. Virtanen, "Dcase 2022 challenge task 6b: Language-based audio retrieval," *arXiv preprint arXiv:2206.06108*, 2022.

[3] E. Tzinis, S. Wisdom, J. R. Hershey, A. Jansen, and D. P. Ellis, "Improving universal sound separation using sound classification," in *Proc. ICASSP*. IEEE, 2020, pp. 96–100.

[4] X. Xu, Z. Xie, M. Wu, and K. Yu, "The SJTU System for DCASE2022 Challenge Task 6: Audio Captioning with Audio-Text Retrieval Pre-training," DCASE2022 Challenge, Tech. Rep., July 2022.

[5] X. Mei, X. Liu, H. Liu, J. Sun, M. D. Plumbley, and W. Wang, "Language-based audio retrieval with pre-trained models," DCASE2022 Challenge, Tech. Rep., July 2022.

[6] T. L. de Gail and D. Kicinski, "Take it easy: Relaxing contrastive ranking loss with CIDEr," DCASE2022 Challenge, Tech. Rep., July 2022.

[7] K. Drossos, S. Lipping, and T. Virtanen, "Clotho: An audio captioning dataset," in *Proc. ICASSP*. IEEE, 2020, pp. 736–740.

[8] https://huggingface.co/sentence-transformers/all-mpnet-base-v2.

[9] https://www.sbert.net/docs/pretrained_models.html.

[10] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.

[11] K. Koutini, J. Schlüter, H. Eghbal-zadeh, and G. Widmer, "Efficient training of audio transformers with patchout," *arXiv preprint arXiv:2110.05069*, 2021.

[12] J. F. Gemmeke, D. P. W. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio set: An ontology and human-labeled dataset for audio events," in *Proc. IEEE ICASSP*, New Orleans, LA, 2017.

[13] Z. Ye, H. Wang, D. Yang, and Y. Zou, "Improving the performance of automated audio captioning via integrating the acoustic and textual information," DCASE2021 Challenge, Tech. Rep., July 2021.

[14] F. Gontier, R. Serizel, and C. Cerisara, "Automated audio captioning by fine-tuning bart with audioset tags," in *Proc. Workshop DCASE*, 2021, pp. 170–174.

[15] Q. Han, W. Yuan, D. Liu, X. Li, and Z. Yang, "Automated audio captioning with weakly supervised pre-training and word selection methods," in *Proc. Workshop DCASE*, Barcelona, Spain, November 2021, pp. 6–10.

[16] D. Harwath, A. Torralba, and J. Glass, "Unsupervised learning of spoken language with visual context," in *Advances in Neural Information Processing Systems*, D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, Eds., vol. 29. Curran Associates, Inc., 2016. [Online]. Available: https://Proc..neurips.cc/paper/2016/file/82b8a3434904411a9fdc43ca87cee70c-Paper.pdf

[17] P. Kaur, H. S. Pannu, and A. K. Malhi, "Comparative analysis on cross-modal information retrieval: A review," *Computer Science Review*, vol. 39, p. 100336, 2021.

[18] A. S. Koepke, A.-M. Oncescu, J. Henriques, Z. Akata, and S. Albanie, "Audio retrieval with natural language queries: A benchmark study," *IEEE Trans. on Multimedia*, pp. 1–11, 2022.

[19] http://dcase.community/challenge2022/.

[20] C. D. Kim, B. Kim, H. Lee, and G. Kim, "Audiocaps: Generating captions for audios in the wild," in *Proc. of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 2019, pp. 119–132.

[21] L. Cances, E. Labbé, and T. Pellegrini, "Comparison of semi-supervised deep learning algorithms for audio classification," *EURASIP Journal on Audio, Speech, and Music Processing*, vol. 2022, no. 23, Sept. 2022. [Online]. Available: https://doi.org/10.1186/s13636-022-00255-6

LATENT AND ADVERSARIAL DATA AUGMENTATIONS FOR SOUND EVENT DETECTION AND CLASSIFICATION

David Perera, Slim Essid, Gaël Richard

LTCI, Télécom Paris, Institut Polytechnique de Paris, France
{david.perera, slim.essid, gael.richard}@telecom-paris.fr

ABSTRACT

Invariance-based learning is a promising approach in deep learning. Among other benefits, it can mitigate the lack of diversity of available datasets and increase the interpretability of trained models. To this end, practitioners often use a consistency cost penalizing the sensitivity of a model to a set of carefully selected data augmentations. However, there is no consensus about how these augmentations should be selected. In this paper, we study the behavior of several augmentation strategies. We consider the task of sound event detection and classification for our experiments. In particular, we show that transformations operating on the internal layers of a deep neural network are beneficial for this task.

Index Terms— sound event detection, data augmentation, adversarial learning

1. INTRODUCTION

Ambient sound analysis is a rapidly growing field, which has several important industrial applications such as security (e.g., audio surveillance), health (e.g., home monitoring, diagnosis based on sound), transportation (e.g., autonomous driving), manufacturing industry (e.g., predictive maintenance) and bioacoustics (e.g., ecosystem evolution tracking). In this context, sound event detection and classification consists in identifying and temporally localizing sound events in a complex acoustic scene. Deep learning has been successfully applied to this task, significantly improving the state of the art. However, this approach has two major drawbacks [1]. On the one hand, it requires a large amount of annotated data, which is costly and time-consuming to gather. On the other hand, the models obtained by this approach lack interpretability, making it hard to assess their reliability in unseen situations.

Data augmentation is an efficient technique which can mitigate the lack of available data and exploit unlabeled data [2]. This method also makes it possible to force trained models to learn specific invariants. By selecting relevant data augmentations, one can increase the model robustness to annotation errors and make them easier to interpret [2]. Moreover, data augmentation can be easily applied to any training algorithm and can improve its performance at a minimal development cost. This explains the popularity of this approach, especially in the field of sound event detection.

Many data augmentation strategies have been designed over the past decades. They now form a large set of methods that is costly to explore exhaustively. As a consequence, default settings are often passed from one system to another with only limited investigation [3], which leads to sub-optimal design choices.

This paper proposes a comparative study of the impact of several data augmentation strategies on task 4 of the DCASE challenge.[1] We focus on domain agnostic data augmentations. In this context, we study the impact of adversarial augmentations and latent augmentations (data augmentations applied directly on the latent space of a deep neural network). We propose a common training framework to compare these augmentation strategies. By carefully selecting them, we show that it is possible to outperform the baseline model we compete against and to simplify its training objective.

This article is organized as follows. Section 2 presents a brief state of the art. Section 3 introduces task 4 of the DCASE challenge and describes its baseline. Section 4 explains the experimental framework that we use. Finally, section 5 discusses the obtained results.

2. STATE OF THE ART

There are two main strategies used to overcome the lack of training data: exploiting additional data from a related domain, and constraining the algorithm using domain knowledge [2]. Combining both ideas, invariance-based approaches penalize the variations of a model f in the neighborhood of the training data. More precisely, these approaches aim to reduce the quantity $\|f(x) - f(\tau(x))\|$, where x denotes a training point, $\|\cdot\|$ denotes a norm and τ denotes a data augmentation. This forces the model f to learn some invariants, which can be defined explicitly. We thus obtain some guarantees on the behavior of f, and consequently improve its interpretability. We briefly review here the main developments in this field.

The Ladder Network [4] adds to the objective function a cost enforcing the invariance of the model to small perturbations of its input and internal representations. A key point is that this method can leverage unlabeled data. In [5], the authors use Dropout augmentation [6] instead of random noise.

Temporal Ensembling (TE) [7] compares the prediction $f(\tau(x))$ computed for an augmented input $\tau(x)$ with an average of the predictions $f^{(n)}(\tau(x))$ computed at different epochs n. This method is inspired by the state of the art in stochastic optimization [8]. The idea is to accelerate the convergence of f by mitigating the randomness resulting from the selection of the mini-batches and the augmentation of the input.

Mean Teacher (MT) [9] is an improvement of TE: instead of averaging the predictions at each training epoch, it maintains an average of the model parameters. Moreover, it updates this average after each mini-batch instead of updating it after each epoch, which speeds up training.

In [10], the authors use Mixup augmentation [11] within the MT framework. This new training method encourages convex in-

[1] https://dcase.community/

terpolation between two input samples. It is based on the following hypothesis: the decision boundaries of a classifier should be located between samples of different classes. Consequently, instead of moving an input sample in a random direction, which is inefficient, it is better to move it in the direction of a sample from a different class.

Following a similar line of research, Virtual Adversarial Training (VAT) promotes invariance to small perturbations of the input. This method uses adversarial perturbations (which maximize the variation of f) instead of using random perturbations. According to the authors, adversarial perturbations constitute a better heuristic, and can speed up the convergence of the algorithm. This algorithm is however based on a second order approximation of the adversarial perturbations, which is slow to compute. The Adversarial Noise Layer [12] generalizes the idea of adversarial perturbation to the intermediate layers of a model. It uses a first order approximation, which is faster to compute. In [13], the authors study a variant of this method based on the Dropout augmentation.

Universal Adversarial Perturbation (UAP) [14] is a further generalization of adversarial attacks. While the approaches presented above focused on fooling one model on a single instance of the training set, UAP seek to fool several models on most instances of the training set. The authors have shown that the resulting perturbations exploit the geometry of the classifiers' decision boundaries. This makes them useful to design an invariance metric. However, their high computational cost discourages any use, aside from evaluation.

Finally, we can mention Mixmatch [15]. In addition to a consistency cost enforcing invariance to data augmentations, this method takes into account an entropy cost. The purpose of this cost is to increase the confidence level of the predictions made by the trained model.

Some of the methods presented above have already been applied to DCASE Task 4. Since 2019, the MT method has been used as the baseline for this task, and many algorithms submitted to the challenge are also based on this method. The authors of [3] propose an in-depth study of the application of MT to this task. The VAT method has also been applied to this task, with three different architectures: A Recurrent Convolutional Network (CRNN) [16], a Gated Recurrent Neural Network [17], and a Gated Recurrent Convolutional Neural Network [18]. More recently, the authors of [19] have studied a variant in which the noise is not adversarial but random. However, both the training method and the architecture change from one of these articles to another. It is therefore difficult to factor out the impact of the data augmentation strategy alone.

Closest to our work, [20] and [21] study the impact of several types of data augmentations on a single model. The authors focus on audio-specific data augmentations: pitch shifting, time shifting, reverberation, frequency masking and time masking. Unlike these studies, however, we focus our work on adversarial and latent data augmentation. There are two main reasons for this. First, these augmentations are domain agnostic, and could potentially work across a vast range of tasks. This makes them very potent and interesting to study. Second, these augmentations can be restricted to the audio domain, giving birth to adversarial audio augmentations (which could be seen as realistic worst-case scenarios), and latent audio augmentations (which would force the internal representations of a deep neural network to keep the structure of audio data, such as time-shift or pitch-shift invariance). The aim of this paper is to provide insights about the impact of adversarial and latent augmentations on sound event detection. We use task 4 of DCASE as a case study.

3. TASK DEFINITION

Task 4 of DCASE uses the dataset Domestic Environment Sound Event Detection (DESED) [20], which is composed of 10-second audio recordings made in domestic environments. DESED is divided into three distinct datasets: an unlabeled dataset $\mathcal{D}_u$, a weakly labelled dataset $\mathcal{D}_w$, and a synthetic strongly labelled dataset $\mathcal{D}_s$. There are ten possible classes for the annotation of sound recordings. Each recording in $\mathcal{D}_w$ is annotated with the set of sound events it contains. For $\mathcal{D}_s$, each event is temporally localized in addition to being identified by a label.

Two metrics are used to evaluate the models. To measure the temporal accuracy of the predictions, we use an event-based F1 macro score, which is denoted by $macro$ in the following. In order to measure the labeling accuracy, we use the Polyphonic Sound Detection Score [22], which is denoted by $psds$ in the following. For the computation of these two metrics, we adopt the parameters proposed for the 2020 edition of the DCASE challenge. We use the $macro$ score, computed on the public evaluation dataset[2] proposed in this same edition, to evaluate and compare the models.

In order to increase the usability of our results, we build our study on [20], which already investigates several design choices concerning task 4 of DCASE. In particular, we use the same baseline model[3] as a basis for our experiments. This baseline exploits a CRNN architecture, which is commonly used in audio. It achieves high performance on the task. Moreover, the impact of its various components has been exhaustively studied. The baseline takes as input Mel spectrograms with 128 Mel bands, built with an analysis window of size 2048 and a hop length of 255. The input signals are sampled at 16kHz. The convolutional block of the CRNN is composed of 7 layers with filter sizes (16, 32, 64, 128, 128, 128, 128) and a kernel of size 3x3. Each convolution is followed by a Batch Normalization layer, a Gated Linear Unit, a Dropout layer with probability 0.5, and a Maxpooling layer. The recurrent block is composed of two Gated Recurrent Units with 128 layers each. It is followed by the attention pooling layer described in [23]. Median filtering is applied during post-processing. This architecture features 11 million trainable parameters. The model is trained on 200 epochs with the Adam optimizer.

4. AUGMENTATION STRATEGIES

4.1. Training framework

We compare several training objectives. In each case, the training objective can be decomposed into three cost functions: a classification metric penalizing prediction errors $\mathcal{L}_{class}$, a consistency metric promoting invariance to data augmentations $\mathcal{L}_{const}$ and a regularization metric $\mathcal{L}_{reg}$.

We segment the training mini-batches into three parts, each corresponding to one of the datasets $\mathcal{D}_u$, $\mathcal{D}_w$ and $\mathcal{D}_s$. We use the respective proportions $(1/2, 1/4, 1/4)$.

We note f the baseline and f_{ema} its exponential moving average across iterations. If we note $f^{(n)}$ the model obtained after iteration n, $\theta^{(n)}$ its parameters (with similar notations for f_{ema} and

[2] https://zenodo.org/record/3588172

[3] https://github.com/turpaultn/dcase20_task4/

θ_{ema}) and $\alpha^{(n)}$ a coefficient varying during the training, then f_{ema} is defined by its parameters update

$$\theta_{ema}^{(n+1)} = \alpha^{(n)}\theta^{(n)} + (1-\alpha^{(n)})\theta_{ema}^{(n)}. \quad (1)$$

Noting x a recording in the DESED dataset, y the corresponding label when it exists, d a Gaussian noise vector, $\mathcal{L}_{BCE}$ the binary cross-entropy and $\mathcal{L}_{MSE}$ the mean square error, we can define the training objective of the baseline as follows:

$$\mathcal{L}_{class} = \begin{cases} \mathcal{L}_{BCE}[f(x),\ y] & \text{if } (x,y) \in \mathcal{D}_w \cup \mathcal{D}_s \\ 0 & \text{else} \end{cases}, \quad (2)$$

$$\mathcal{L}_{const} = \mathcal{L}_{MSE}[f(x),\ f_{ema}(x+d)], \quad (3)$$

$$\mathcal{L}_{reg} = 0. \quad (4)$$

Through our experiments, we keep the classification cost used by the baseline. However, we will experiment with other consistency and regularization costs.

4.2. Consistency costs

The simplest consistency cost we considered is the $\mathcal{L}_2$ distance between a prediction $f(x)$ and a perturbed prediction $f(x+d)$ (with d a Gaussian noise vector):

$$\mathcal{L}_{const} = \mathcal{L}_{MSE}[f(x),\ f(x+d)]. \quad (5)$$

We have also considered an adversarial consistency cost,

$$d = \nabla_d\, \mathcal{L}_{MSE}[f(x),\ f(x+d)]\,|_{d=0}, \quad (6)$$

$$\mathcal{L}_{const} = \mathcal{L}_{MSE}[f(x),\ f(x+d)], \quad (7)$$

as well as the following variant of VAT:

$$d = argmax_{\|d\|\leq\epsilon}\, \mathcal{L}_{MSE}[f(x),\ f(x+d)], \quad (8)$$

$$\mathcal{L}_{const} = \mathcal{L}_{MSE}[f(x),\ f(x+d)]. \quad (9)$$

The original VAT algorithm uses the KL-divergence as a metric to compare the two outputs of the classifier $f(x)$ and $f(x+d)$. However, the $\mathcal{L}_2$ distance proved to be empirically better for this task. This choice also has the added benefit to homogenize the definition of the consistency costs $\mathcal{L}_{const}$ that we study. Indeed, these costs differ now only by the definition of the perturbation d.

Finally, we use a Mixup consistency cost, computed from a second sample x'

$$\mathcal{L}_{const} = \mathcal{L}_{MSE}[Mixup(f(x), f(x')),\ f(Mixup(x,x'))]. \quad (10)$$

All these augmentations can be applied to the internal representations of the model. We apply data augmentation to the output of the CNN block and to the output of the RNN block of the baseline (see Figure 1).

4.3. Regularization costs

The baseline does not use an explicit regularization cost. This is because MT combines a consistency criterion and regularization criterion into a single training objective

$$\mathcal{L}_{const} = \mathcal{L}_{MSE}[f(x),\ f_{ema}(x+d)]. \quad (11)$$

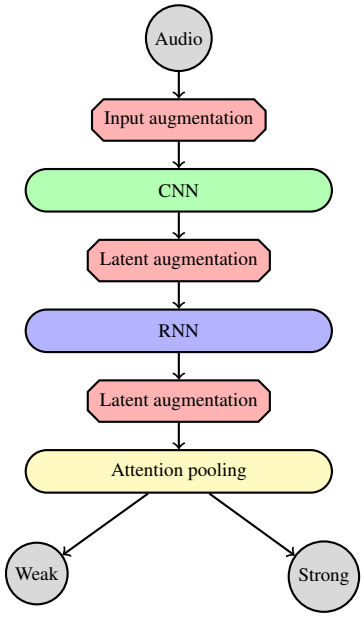


Figure 1: Architecture of the baseline system and location of the data augmentation modules.

However, this method prevents us from fine tuning the trade-off between regularization and consistency in the final training objective. Yet, we have observed that this fine tuning is crucial when the model is dealing with difficult augmentations, such as adversarial attacks (see subsection 5.1).

For this reason, we have also experimented with a more traditional $\mathcal{L}_2$ regularization cost

$$\mathcal{L}_{const} = \mathcal{L}_{MSE}[f(x),\ f(x+d)], \quad (12)$$

$$\mathcal{L}_{reg} = \mathcal{L}_2[f], \quad (13)$$

where $\mathcal{L}_2[f]$ is computed on the model parameters.

5. EXPERIMENTAL RESULTS

The different training methods that we have used and the scores that we have obtained are summarized in Table 1. Hyperparameters have been selected using the validation score computed on a split of $\mathcal{D}_s$.

Table 1: Comparison of training methods. Highest values are shown in bold, and scores above the baseline are underlined. The baseline uses random noise as an augmentation strategy.

	Augmentation	Location		Regularization		Scores	
		Input	Latent	MT	$\mathcal{L}_2$	macro	psds
1	None					0.291	0.500
2	**Baseline**	x		x		0.381	0.552
3	Random	x			x	**0.397**	0.534
4	Random		x		x	0.388	**0.565**
5	Adversarial		x		x	0.374	0.559
6	VAT	x		x		0.311	0.461
7	VAT	x			x	0.358	0.539
8	VAT		x		x	0.362	0.542
9	Mixup		x		x	0.351	0.507

5.1. Separate regularization and consistency costs

The experiments that we conducted on random and adversarial augmentations suggest that it is advantageous to keep a separate regularization and consistency cost.

If we compare lines 2 and 3 of Table 1, which both use input noise as an augmentation strategy and differ only by the regularization method, we notice an improvement in $macro$ score (increasing from 0.381 to 0.397) balanced by a drop in $psds$ score (decreasing from 0.552 to 0.534). This leads to the following observation. Using $\mathcal{L}_2$ regularization instead of MT regularization gives us an additional degree of freedom, which can be used to optimize either the $macro$ score or the $psds$ score. This property is useful. Indeed, $macro$ and $psds$ scores are partially conflicting: depending on the use case, it might be advantageous to optimize either of them [24].

If we compare lines 6 and 7 of Table 1, which both use VAT as an augmentation strategy and differ only by the regularization method, we notice this time an improvement both in $macro$ score (increasing from 0.311 to 0.358) and $psds$ score (increasing from 0.461 to 0.539). We hypothesize that, when the data augmentations become harder to handle for the model, it becomes increasingly advantageous to keep a separate regularization and consistency cost.

5.2. Simplified training method

As we have seen by focusing on the $macro$ metric and comparing lines 2 and 3 of Table 1, the MT objective can be advantageously replaced by a $\mathcal{L}_2$ regularization cost and a consistency cost penalizing the sensitivity of the model f to input noise. Further experiments have shown that this advantage is maintained even when the amount of data used for training is decreased. Moreover, the classification score is improved across classes. This algorithm thus seems to escape the curse of class dependency that has been recently discussed in the literature on regularization and data augmentations [25].

This new method is a simplification of the MT approach. As we have already mentioned, it makes it easier to analyze the individual impact of the training objectives (classification, regularization, and consistency), and to fine-tune their relative contribution during training. Moreover, since the f_{ema} model is no longer necessary, we can divide by two the number of parameters kept in memory during training. Although we did not observe it during our experiments, this method may also lead to faster training. Indeed, the update of f_{ema} is not necessary anymore, and the added regularization cost is already computed implicitly by some implementation of the Adam optimizer (for instance, this is the case in the widely used framework *Pytorch*). However, these advantages come at the cost of an additional hyperparameter to adjust.

5.3. Advantage of depth

The experiments that we conducted on random and adversarial augmentations suggest that it is advantageous to use latent augmentations.

If we compare lines 3 and 4 of Table 1, we notice that using latent noise instead of input noise leads to an increase in $psds$ score. This training method offers a compromise between $macro$ score (decreasing slightly from 0.397 to 0.388) and $psds$ score (increasing from 0.534 to 0.565). This result could be explained by the following. The authors of [2] suggest that adding noise to the input of a classifier moves its decision boundaries away from the training data, and thus improves its generalization power. Following this hypothesis, adding noise to the internal layers of f improves its performance as a classifier. This leads to an improvement of the $psds$ score, which is more sensitive to the labeling accuracy. On the other hand, the $macro$ score, which is more sensitive to the temporal accuracy, does not vary as much.

5.4. Adversarial and Mixup augmentations

Despite encouraging first results, the experiments we performed with adversarial and Mixup augmentations did not yield any improvement over our experiments with random noise. We may explain the failure of adversarial perturbations by observing that the generated samples are not realistic. Consequently, the invariants developed with these methods are not useful for the detection and classification of sound events. This motivates the study of data augmentations that are specific to the audio domain, and are more relevant for this task.

6. CONCLUSION

This article analyzes the impact of several domain agnostic data augmentation strategies on the baseline of task 4 of the DCASE challenge. We propose a simple variant of the MT method, which improves its performance. The results that we obtained with adversarial augmentations suggest that it may be advantageous to restrict the search space to realistic augmentations. In a future study, we will test this hypothesis and examine the impact of audio related augmentations on sound event detection. As the dimension of the augmentation space increases, we anticipate that the search strategy will become a central issue.

7. ACKNOWLEDGMENT

The material contained in this document is based upon work funded by the Agence National de la Recherche en Intelligence Artificielle (PhD program in AI) and Hi! PARIS through its PhD funding program.

8. REFERENCES

[1] Y. Wang, Q. Yao, J. T. Kwok, and L. M. Ni, "Generalizing from a few examples: A survey on few-shot learning," *ACM computing surveys (csur)*, vol. 53, no. 3, pp. 1–34, 2020.

[2] J. E. Van Engelen and H. H. Hoos, "A survey on semi-supervised learning," *Machine Learning*, vol. 109, no. 2, pp. 373–440, 2020.

[3] N. Turpault and R. Serizel, "Training sound event detection on a heterogeneous dataset," *arXiv preprint arXiv:2007.03931*, 2020.

[4] A. Rasmus, M. Berglund, M. Honkala, H. Valpola, and T. Raiko, "Semi-supervised learning with ladder networks," in *Advances in Neural Information Processing Systems*, vol. 28. Curran Associates, Inc., 2015. [Online]. Available: https://proceedings.neurips.cc/paper/2015/hash/378a063b8fdb1db941e34f4bde584c7d-Abstract.html

[5] P. Bachman, O. Alsharif, and D. Precup, "Learning with pseudo-ensembles," in *Advances in Neural Information Processing Systems*, vol. 27. Curran Associates, Inc., 2014. [Online]. Available: https://proceedings.neurips.cc/paper/2014/hash/66be31e4c40d676991f2405aaecc6934-Abstract.html

[6] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[7] S. Laine and T. Aila, "Temporal ensembling for semi-supervised learning," *arXiv preprint arXiv:1610.02242*, 2016.

[8] B. T. Polyak and A. B. Juditsky, "Acceleration of stochastic approximation by averaging," *Journal on control and optimization*, vol. 30, no. 4, pp. 838–855, 1992.

[9] A. Tarvainen and H. Valpola, "Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results," *Advances in neural information processing systems*, vol. 30, 2017.

[10] V. Verma, K. Kawaguchi, A. Lamb, J. Kannala, Y. Bengio, and D. Lopez-Paz, "Interpolation consistency training for semi-supervised learning," *arXiv preprint arXiv:1903.03825*, 2019.

[11] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," *arXiv preprint arXiv:1710.09412*, 2017.

[12] Z. You, J. Ye, K. Li, Z. Xu, and P. Wang, "Adversarial noise layer: Regularize neural network by adding noise," in *International Conference on Image Processing (ICIP)*. IEEE, 2019.

[13] S. Park, J. Park, S.-J. Shin, and I.-C. Moon, "Adversarial dropout for supervised and semi-supervised learning," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 32, no. 1, 2018.

[14] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard, "Universal adversarial perturbations," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 1765–1773.

[15] D. Berthelot, N. Carlini, I. Goodfellow, N. Papernot, A. Oliver, and C. A. Raffel, "Mixmatch: A holistic approach to semi-supervised learning," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[16] A. Agnone and U. Altaf, "Virtual adversarial training system for DCASE 2019 task 4," *Detection and Classification of Acoustics Scenes and Events (DCASE) Challenge*, 2019.

[17] M. Zöhrer and F. Pernkopf, "Virtual adversarial training and data augmentation for acoustic event detection with gated recurrent neural networks." in *Interspeech*, 2017.

[18] R. Harb and F. Pernkopf, "Sound event detection using weakly-labeled semi-supervised data with GCRNNS, vat and self-adaptive label refinement," *arXiv preprint arXiv:1810.06897*, 2018.

[19] H. Dinkel, X. Cai, Z. Yan, Y. Wang, J. Zhang, and Y. Wang, "A lightweight approach for semi-supervised sound event detection with unsupervised data augmentation," in *Proceedings of the 6th Workshop on Detection and Classification of Acoustic Scenes and Events (DCASE), Online*, 2021.

[20] N. Turpault, R. Serizel, A. Parag Shah, and J. Salamon, "Sound event detection in domestic environments with weakly labeled data and soundscape synthesis," in *Workshop on Detection and Classification of Acoustic Scenes and Events*, New York City, United States, October 2019. [Online]. Available: https://hal.inria.fr/hal-02160855

[21] M. Bertola, "Data augmentation methods exploration for sound event detection," DCASE2022 Challenge, Tech. Rep., June 2022.

[22] C. Bilen, G. Ferroni, F. Tuveri, J. Azcarreta, and S. Krstulovic, "A framework for the robust evaluation of sound event detection," *arXiv preprint arXiv:1910.08440*, 2019. [Online]. Available: https://arxiv.org/abs/1910.08440

[23] N. Turpault, R. Serizel, and E. Vincent, "Analysis of weak labels for sound event tagging," Apr. 2021, working paper or preprint. [Online]. Available: https://hal.inria.fr/hal-03203692

[24] G. Ferroni, N. Turpault, J. Azcarreta, F. Tuveri, R. Serizel, Ç. Bilen, and S. Krstulović, "Improving sound event detection metrics: insights from DCASE 2020," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021.

[25] R. Balestriero, L. Bottou, and Y. LeCun, "The effects of regularization and data augmentation are class dependent," *arXiv preprint arXiv:2204.03632*, 2022.

STARSS22: A DATASET OF SPATIAL RECORDINGS OF REAL SCENES WITH SPATIOTEMPORAL ANNOTATIONS OF SOUND EVENTS

Archontis Politis[1], *Kazuki Shimada*[2], *Parthasaarathy Sudarsanam*[1], *Sharath Adavanne*[1], *Daniel Krause*[1]
Yuichiro Koyama[2], *Naoya Takahashi*[2], *Shusuke Takahashi*[2], *Yuki Mitsufuji*[2] *Tuomas Virtanen*[1],

[1] Audio Research Group, Tampere University, Tampere, Finland
[2] Sony Group Corporation, Tokyo, Japan

ABSTRACT

This report presents the Sony-TAu Realistic Spatial Soundscapes 2022 (STARS22) dataset of spatial recordings of real sound scenes collected in various interiors at two different sites. The dataset is captured with a high resolution spherical microphone array and delivered in two 4-channel formats, first-order Ambisonics and tetrahedral microphone array. Sound events belonging to 13 target classes are annotated both temporally and spatially through a combination of human annotation and optical tracking. STARSS22 serves as the development and evaluation dataset for Task 3 (Sound Event Localization and Detection) of the DCASE2022 Challenge and it introduces significant new challenges with regard to the previous iterations, which were based on synthetic data. Additionally, the report introduces the baseline system that accompanies the dataset with emphasis on its differences to the baseline of the previous challenge. Baseline results indicate that with a suitable training strategy a reasonable detection and localization performance can be achieved on real sound scene recordings. The dataset is available in `https://zenodo.org/record/6600531`.

Index Terms— Sound event localization and detection, sound source localization, acoustic scene analysis, microphone arrays

1. INTRODUCTION

Sound event localization and detection (SELD) refers to the task of simultaneously detecting the presence and tracking the location of sound types of interest over time. It relates strongly to the more established tasks of sound event detection (SED) and sound source localization (SSL) but it adds spatial information to the first and semantic information to the second. The SELD task has recently seen increased research interest in part due to its introduction to the DCASE Challenge in 2019 [1]. The challenge dataset was generated with a collection of spatial room impulse responses (SRIRs) from 5 spaces and multiple source positions, convolved with dry isolated sound event recordings [2]. The next iteration of DCASE2020 increased the dataset diversity by including SRIRs of 10 additional rooms with stronger reverberation and, more importantly, by emulating dynamic scenes with both moving and static sound sources [3], while the DCASE2021 dataset introduced additionally directional interfering events out of the target classes [4].

The three SELD datasets of DCASE2019-2021 contributed to the continuous development and improvement of SELD methods by aiming to emulate accurately spatial and acoustical properties of sound scenes and to increase gradually scene complexity towards more realistic conditions. However, there are certain limitations inherent to generating synthetic mixtures. One such limitation is the random presence of target classes and the random sequencing of sound events, discarding natural temporal occurrences or co-occurrences of certain sounds in a real scene. Another limitation is the randomized spatial distribution of sound events ignoring their spatial constraints and connections in a scene. To overcome such limitations, SELD systems should transition to training and evaluation with recordings of real sound scenes. Such datasets require strong event labels provided by human annotators and simultaneous spatial annotations provided by some form of automated tracking. Due to the required annotation effort and complexity, there are no published SELD datasets we know of except for the SECL-UMons one in [5], capturing natural sound events of 11 classes occuring at pre-defined locations in two spaces. However, even though the events have a natural spatial distribution, the dataset is limited to single event recordings in isolation or to combinations of two simultaneous events, ignoring spatio-temporal information linking events in a natural scene. A few more synthetic SELD datasets exist with the same limitations as the DCASE datasets, based on captured SRIRs and targeting certain applications, such as wearable arrays [6] or positional localization in a room with distributed arrays [7].

This report presents the first SELD dataset we are aware of where realistic scenes, loosely acted by multiple actors, are captured and annotated with strong labels temporally and spatially. The challenges of such annotations are dealt with a combination of human listening and optical tracking, employing multiple sensors and modalities. Since the sound scenes are acted naturally, the dataset overcomes the limitations of synthetic datasets discussed earlier. Target sound classes are not combined randomly but are instead constrained by the environment and the participants, while the presence of each class is determined by the natural composition of each scene. Causal and sequential occurrences of sound events, as well as co-occurrences, follow the actions of the actors and their interactions with the environment. The same holds for the location of events and their trajectories in case they are moving; their spatial distributions are naturally constrained by the type of event, while event trajectories can reveal scene information on the agents and their actions. Hence, the dataset opens certain new possibilities for SELD systems apart from evaluation in realistic scenarios.

The STARSS22 dataset serves as the development and evaluation dataset of DCASE2022 Task 3, and it is followed by a suitable baseline and evaluation setup. Changes with respect to the previous DCASE challenges are elaborated. Since the duration of the dataset is limited compared to the synthetic datasets used in previous years, use of external data is allowed in this iteration to improve model training and generalization. An example strategy based on additional synthetic data is presented for the baseline. Finally, results are presented on the development and evaluation set.

2. DATASET

The **Sony-TAu Realistic Spatial Soundscapes 2022** (**STARSS22**) dataset consists of recordings of real scenes captured with a high channel-count spherical microphone array (SMA). The recordings are conducted by two different teams at two different sites, Tampere University facilities in Tampere, Finland, and Sony facilities in Tokyo, Japan. Recordings at both sites share the same capturing and annotation process, organized in sessions corresponding to distinct rooms, human participants, and sound making props with a few exceptions. In each session, various clips are recorded with combinations of that session's participants acting some simple scenes and interacting between them and with the sound props. The scenes are not strongly scripted; they are based on generic instructions on the desired sound events and are otherwise improvised by the participants. The instructions serve as a rough guide to ensure adequate event activity and occurrences of the target sound classes in a clip.

Similarly to the previous challenges, the recordings are converted to two 4-channel spatial formats: first-order Ambisonics (FOA) and tetrahedral microphone array (MIC), both derived from the original 32-channel recordings. Conversion of the Eigenmike recordings to FOA following the SN3D normalization scheme (or ambiX) was performed with measurement-based filters according to [8]. Regarding the MIC format, channels 6, 10, 26, and 22 of the Eigenmike were selected, corresponding to a nearly tetrahedral arrangement. Analytical expressions of the directional responses of each format can be found in the DCASE2020 challenge report [3]. Finally, the converted recordings were downsampled to 24kHz.

The dataset is split into a development set (*dev-set*) and evaluation set (*eval-set*). The development set totals about 4 hrs 52 mins, of which 70 recording clips amounting to about 2 hrs are recorded in 4 different rooms in Tokyo and 51 recordings amounting to about 3 hrs are recorded in 7 different rooms in Tampere. To aid the development process, the development set is further split into a training part (*dev-set-train*, 40+27 clips in 2+4 rooms in Tokyo+Tampere) and a testing part (*dev-set-test*, 30+24 clips in 2+3 rooms in Tokyo+Tampere). The evaluation set is close to 2 hrs, recorded in 2 different rooms in Tokyo (35 clips) and in 3 different rooms in Tampere (17 clips).

2.1. Recording setup and process

Each scene was captured with 4 types of sensors: a) a high resolution 32-channel SMA (Eigenmike em32[1]) recording the main multichannel audio for the challenge, b) a 360°camera (Ricoh Theta V[2]) mounted about 10 cm above the SMA, c) a motion capture (mocap) system of infrared cameras surrounding the scene, tracking reflective markers mounted on the main actors and sound sources of interest (Optitrack Flex 13[3]), and d) wireless microphones mounted on the same tracked actors and sound sources, providing close-miked recordings of the main sound events (Røde Wireless Go II[4]). For each recording session, a suitable position of the Eigenmike and Ricoh Theta V was determined in order to cover the scene from a central position, while taking into account the intended scenarios and the specific room constraints. The origin of the mocap system was then set at ground level on the same position and the height of the Eigenmike was set at 1.5 m, while the mocap cameras were positioned at the boundaries of the room. Tracking markers were mounted to independent sound sources (such as next to the water sink, on a mobile phone on a table, on a hoover, or next to a guitar's soundhole). Head markers were additionally provided to the participants before each scene recording, in the form of headbands or hats. Tracking the head served as the reference point for all human made sounds. Mouth position for *speech* and *laughter* sounds, feet stepping position for *footstep* sounds, and hand position for *clapping* sounds were each approximated with a fixed translation from the head-tracking center close to the top of the head. Regarding clapping, participants were instructed to clap about 20 cm in front of their face to improve the position approximation. Head rotations were also logged during the scene with respect to the global coordinate frame of the mocap system. Finally, the wireless microphones were mounted to the lapel of each actor and to additional independent sound sources. A clapper sound was used to initiate the acting and to serve as a reference signal for synchronization between the different types of recordings.

Target Class	Related Audioset subclasses
Telephone	*Telephone bell ringing*, *Ringtone* (no musical ringtones)
Domestic sounds	*Vacuum cleaner, Mechanical fan, Boiling* (produced by hoover, air circulator, water boiler)
Door, open or close	Combination of *Door* & *Cupboard, open or close*
Music	*Background music* & *Pop music*, (played by a loudspeaker in the room)
Musical instrument	*Acoustic guitar, Marimba, Xylophone, Cowbell, Piano, Rattle (instrument)*
Bell	Combination of sounds from hotel bell and glass bell, closer to *Bicycle bell* & single *Chime*

Table 1: Relation of target classes to specific Audioset classes. Target classes not included in the table have an one-to-one relationship with the similarly named Audioset ones.

2.2. Annotation process

Spatiotemporal annotations of the sound events were conducted manually by the authors and research assistants. Three types of information were required in order to obtain such annotations: a) the subset of the target classes that were active in each scene, b) the temporal activity of such class instances, and c) the position of each such instance when active. (a) was observed and logged during each scene recording. (b) was manually annotated by listening to the wireless microphone recordings. Since each such microphone would capture prominently sounds produced by the human actor or source it was assigned to, onset, offsets, source, and class information of each event could be conveniently extracted. In scenes or instances where associating an event to a source was ambiguous purely by listening, annotators would consult the video recordings to establish the correct association. The temporal annotation resolution was set to 100 msec.

After onset, offset, and class information of events was established for each source and actor in the scene, the positional annotations (c) were extracted for each such event by masking the tracker data with the temporal activity window of the event. Additionally,

[1] https://mhacoustics.com/products#eigenmike1
[2] https://theta360.com/en/about/theta/v.html
[3] https://optitrack.com/cameras/flex-13/
[4] https://rode.com/en/microphones/wireless/wirelessgoii

	Global	Fem. speech	Male speech	Clap	Phone	Laugh	Dom. sounds	Footsteps	Door	Music	Music. instr.	Faucet	Bell	Knock
Frame coverage (% total frames)	84.7	20.4	37.6	0.7	1.4	2.7	17.9	1.3	0.6	29.4	4.0	1.7	1.5	0.1
Max. polyphony	5	2	3	2	1	4	1	1	1	1	4	1	1	1
Mean polyphony	1.5	1.04	1.07	1.17	1.00	1.18	1.00	1.00	1.00	1.00	1.86	1.00	1.00	1.00
Polyphony 1 (% active frames)	61.5	96.1	93.3	83.4	100	84.0	100	100	100	100	52.2	100	100	100
Polyphony 2	29.55	3.9	6.5	16.6	0	14.5	0	0	0	0	16.6	0	0	0
Polyphony 3	7.15	0	0.2	0	0	1.1	0	0	0	0	24.2	0	0	0
Polyphony 4	1.6	0	0	0	0	0.4	0	0	0	0	7.0	0	0	0
Polyphony 5	0.2	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 2: Dataset class activity information. The mean polyphony is computed over active frames only having one or more events present.

class-specific translations to the tracking data were applied if necessary, as mentioned earlier for most human made sounds. Positional information was logged in Cartesian coordinates with respect to the mocap system's origin and subsequently converted to directions-of-arrival with respect to the center of the Eigenmike. Finally, the class, temporal, and spatial annotations were combined and converted to the text format used in the previous DCASE2019-2021 challenges. Validation of the annotations was performed by observing and listening to the 360° videos, overlaid with labeled markers positioned at the DOAs of the annotated events on the 360° video plane.

2.3. Target sound classes

A set of 13 target sound classes are selected to be annotated, based on the sound events captured prominently in the recorded scenes. The class labels are selected to conform to the Audioset ontology [9] and they are: ***female speech/woman speaking, male speech/man speaking, clapping, telephone, laughter, domestic sounds, walk/footsteps, door open or close, music, musical instrument, water tap/faucet, bell, knock***. Since some of these labels correspond to superclasses with a large diversity of sounds and number of subclasses in the ontology (e.g. *domestic sounds* or *musical instrument*) we provide some additional information on the subset of sounds encountered in the recordings for some of the target classes, in the form of more specific audioset-related labels. This information is summarized in Table 1 and it can aid training and testing of methods. Certain directional sound events in the recordings are not annotated and are treated as directional interferers; examples include *computer keyboard*, *shuffling cards*, and *dishes, pots, and pans*. Additionally, there is natural background noise in all recordings, mostly HVAC-related, ranging from low to considerable levels. Based on the annotations, information on the percentage of frames that each class is active and the degree of polyphony globally and of each class separately is presented in Table 2.

3. BASELINE

3.1. Model architecture

The baseline of the DCASE2022 Task 3 challenge is similar to the one used in used in DCASE2021; a SELDnet-inspired CRNN architecture [10] improved with the ACCDOA output representation and loss [11]. However, due to the inability of the original ACCDOA representation to handle co-occuring events of the same class, the baseline adopts the recent *multi-ACCDOA* (mACCDOA) extension [12]. The mACCDOA model receives a sequence of T STFT frames of multichannel features and outputs $T/5 \times N \times C \times 3$ vector coordinates, where C is the number of target classes and N the maximum assumed number of co-occuring events in the recordings. For the current baseline N is set to 3 maximum simultaneous sources, while a value of 0.5 is used as the threshold on the length of the output vectors to indicate track and class activity. Note that a reduction of the STFT temporal resolution by a factor of 5 is performed to match the resolution of the annotations at every 100 msec.

Input features remain similar to the previous challenge [3]; namely, 4 channel 64-band log-mel spectrograms combined with acoustic intensity vectors for the FOA format or combined with generalized cross-correlation (GCC) sequences for the MIC format, following [13]. Additionally, the option of the *SALSA-lite* spatial features for the MIC format is added in the current baseline, recently shown to offer better performance than GCC in multi-source scenarios [14]. In this case, the original STFT spectrograms and the SALSA-lite features are truncated to include bins up to about 9 kHz, without mel-band aggregation, following [14]. A block diagram of the model architecture is presented in Fig. 1.

3.2. Model training

The baseline model is trained and evaluated twice: firstly only on the development set reporting baseline results for the participants to compare against during development. Secondly, it is trained on the development set and tested on the evaluation set, with results reported after the completion of the evaluation phase of the challenge. Since, the amount of training material is insufficient for the complexity of the task, additional material is synthesized for training. Those synthetic mixtures are generated with the same generation method and SRIRs as the DCASE2020-2021 datasets [3, 4]. 1200 one-minute spatial mixtures are synthesized (*synth-set*) using measured SRIRs from 9 rooms in TAU and sound event samples sourced from FSD50K [15]. The samples are chosen to match the target classes on the basis of their annotated labels which follow the Audioset ontology. The synthetic mixtures are made publicly available for reproducibility[5] along with the list of the selected FSD50K sound samples. Additionally, the SRIRs are also publicly shared[6] along with the scene generation code[7], so that participants can generate their own synthetic mixtures for training following the same process if desired. The sets and splits for training and testing of the baseline for each phase are summarized in Table 3.

[5] https://doi.org/10.5281/zenodo.6406873

[6] https://doi.org/10.5281/zenodo.6408611

[7] https://github.com/danielkrause/DCASE2022-data-generator

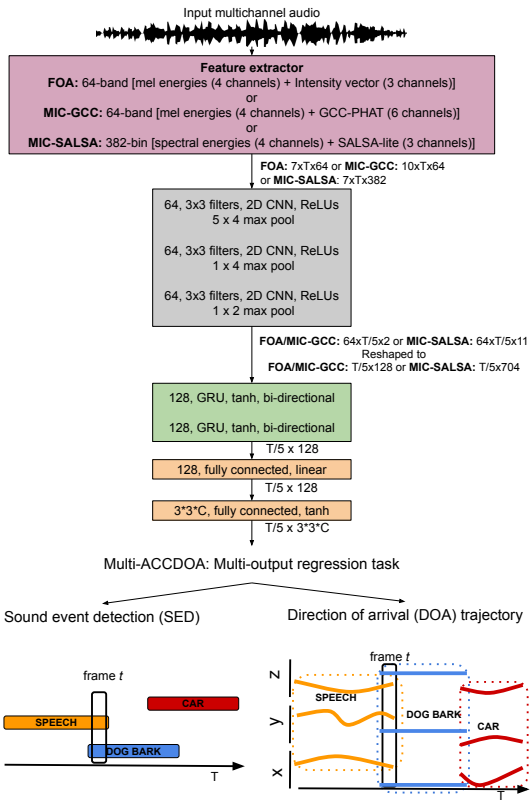


Figure 1: Baseline CRNN model with mACCDOA output.

4. EVALUATION

Intermediate development set results are reported by the participants in the development set, while evaluation set results are computed by the organizers based on the submitted system outputs on the unseen evaluation set. Contrary to the previous challenges, participants are allowed to use external data during training, such as sample banks of sound events, room simulators, SRIR databases, spatial background noise recordings, pre-trained networks, and others. Generating the *synth-set* dataset to improve the baseline performance constitutes just one such example of external data usage.

The submissions are evaluated with the joint localization-detection metrics studied in [16, 1] and introduced first-time in DCASE2020. These are the location-dependent error rate (ER_{20°) and F1-score (F_{20°) for a spatial threshold 20° and the class-dependent localizastion error (LE_{CD}) and localization recall (LR_{CD}). Contrary to the previous challenges, in which F_{20° was micro-averaged, in this challenge evaluation is based on macro-averaging of F1-score to account better for the imbalanced presence of the target classes in the dataset.

Phase	Training	Testing
Development	synth-set + dev-set-train	dev-set-test
Evaluation	synth-set + dev-set-train + dev-set-test	eval-set

Table 3: Datasets & splits used for baseline training and evaluation.

	$ER_{20^\circ} \downarrow$	$F_{20^\circ} \uparrow$ (macro)	$F_{20^\circ} \uparrow$ (micro)	$LE_{CD} \downarrow$	$LR_{CD} \uparrow$
Development set					
FOA-real	0.78	0.11	-	64.1°	0.24
FOA-mixed	0.71	0.21	0.36	29.3°	0.46
MIC-mixed	0.71	0.18	0.36	32.2°	0.47
Evaluation set					
FOA-mixed	0.61	0.24	0.39	22.9°	0.51
MIC-mixed	0.61	0.22	0.41	25.9°	0.48

Table 4: Baseline results on development and evaluation set. *FOA-real* refers to training only on the development set of STARSS22, *FOA/MIC-mixed* refers to training using additionally synthetic data.

4.1. Results

Results of the baseline on the development and evaluation set are presented on Table 4, for both FOA and MIC formats. The baseline was trained as indicated in Sec. 3.2 using the additional synthetic spatial mixtures of *synth-set*. For comparison purposes, an example of the model with FOA input trained only with real recordings is also reported (FOA-real), with the training and testing splits of Table 3 excluding the synthetic data (*synth-set*). It can be seen that the performance is very low in this case, at least without using data augmentation strategies. Two training strategies were tested with regards to incorporating the synthetic data. The first was based on initial training of the model on the synthetic data, followed by fine-tuning with the development dataset. The second simply mixed both the synthetic and the development recordings and trained with the combined dataset. Better results were obtained with the mixed strategy and these are the ones presented here (FOA/MIC-mixed). It is noted that the SRIRs used for the generation of *synth-set* were captured in TAU spaces that were different than the ones were the scene recordings of the STARS22 dataset occurred. Regarding the MIC format, both the GCC features and the SALSA-lite features were tested. Slightly better results were obtained with the GCC features and reported here. That may be attributed to the fact that even though the SALSA-lite features show a clear advantage for densely populated multi-source scenes such as the ones in DCASE2021 dataset [14], for the more sparse scenes of STARSS22 that advantage may be diminished. Finally, both the micro and macro versions of the F1-score are presented here, with a clear drop in performance in the macro version, as expected with a dataset of such unbalanced presence of target classes (evident in Table 2).

5. CONCLUSIONS

This report presents the specifications of the STARS22 dataset, intended for evaluation of SELD systems in challenging real conditions with a natural composition of sound events. The dataset serves as the development and evaluation dataset of the SELD challenge of DCASE2022 and it is accompanied by a baseline model which, with use of external data and a suitable training strategy, can achieve a reasonable performance on the evaluation dataset.

6. ACKNOWLEDGMENT

The dataset collection and annotation at Tampere University has been funded by Google. This work was carried out with the support of Centre for Immersive Visual Technologies (CIVIT) research infrastructure, Tampere University, Finland.

7. REFERENCES

[1] A. Politis, A. Mesaros, S. Adavanne, T. Heittola, and T. Virtanen, "Overview and evaluation of sound event localization and detection in DCASE 2019," *IEEE/ACM Trans. Audio, Speech, and Language Proc.*, 2020.

[2] S. Adavanne, A. Politis, and T. Virtanen, "A multi-room reverberant dataset for sound event localization and detection," in *Work. on Detection and Classification of Acoustic Scenes and Events (DCASE)*, October 2019, pp. 10–14.

[3] A. Politis, S. Adavanne, and T. Virtanen, "A dataset of reverberant spatial sound scenes with moving sources for sound event localization and detection," in *Work. on Detection and Classification of Acoustic Scenes and Events (DCASE)*, 2020, pp. 165–169.

[4] A. Politis, S. Adavanne, D. Krause, A. Deleforge, P. Srivastava, and T. Virtanen, "A dataset of dynamic reverberant sound scenes with directional interferers for sound event localization and detection," in *Work. on Detection and Classification of Acoustic Scenes and Events (DCASE)*, 2021, pp. 125–129.

[5] M. Brousmiche, J. Rouat, and S. Dupont, "Secl-umons database for sound event classification and localization," in *IEEE Int. Conf. on Acoustics, Speech and Sig. Proc. (ICASSP)*, 2020, pp. 756–760.

[6] K. Nagatomo, M. Yasuda, K. Yatabe, S. Saito, and Y. Oikawa, "Wearable seld dataset: Dataset for sound event localization and detection using wearable devices around head," in *IEEE Int. Conf. on Acoustics, Speech and Sig. Proc. (ICASSP)*, 2022, pp. 156–160.

[7] E. Guizzo, R. F. Gramaccioni, S. Jamili, C. Marinoni, E. Massaro, C. Medaglia, G. Nachira, L. Nucciarelli, L. Paglialunga, M. Pennese, *et al.*, "L3DAS21 challenge: Machine learning for 3D audio signal processing," in *IEEE Int. Work. on Machine Learning for Sig. Proc. (MLSP)*, 2021, pp. 1–6.

[8] A. Politis and H. Gamper, "Comparing modeled and measurement-based spherical harmonic encoding filters for spherical microphone arrays," in *IEEE Work. on Applications of Sig. Proc. to Audio and Acoustics (WASPAA)*, 2017, pp. 224–228.

[9] J. F. Gemmeke, D. P. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio set: An ontology and human-labeled dataset for audio events," in *IEEE Int. Conf. on Acoustics, Speech and Sig. Proc. (ICASSP)*, 2017, pp. 776–780.

[10] S. Adavanne, A. Politis, J. Nikunen, and T. Virtanen, "Sound event localization and detection of overlapping sources using convolutional recurrent neural networks," *IEEE J. Selected Topics in Sig. Proc.*, vol. 13, no. 1, pp. 34–48, 2018.

[11] K. Shimada, Y. Koyama, N. Takahashi, S. Takahashi, and Y. Mitsufuji, "ACCDOA: Activity-coupled cartesian direction of arrival representation for sound event localization and detection," in *IEEE Int. Conf. on Acoustics, Speech and Sig. Proc. (ICASSP)*, 2021, pp. 915–919.

[12] K. Shimada, Y. Koyama, S. Takahashi, N. Takahashi, E. Tsunoo, and Y. Mitsufuji, "Multi-accdoa: Localizing and detecting overlapping sounds from the same class with auxiliary duplicating permutation invariant training," in *IEEE Int. Conf. on Acoustics, Speech and Sig. Proc. (ICASSP)*, 2022, pp. 316–320.

[13] Y. Cao, Q. Kong, T. Iqbal, F. An, W. Wang, and M. Plumbley, "Polyphonic sound event detection and localization using a two-stage strategy," in *Work. on Detection and Classification of Acoustic Scenes and Events (DCASE)*, 2019.

[14] T. N. T. Nguyen, D. L. Jones, K. N. Watcharasupat, H. Phan, and W.-S. Gan, "Salsa-lite: A fast and effective feature for polyphonic sound event localization and detection with microphone arrays," in *IEEE Int. Conf. on Acoustics, Speech and Sig. Proc. (ICASSP)*, 2022, pp. 716–720.

[15] E. Fonseca, X. Favory, J. Pons, F. Font, and X. Serra, "Fsd50k: an open dataset of human-labeled sound events," *IEEE/ACM Trans. on Audio, Speech, and Language Proc.*, vol. 30, pp. 829–852, 2021.

[16] A. Mesaros, S. Adavanne, A. Politis, T. Heittola, and T. Virtanen, "Joint measurement of localization and detection of sound events," in *IEEE Work. on Applications of Sig. Proc. to Audio and Acoustics (WASPAA)*, 2019, pp. 333–337.

IMPROVING NATURAL-LANGUAGE-BASED AUDIO RETRIEVAL WITH TRANSFER LEARNING AND AUDIO & TEXT AUGMENTATIONS

Paul Primus[1], *Gerhard Widmer*[1,2]

[1]Institute of Computational Perception (CP-JKU)
[2]LIT Artificial Intelligence Lab
Johannes Kepler University, Austria

ABSTRACT

The absence of large labeled datasets remains a significant challenge in many application areas of deep learning. Researchers and practitioners typically resort to transfer learning and data augmentation to alleviate this issue. We study these strategies in the context of audio retrieval with natural language queries (Task 6b of the DCASE 2022 Challenge). Our proposed system uses pretrained embedding models to project recordings and textual descriptions into a shared audio-caption space in which related examples from different modalities are close. We employ various data augmentation techniques on audio and text inputs and systematically tune their corresponding hyperparameters with sequential model-based optimization. Our results show that the used augmentations strategies reduce overfitting and improve retrieval performance.

Index Terms— Language-based Audio Retrieval, Transfer Learning, Audio Augmentation, Text Augmentation

1. INTRODUCTION

Natural-language-based audio retrieval is concerned with ranking audio recordings depending on their content's similarity to textual descriptions. Retrieval tasks like this are typically solved by converting recordings and textual descriptions into high-level representations and then aligning them in a shared audio-caption space; ranking can then be done based on the distance between embeddings. These systems' retrieval performance highly depends on the quality of the audio and text embedding models, which must extract features that accurately and discriminatively represent the high-level content. Current state-of-the-art approaches [1, 2, 3] create such feature extractors by training models with millions of parameters directly from raw input features, i.e., deep learning. These large embedding models require a large number of training examples, such as the 400 million image-text pairs used to train CLIP [4], a cutting-edge image-retrieval model. However, publicly available audio-caption datasets like Clotho and AudioCaps are significantly smaller. This work showcases how to use off-the-shelf pretrained audio and text neural networks to create a state-of-the-art retrieval model under this limiting condition. We evaluate our approach in the context of task 6b[1] of the 2022's DCASE Challenge [5], which is concerned with audio retrieval from natural language descriptions. We demonstrate how an already well-performing baseline model can be further improved by using a range of audio and text augmentation methods and pretraining on AudioSet.

[1]https://dcase.community/challenge2022/task-language-based-audio-retrieval

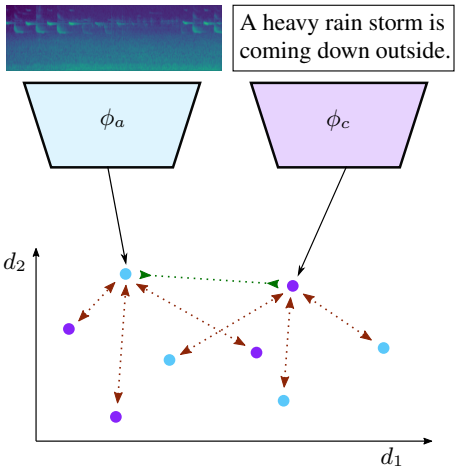


Figure 1: The proposed audio-retrieval system in a nutshell: Audio and descriptions are transformed into the shared audio-caption embedding space via the audio and description embedding models ϕ_a and ϕ_c, respectively. The contrastive loss maximizes the similarities between matching pairs.

2. RELATED WORK

The idea of aligning text and audio features for content-based retrieval is not new: Early audio retrieval methods connected bag-of-words text queries and MFCC features via density or discriminative models [6]. However, the handcrafted features and the relatively small vocabulary limited these methods' performance. Current methods build on top of learnable feature extractors that produce high-level audio and text representations from raw input features. Xie et al. [2], for example, used a convolutional recurrent neural network to extract frame-wise acoustic embeddings and aligned those to Word2Vec features via a linear transformation. Recently, language-based audio retrieval has received increased attention due to the newly introduced task 6b in the 2022's DCASE challenge [5]. The task's objective was to create a retrieval system that takes natural-language queries as input and retrieves the ten best-matching recordings from a test set. The top ranking systems among the nine submitted ones leveraged large pretrained audio and text embedding models like CNN14 [7] and BERT [8], respectively. While most systems applied SpecAugment [9], other data augmentation methods, especially text augmentations, have received little to no attention. We address this paucity and study a range of audio and text augmentation methods in the context of audio retrieval.

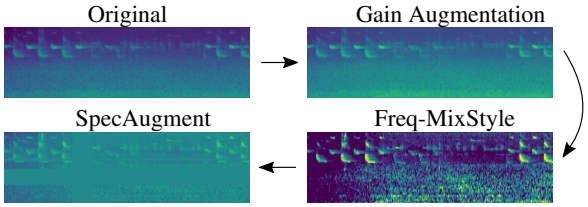


Figure 2: Overview of the audio augmentation pipeline.

Augmentation	Caption
Original	The rain pours down.
Back Translation	It rains cats and dogs.
Insert	It tree rains cats and dogs.
Delete	It rains cats and ~~dogs~~.
Swap	It and cats rains dogs.
Synonym	It drizzles cats and dogs.

Table 1: Overview of the text augmentation pipeline

3. RETRIEVAL SYSTEM

Our model uses separate audio and caption embedding networks $\phi_a(\cdot)$ and $\phi_c(\cdot)$ to embed tuples of spectrograms and descriptions $\{(a_i, c_i)\}_{i=1}^{N}$ into a shared D-dimensional space in a manner that representations of matching audio-caption pairs are close. This behavior is achieved by contrastive training, which equalizes the embeddings of matching audio-caption pairs (a_i, c_i), while pushing the representations of mismatching pairs $(a_i, c_{j;j\neq i})$ apart. The agreement between audio a_i and description c_j is estimated via the normalized dot product in the shared embedding space:

$$C_{ij} = \frac{\phi_{\mathrm{a}}(a_i)^T \cdot \phi_{\mathrm{c}}(c_j)}{\|\phi_{\mathrm{a}}(a_i)\|^2 \, \|\phi_{\mathrm{t}}(c_j)\|^2}$$

The similarity matrix $\mathbf{C} \in \mathbb{R}^{N\times N}$ holds the agreement of matching pairs on the diagonal and the agreement of mismatching pairs off-diagonal. We train the system using the NT-Xent [10] loss, which is defined as the average Cross Entropy (CE) loss over the audio and text dimension; the ground truth is given by the identity matrix $\mathbf{I} \in \mathbb{R}^{N\times N}$:

$$\mathcal{L} = \frac{1}{2\cdot N}\sum_{i=1}^{N} \mathrm{CE}(\mathbf{C}_{i*}, \mathbf{I}_{i*}) + \mathrm{CE}(\mathbf{C}_{*i}, \mathbf{I}_{*i})$$

4. AUDIO AUGMENTATIONS

To reduce overfitting of the audio embedding model and improve generalization, we employ three regularization techniques during training: Gain augmentation, MixStyle [11] along the frequency dimension (Freq-MixStype), and SpecAugment [9]. Figure 2 gives an overview of the audio augmentation pipeline.

Gain Augmentation tries to make the model invariant changes in volume by randomly altering the loudness of the raw audio input signal. Volume manipulations are done by multiplying the waveform with factor W:

$$w = 10^{(g/20)}$$

The change in volume (in dB) is controlled with hyperparameter g; its value is randomly drawn from a uniform distribution in the range $[-g_{\max}, g_{\max}]$.

SpecAugment [9] randomly masks time and frequency stripes in the input spectrogram, thereby reducing the audio embedding model's reliance on specific input patterns. The number of stripes along the time and frequency dimensions is controlled via hyperparameters n_f and n_t, respectively. Parameters w_f and w_t control the maximum width of the time and frequency stripes, respectively. The actual width and the offset of the stripes are chosen from a uniform distribution; masked values are replaced with zeros. We omitted the warping transformation proposed in the original work as it is computationally expensive and reportedly only lead to marginal improvements.

Freq-MixStyle [11] aims to transfer device-style characteristics between recordings by exchanging statistics along the frequency dimension of spectrograms. To this end, the original spectrogram is first normalized to zero mean unit variance along the frequency dimension and then un-normalized with adjusted mean and standard deviation statistics. The adjusted statistics are a convex combination of the original statistics $(\mu_i, \sigma - i)$ and the statistic of a randomly selected spectrogram (μ_j, σ_j):

$$\mu_{\text{new}} = \lambda\mu_i + (1-\lambda)\mu_j$$

$$\sigma_{\text{new}} = \lambda\sigma_i + (1-\lambda)\sigma_j$$

The coefficient λ is drawn from a symmetric beta distribution in a manner that the original statistics always receive a higher weight:

$$\lambda \sim \mathrm{Beta}(\alpha, \alpha)$$

α controls the shape of the Beta distribution. Freq-MixStyle is applied to each input example with a probability of p_{MS}.

5. TEXT AUGMENTATIONS

We apply Back Translation [12] and Easy Data Augmentation [13] (in that order) to reduce overfitting of the sentence embedding model. Examples of these augmentations are given in Table 1.

Back Translation (BT) [12] introduces variation into the input sentence without changing its semantics by translating the input sentence to a foreign language and back to the source language. We translate the training captions from English to German, French, or Spanish, and back to English using Google Translate.

Easy Data Augmentation (EDA) [13] chooses one of four word-level manipulations and applies the selected operation to each word with a certain probability (indicated in parenthesis): insertion of a random word (p_{ins}), deletion (p_{del}), swap with another word in the sentence (p_{swp}), or replacement with a synonym according to WordNet [14] (p_{syn}). EDA is applied with a probability of p_{EDA}.

6. EXPERIMENTS

We first established a baseline without augmentation and then conducted a series of experiments to investigate the impact of using pretrained weights for the audio embedding model, augmenting the

audio and text inputs, and pretraining on AudioCaps. We further investigate the impact of all augmentation methods separately in an ablation study. The model architecture and the exact experimental setup are discussed below.

6.1. Dataset & Input Features

We trained our proposed system on ClothoV2.1 [15], which contains 10-30 second long audio recordings sampled at 32kHz and five human-generated captions for each recording. We used the training, validation, and test split into 3839, 1045, and 1045 examples, respectively, as suggested by the dataset's creators. To make processing in batches easier, we zero-padded all audio snippets to the maximum audio length in the batch. The resulting waveforms were converted to 64-bin log-MEL spectrograms using a 1024-point FFT (32ms) and hop size of 320 (10ms). The audio features were normalized via batch normalization [16] along the frequency dimension before feeding them into the CNN10 embedding model. The input sentences were pre-processed by converting all characters to lowercase and removing punctuation. The resulting strings were tokenized with the WordPiece tokenizer [17], padded to the maximum sequence length in the batch, and truncated to 32 tokens.

6.2. Audio Embedding Model

We used a slightly modified version of the popular CNN10 architecture [7] to embed spectrograms into the 1024-dimensional audio-caption space. The architecture is detailed in Table 2. The network aggregates the output after the last convolutional block over the frequency and time dimensions and transforms the result with a two-layer neural network. The audio embedding model has approximately 9 Million parameters. We chose this simple architecture because it allowed us to train on a single customer-grade GPU with reasonable batch size. For the experiments with pretrained audio embedding model parameters, we transferred the weights of the convolutional blocks from a custom pretrained audio tagger and randomly initialized the fully-connected layers. The data set used for pretraining the embedding model, AudioSet [18], contains approximately 2 Million ten-second audio recordings labeled for 527 hierarchically organized classes. We used the pre-defined split of AudioSet into a large, unbalanced set for training and two smaller, more balanced sets for validation and testing. Pretraining of the embedding model was done as described in [7].

CNN10
$2 \times (3 \times 3)$@64, BN, ReLU
Pool (2×2)
$2 \times (3 \times 3)$@128, BN, ReLU
Pool (2×2)
$2 \times (3 \times 3)$@256, BN, ReLU
Pool (2×2)
$2 \times (3 \times 3)$@512, BN, ReLU
Pool (2×2)
Frequency Pooling (mean)
Time Pooling (average of mean and max)
FC 2048, ReLU
FC 1024

Table 2: The architecture of the audio embedding model (CNN10).

6.3. Text Embedding Model

We used a pretrained BERT model [8] ('bert-base-uncased') to generate embeddings for the audio captions. BERT is a bi-directional self-attention-based sentence encoder that was pretrained on BookCorpus [19] and WikiText datasets [20] for masked language modeling and next sentence prediction. The learned semantic representations proved effective in multiple downstream tasks. We projected the output vector that corresponds to the class token into the shared audio-caption space by using a neural network with one hidden layer of size 2048 and ReLu activations. The text embedding model has approximately 112 Million parameters.

6.4. Training & Evaluation

We train all variants of the proposed system on CLothoV2.1's training set, select hyperparameters according to the performance on the validation set, and report the final results on the test set in section 7. Our main evaluation criterion was the mean Average Precision among the top-10 results (mAP) because this criterion takes the rank of the correct recording into account. We also report the recall among the top-1, top-5, and top-10 retrieved results. All results are averaged over three runs. Both embedding models were jointly optimized using gradient descent with a batch size of 30. We used the Adam update rule [21] for 50 epochs, set the initial learning rate to 10^{-4}, and dropped it by a factor of 3 every 10 epochs. The hyperparameters of the optimizer were set to PyTorch's [22] defaults.

6.5. Sequential Model-Based Optimization

We performed sequential model-based optimization (SMBO) in the hyperparameter space of the audio and text augmentations to optimize the mAP-score on the validation set without manual tuning. Sequential Model-based Optimization (SMBO) utilizes the outcomes of prior experiments to build a surrogate model that estimates the relationship between validation-mAP and a given parameter configuration. Subsequent runs use this surrogate model to sample hyperparameter configurations from a distribution that is proportional to the expected mAP improvement. We initialized SMBO with ten runs using randomly chosen hyperparameters. After that, we performed 100 trials with hyperparameters sampled using the Tree-structured Parzen Estimator algorithm [23]. To reduce the overall computation time, we stopped runs for which the mAP on the validation set did not increase for ten consecutive epochs. Table 4 defines the hyperparameter search space for the SMBO.

7. RESULTS & DISCUSSION

The results of our experiments are summarized in Table 3 and discussed in the following section.

	R@1	R@5	R@10	mAP@10
DCASE baseline	3.50	11.50	19.50	7.50 ± 0.00
baseline	6.63	20.06	31.52	12.53 ± 0.08
+ AudioSet pretraining	13.18	35.30	48.61	22.80 ± 0.29
+ augmentations	14.50	37.24	51.04	24.27 ± 0.19
+ AudioCaps pretraining	14.34	38.12	52.04	24.57 ± 0.15

Table 3: Audio retrieval performance of the DCASE baseline and the custom system in four variants.

7.1. Baseline

The performance of our custom baseline system, which was trained with randomly initialized audio embedding model parameters and without augmentation, is given in Table 3. The resulting system's mAP is 5 pp. higher than the mAP of the DCASE baseline system [5] which we attribute to the more powerful text embedding model (we used BERT [8] instead of Word2Vec [24]).

7.2. AudioSet Pretraining

Next, we investigated the impact of using pretrained weights to initialize the audio embedding model. To that end, we retrained our baseline system but transferred the initial audio embedding model parameters from a CNN10 pretrained for tagging on AudioSet. The resulting model is approximately 10 pp. mAP better than the system that used randomly initialized weights for the audio encoder. This confirms that using pretrained audio embedding models is an effective strategy to alleviate the data scarcity problem.

7.3. Augmentations

We build upon the system that uses the AudioSet pretrained embedding model and perform SMBO to find a good hyperparameter configuration. The resulting best hyperparameters on the validation set and the performance on the test set are given in Tables 4 and 3, respectively. The best configuration found suggests that the parameter which controls the frequency of EDA is superfluous as the best value is very close to one. Synonym replacement appears beneficial, and future experiments should search for the optimal value for this parameter in a larger range. The probability of swapping and inserting random words is close to zero, suggesting that these two transformations are less beneficial or even detrimental. All in all, we observed an absolute improvement of approximately 1.5 pp. mAP when training with text and audio augmentations.

	Augmentation	Parameter	Range	best
Text	EDA	p_{EDA}	$[0, 1.0]$	.9936
		p_{syn}	$[0, 0.3]$	.2962
		p_{swp}	$[0, 0.3]$	.0085
		p_{ins}	$[0, 0.3]$	.0269
		p_{del}	$[0, 0.3]$	.1944
	Backtranslation	p_{bt}	$[0, 1.0]$	.1812
Audio	SpecAugment	n_{f}	$\{0, 1\}$	1
		w_{f}	$\{1, \ldots, 32\}$	4
		n_{t}	$\{0, \ldots, 8\}$	7
		w_{t}	$\{1, \ldots, 64\}$	58
	Audio Gain	g_{max}	$\{0, \ldots, 6\}$	3
	Freq-MixStyle	p_{MS}	$[0, 1.0]$	.1045
		α	$[0, 1.0]$	.8286

Table 4: Hyperparameter search space for the sequential model-based optimization, and the best configuration found.

7.4. AudioCaps Pretraining

We hypothesized that pretraining the retrieval system on additional audio-caption pairs could further improve audio-retrieval results; we, therefore, pretrained the system (with AudioSet pretraining and augmentations) on the $46K$ training examples in AudioCaps [25]. To this end, we used the same training procedure as described in Section 6.4 for pretraining and fine-tuning but decreased the initial learning rate for fine-tuning by a factor of 10. Table 3 gives the results. Pretraining on AudioCaps resulted only in a marginal improvement of 0.3pp mAP, which suggests that using AudioCaps for transfer learning in this naive way has no significant impact.

7.5. Ablation Study: Augmentations

Based on the previous results, we performed another ablation study to investigate the effect of the audio and text augmentations. To that end, we re-trained the system with AudioSet pretraining and augmentations twice: once without audio augmentations and once without text augmentations. The results are given in Table 5. Using all augmentations gave the best results. We observed a drop of 1.1 and 0.5 pp mAP without text and audio augmentations, respectively. This might indicate that the text augmentations have a larger impact than the audio augmentations, which might be caused by the large difference in trainable parameters between the sentence and audio embedding models.

To isolate the effect of each individual augmentation method, we further re-trained the system (with AudioSet pretraining and augmentations) in five variants, always leaving out one of the augmentation methods. The results are summarized in Table 5. The text augmentations have the largest impact, which is in line with the previous results: Leaving out EDA and BT reduced the mAP by 1.0 and 0.7 pp., respectively. Eliminating SpecAugment and Freq-MixStyle reduced the performance by 0.7 and 0.6 pp., respectively. Gain augmentation seems to have the least impact: eliminating it reduced the mAP by only 0.2pp.

	R@1	R@5	R@10	mAP@10
SMBO	14.50	37.24	51.04	24.27 ± 0.19
no audio aug	13.88	36.94	51.06	23.74 ± 0.16
no text aug	13.12	35.77	49.25	22.91 ± 0.08
no SpeAugment	13.50	36.60	50.91	23.53 ± 0.20
no FreqMixStyle	13.61	36.91	50.69	23.62 ± 0.14
no Gain Augment	14.84	37.81	50.95	24.05 ± 0.26
no BT	13.61	36.38	50.00	23.43 ± 0.18
no EDA	13.33	37.02	49.94	23.27 ± 0.03

Table 5: Results of the ablation study on data augmentation.

8. CONCLUSION

This study set out to investigate transfer learning and data augmentation strategies to alleviate the data scarcity problem in natural-language-based audio retrieval. Our research has shown that using pretrained audio and text embedding models greatly increases the retrieval performance on ClothoV2. We enriched this already well-performing retrieval system with a range of augmentation methods and showed that augmenting both text and audio inputs significantly reduces overfitting. Finally, we further found that pretraining on AudioCaps only leads to non-significant improvements.

9. ACKNOWLEDGMENT

The LIT AI Lab is financed by the Federal State of Upper Austria.

10. REFERENCES

[1] A. Oncescu, A. S. Koepke, J. F. Henriques, Z. Akata, and S. Albanie, "Audio retrieval with natural language queries," in *22nd Annual Conf. of the Int. Speech Communication Association, Interspeech*, 2021.

[2] H. Xie, O. Räsänen, K. Drossos, and T. Virtanen, "Unsupervised audio-caption aligning learns correspondences between individual sound events and textual phrases," in *Proc. IEEE Int. Conf. Acoustic., Speech and Signal Process., ICASSP*, 2022.

[3] B. Elizalde, S. Deshmukh, M. A. Ismail, and H. Wang, "CLAP: learning audio concepts from natural language supervision," *CoRR*, vol. abs/2206.04769, 2022.

[4] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, "Learning transferable visual models from natural language supervision," in *Proc. of the 38nd Int. Conf. on Machine Learning, ICML*, 2021.

[5] X. Huang, S. Lipping, and T. n. Virtanen, "DCASE 2022 Challenge Task 6b: Language-Based Audio Retrieval," *CoRR*, vol. abs/2206.06108, 2022.

[6] D. Turnbull, L. Barrington, D. A. Torres, and G. R. Lanckriet, "Semantic annotation and retrieval of music and sound effects," *IEEE Trans. Speech Audio Process.*, 2008.

[7] Q. Kong, Y. Cao, T. Iqbal, Y. Wang, W. Wang, and M. D. Plumbley, "PANNs: Large-scale pretrained audio neural networks for audio pattern recognition," *IEEE ACM Trans. Audio Speech Lang. Process.*, 2020.

[8] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," in *Proc. of the North American Ch. of the Ass. for Computational Linguistics: Human Language Technologies, NAACL-HLT*, 2019.

[9] D. S. Park, W. Chan, Y. Zhang, C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "SpecAugment: A simple data augmentation method for automatic speech recognition," in *20th Annual Conf. of the Int. Speech Communication Association, Interspeech*, 2019.

[10] T. Chen, S. Kornblith, M. Norouzi, and G. E. Hinton, "A simple framework for contrastive learning of visual representations," in *Proc. of the 37nd Int. Conf. on Machine Learning, ICML*, 2020.

[11] K. Zhou, Y. Yang, Y. Qiao, and T. Xiang, "Domain generalization with mixstyle," in *9th Int. Conf. on Learning Representations, ICLR*, 2021.

[12] R. Sennrich, B. Haddow, and A. Birch, "Improving neural machine translation models with monolingual data," in *Proc. of the 54th Annual Meeting of the Association for Computational Linguistics, ACL*, 2016.

[13] J. W. Wei and K. Zou, "EDA: easy data augmentation techniques for boosting performance on text classification tasks," in *Proc. of the 2019 Conf. on Empirical Methods in Natural Language Processing and the 9th Int. Joint Conf. on Natural Language Processing, EMNLP-IJCNLP*, K. Inui, J. Jiang, V. Ng, and X. Wan, Eds., 2019.

[14] G. A. Miller, "Wordnet: A lexical database for english," *Commun. ACM*, 1995.

[15] K. Drossos, S. Lipping, and T. Virtanen, "Clotho: an Audio Captioning Dataset," in *Proc. IEEE Int. Conf. Acoustic., Speech and Signal Process., ICASSP*, 2020.

[16] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. of the 32nd Int. Conf. on Machine Learning, ICML*, 2015.

[17] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, J. Klingner, A. Shah, M. Johnson, X. Liu, L. Kaiser, S. Gouws, Y. Kato, T. Kudo, H. Kazawa, K. Stevens, G. Kurian, N. Patil, W. Wang, C. Young, J. Smith, J. Riesa, A. Rudnick, O. Vinyals, G. Corrado, M. Hughes, and J. Dean, "Google's neural machine translation system: Bridging the gap between human and machine translation," *CoRR*, vol. abs/1609.08144, 2016.

[18] J. F. Gemmeke, D. P. W. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio Set: An ontology and human-labeled dataset for audio events," in *Proc. IEEE Int. Conf. Acoustic., Speech and Signal Process., ICASSP*, 2017.

[19] Y. Zhu, R. Kiros, R. S. Zemel, R. Salakhutdinov, R. Urtasun, A. Torralba, and S. Fidler, "Aligning books and movies: Towards story-like visual explanations by watching movies and reading books," in *IEEE Int. Conf. on Computer Vision, ICCV*, 2015.

[20] S. Merity, C. Xiong, J. Bradbury, and R. Socher, "Pointer sentinel mixture models," in *5th Int. Conf. on Learning Representations, ICLR*, 2017.

[21] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd Int. Conf. on Learning Representations, ICLR*, 2015.

[22] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Z. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "PyTorch: An imperative style, high-performance deep learning library," in *Annual Conf. on Neural Information Processing Systems, NEURIPS*, 2019.

[23] J. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl, "Algorithms for hyper-parameter optimization," in *Annual Conf. on Neural Information Processing Systems, NEURIPS*, 2011.

[24] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," in *1st Int. Conf. on Learning Representations, ICLR*, 2013.

[25] C. D. Kim, B. Kim, H. Lee, and G. Kim, "AudioCaps: Generating captions for audios in the wild," in *Proc. of the North American Ch. of the Ass. for Computational Linguistics: Human Language Technologies, NAACL-HLT*, 2019.

DESCRIPTION AND ANALYSIS OF NOVELTIES INTRODUCED IN DCASE TASK 4 2022 ON THE BASELINE SYSTEM

Francesca Ronchini[1*], *Samuele Cornell*[2*], *Romain Serizel*[1], *Nicolas Turpault*[1], *Eduardo Fonseca*[3], *Daniel P. W. Ellis*[3]

[1] Universite de Lorraine, CNRS, Inria, Loria, Nancy, France
[2]Department of Information Engineering, Universita Politecnica delle Marche, Italy
[3] Google Research, United States

ABSTRACT

The aim of the Detection and Classification of Acoustic Scenes and Events Challenge Task 4 is to evaluate systems for the detection of sound events in domestic environments using an heterogeneous dataset. The systems need to be able to correctly detect the sound events present in a recorded audio clip, as well as localize the events in time. This year's task is a follow-up of DCASE 2021 Task 4, with some important novelties. The goal of this paper is to describe and motivate these new additions, and report an analysis of their impact on the baseline system. We introduced three main novelties: the use of external datasets, including recently released strongly annotated clips from Audioset, the possibility of leveraging pre-trained models, and a new energy consumption metric to raise awareness about the ecological impact of training sound events detectors. The results on the baseline system show that leveraging open-source pre-trained on AudioSet improves the results significantly in terms of event classification but not in terms of event segmentation.

Index Terms— Sound event detection, synthetic soundscapes, deep learning, external data, energy consumption

1. INTRODUCTION

Machine Listening has a substantial impact on applications such as noise monitoring in smart cities [1, 2], smart homes and home security solutions [3, 4], health monitoring systems [5], bio-acoustics [6], and hearing aids [7], among others. Sound Event Detection (SED) is one particularly important task in the field of Machine Listening. Its goal is to correctly output the class of different sound events present in an audio clip, together with each sound event's time boundaries [8]. Multiple events can be active simultaneously in each audio recording, but target sound events can also overlap with other non-target sound events. This research area is in continual development, attracting an expanding community.

Since 2018, the Detection and Classification Acoustic Scene and Events (DCASE) Challenge Task 4 evaluates systems for detection of sound events in audio clips recorded in domestic environments. The challenge's main goal is to explore to what extent it is possible to exploit an unbalanced and heterogeneous dataset, e.g. containing a limited subset of weakly labeled and strongly labeled data and a larger amount of unlabeled data. This is especially important as strongly labeled data is expensive and time-consuming to obtain while weakly labeled and in particular unlabeled data are much more accessible and scalable. Moreover, unlabeled data and weakly labeled data raise significantly fewer privacy concerns. In fact, weak labels could be obtained in an automated manner and, as such, human annotation is not required. This allows using on-device federated techniques which are more privacy-friendly.

Unlabeled data is usually leveraged via self-supervised learning. Indeed, past iterations of DCASE Challenge Task 4 have mainly explored this path, with most of the participants employing the mean-teacher technique [9] for this purpose. However, another possible direction is leveraging pre-trained models from related tasks such as Sound Event Classification (SEC) to obtain a SED system, e.g. via pseudo-labeling, fine-tuning or by using the pre-trained models' internal activations as additional high-level features. In the last couple of years several SEC models [10, 11] trained on the large AudioSet [12] have been open-sourced, making this approach particularly appealing. The use of such pre-trained models has been recently boosted by advancements in self-supervised learning, which enables the training in an unsupervised way on massive amounts of unlabeled data. These models can then be fine-tuned for multiple downstream tasks. A shining example in the audio domain is Wav2Vec 2.0 [13]. While powerful, these models are usually expensive to train and run (in particular self-attention based ones such as Wav2Vec 2.0 [13]), and unsuitable for widespread deployment on on-edge devices. On this premise, this year we introduced the following novelties for the DCASE Challenge Task 4:

- we allowed the use of external datasets and embeddings extracted from open-source pre-trained models. Participants were encouraged to propose open-source models to use and external data sources, such as AudioSet [12], which might also include real-word strongly annotations [14].
- we developed a new codebase[1] and new baseline models to encourage participants to explore pre-trained models and such new datasets. We present the results of such baselines in Section 5.
- finally, we introduced a new energy consumption metric based on CodeCarbon toolkit [15]. This direction aims to foster interest among participants in finding new solutions for effective but also efficient SED systems. This was mainly a pilot experiment as the reliability of CodeCarbon across various computational platforms has not been fully investigated. We explain in detail the new metric in Section 3.2, together with the challenges that needs to be addressed in order to meaningfully compare the energy consumption of different models on different hardware

[*] denotes equal contribution.

[1]Available at `github.com/DCASE-REPO/DESED_task/blob/master/recipes/dcase2022_task4_baseline`

platforms.

The goal of this paper is to describe and motivate these new directions of this year's challenge, and to analyse their impact on the baseline performance.

2. DCASE 2022 CHALLENGE TASK 4: NOVELTIES AND MOTIVATIONS

In every edition, the DCASE Challenge Task 4 proposes to focus on new research questions, targeting specific aspects of the SED task that are considered interesting and timely for the research community, with the ultimate goal of advancing the state-of-the-art.

This year, we investigate three main aspects that are motivated by the availability of large scale datasets related to the SED task, the recent popularity of pre-trained generic audio representation and the growing concerns regarding the environmental impact of our digital life. These developments were proposed in order to target three key scientific questions related to SED systems that we believe are worth addressing.

2.1. What is the impact of using external data and pre-trained models on SED systems?

In the DCASE 2021 Challenge Task 4 [16], we found that using a separation model trained in an unsupervised way using MixIT [17] on the massive YFC100m dataset [18] yielded a significant performance boost when used in conjunction with the baseline SED model on the evaluation set even if results on the development set were not that promising. Motivated by such result, this year we allowed participants to also use external data for the purpose of improving SED performance. To encourage participants to explore this path, we also provided a baseline which employs embeddings from two state-of-the-art popular models trained on AudioSet [12]: PANNs [10] and AST [11]. In addition to being allowed to use AudioSet and these two models, participants were granted the use of other pre-trained models such as YAMNet [19] and also datasets not strictly related to SED such as MUSAN [20] and ImageNet (used in AST [11] for example). Participants were also welcomed to propose other pre-trained models and external datasets. The full list of external resources allowed for the task can be found on the task website [2] and in [21, 22].

Each team was allowed to submit four different systems. However, in order to highlight the impact of external resources on SED systems, we required each team to submit at least one system that was not using external data.

2.2. Is strongly annotated real-world data necessary to build an effective SED system?

In Hershey et al. [14] it was found that strong labels (with temporally precise onset and offset together with sound-event class label) can bring substantial benefits in SEC applications even when they are provided for a small fraction of the total data, which can remain weakly labeled (only the sound class label is provided without any temporal precision). This is especially remarkable since manual annotating data is costly and time-consuming, but also bias-prone due to human errors and disagreement on the perception of some sound event onsets and offsets. On the other hand, synthetic data is cheap to obtain, but also inherently mismatched with respect to real-world data, leading to potentially sub-optimal performance [23]. In an attempt to mitigate this mismatch, and make the synthetic audio more realistic, in the past edition, non-target events have been included in the synthetic split of the training dataset [24]. This year, among the allowable external data source, we also included 3470 strongly labeled recorded audio clips coming from AudioSet [14]. The goal is to assess if the substantial improvement observed for SEC in Hershey et al. [14] also translates to SED and to what extent it may be worth spending extra resources for more manual annotation.

2.3. What is the environmental footprint of our SED systems?

Current state-of-the-art SED systems heavily rely on deep learning. Numerous recent works [25, 26] have raised concerns about the massive environmental costs of training deep learning models with large amount of parameters on massive amounts of data. For example, focusing on the audio domain, in Parcollet et al. [26], a study on the carbon footprint for ASR training was performed using the CodeCarbon toolkit [3], a software package that estimates the amount of energy consumption and carbon dioxide produced by the cloud or personal computing resources used to execute the code. One of the key takeaways from this work is the tremendous inefficiency of many current ASR state-of-the-art models that trade off significant energy consumption (and thus pollution) for a marginal increase in performance that is likely not significant in actual deployment scenarios. These results raise important questions regarding the direction of ASR research, and suggests that the blind pursuit of the best possible performance in spite of the energy efficiency is likely not worth from a practical standpoint.

Motivated by this study, this year we added support in the DCASE Challenge 2022 Task 4 baseline and repository for energy consumption benchmarking based on CodeCarbon. Participants were encouraged to submit their CodeCarbon estimated kilowatt-hour kWh energy consumption figures for each submission, both for model training and, contrary to Parcollet et al. [26], also inference on evaluation data. In fact, we argue that the energy consumption in inference is more important than the training one as the model could end up being deployed on thousands of devices and ran for years, with cumulative energy consumption quickly outpacing training phase. We describe more in detail the new CodeCarbon energy consumption metric and report energy consumption figures for this year baselines in Section 3.2.

3. DCASE 2022 CHALLENGE TASK 4 BASELINE SYSTEM

This year's challenge baseline is based on a convolutional recurrent neural network (CRNN) but includes the main novelty of having the possibility of using features extracted from pre-trained models. CRNN was already found to be the architecture, mainly taken from [27], is composed of a CNN module followed by a 2-layers bi-directional gated recurrent unit (biGRU). The CNN has 7 layers, each composed of batch normalization, gated linear unit and dropout. Input features are Log-Mel Filterbank Energies extracted with a 128 ms window and 16 ms stride. The model is trained with the mean-teacher strategy [9, 27] on audio data resampled at 16 kHz and outputs one frame-wise prediction each 64 ms. To leverage more effectively weakly and unlabeled labeled data, attention pooling is employed, as outlined in [27], to derive clip-level predictions

[2] dcase.community/challenge2022/task-sound-event-detection-in-domestic-environments#external-data-resources

[3] https://codecarbon.io/

from frame-level predictions. From 2020, small changes, such as MixUp [28], and hyper-parameters improvements have been implemented on the baseline system, based on top-ranked systems submitted every years from participants.

3.1. Integration with Pre-Trained Models

As mentioned in Section 2.1, among the main novelties in this year challenge is the allowance of pre-trained models and external data. Participants were encouraged to explore this direction by an expanded codebase in the official challenge repository. We implemented support for two different pre-trained models: PANNs and AST, which we use to extract embeddings to aid in the SED task. In detail, these embeddings are late-fused into the CRNN classifier described previously, before the biGRU module, by combining them with the features extracted by the CNN module. During training the pre-trained model is kept frozen, and it is used only as an embeddings extractor. The CRNN SED classifier is instead trained with the mean-teacher strategy as outlined before.

We consider two types of embeddings extracted from pre-trained models internal activations: global embeddings, which are extracted at clip-level $\in \mathbb{R}^C$, and frame-wise embeddings, which instead are frame-wise $\in \mathbb{R}^{S_e \times C}$, where C is the dimension of each vector and S_e is the length of the embeddings sequence.

Regarding PANNs, global embeddings are extracted after the mean and max pooling layers, following the original work [10]. The frame-wise embeddings instead are extracted from the third convolutional block, after dropout. Regarding AST, also in this case we follow the original work for global embeddings [11] and derive them from the layer before the classification head. Frame-wise embeddings instead come from the last transformer layer.

As illustrated in Figure 1, left panel, global embeddings are simply fused with CNN features $\in \mathbb{R}^{S_c \times D}$ via concatenation at each time-step $s \in [1, \dots S_c]$. In this work $D = 128$. We firstly map the embeddings to same dimensionality as the CNN features D and apply layer-normalization. Then these are concatenated with the CNN features at each time-step along the channel dimension, obtaining a tensor $\mathbb{R}^{S_c \times 2D}$, another dense layer plus layer-normalization is used to shrink back the representation to $\mathbb{R}^{S_c \times D}$ before the biGRU. For frame-wise embeddings they cannot be concatenated right away as usually $S_e \neq S_c$, since PANNs, AST and the CNN front-end have different pooling factors. Thus we employ a single-layer BiGRU to encode the frame-wise embeddings into a representation with fixed dimensionality. As illustrated in Figure 1, right panel, we use the biGRU encoder last step output $\in \mathbb{R}^H$, where H is 1024 channels, twice the hidden size of the biGRU encoder, here we use 512 neurons. This encoded representation is then fed to the same pipeline employed for global embeddings.

3.2. CodeCarbon metric

As mentioned in Section 2.1, this year we proposed to optionally report the energy consumption in kWh at training and test time. The goal here is to raise awareness regarding the environmental footprint of machine listening systems and SED systems in particular. Since every participant uses a different hardware platform to perform training and inference, absolute energy consumption figures are not directly comparable and cannot be used to assess each system efficiency. To allow for a more fair and meaningful comparison we asked for the participants to also provide, for the dev-test and eval datasets, the energy consumption obtained when inference is performed with the CRNN baseline system. The CRNN baseline kWh can then be used effectively as a common measure, mitigating the factors of variations from hardware difference. As such, in order to analyze the systems performance in terms of SED together with their energy consumption, we used a tentative, trivial energy weighted polyphonic sound detection score (EW-PSDS):

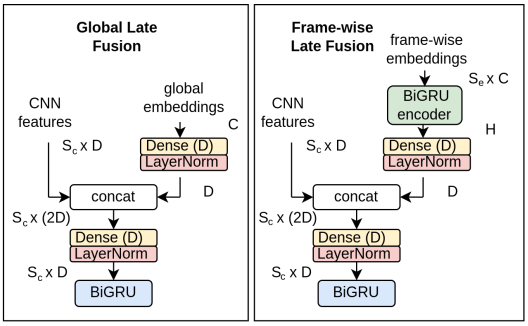


Figure 1: Combining pre-trained models embeddings with the CRNN Baseline via late-fusion before the biGRU module. Left: fusion with global embeddings. Right: fusion with frame-wise embeddings. We report the tensor dimensions as *sequenceLength* $\times$ *Channels*.

$$\text{EW-PSDS} = \text{PSDS} * \frac{\text{kWh}_{\text{baseline}}}{\text{kWh}_{\text{submission}}} \quad (1)$$

where PSDS is the polyphonic sound event detection scores [29], $\text{kWh}_{\text{baseline}}$ is the energy consumption reported for the baseline, and $\text{kWh}_{\text{submission}}$ is the energy consumption of the submitted system. Since providing energy consumption was not mandatory for participants, this initial experiment aims to provide insights to design more reliable protocols to obtain energy consumption report from challenge participants and more reliable metrics to report SED performance and energy consumption altogether. In particular, the proposed metric is very challenge-centric as it systematically relies on the energy consumption of the baseline as reference. Additionally, early results have shown that it is heavily biased by large energy consumption differences.

4. CHALLENGE DATASETS AND EVALUATION METRICS

The dataset considered on this paper is the DESED dataset[4] [30, 31], which is the same as provided for the DCASE 2021 Challenge Task 4. It is composed of 10 seconds length audio clips either recorded in a domestic environment or synthesized to reproduce such an environment[5]. The synthetic part of the dataset is generated with Scaper [32], a Python library for soundscape synthesis and augmentation.

The foreground events (both target and non-target) are obtained from the Freesound Dataset (FSD50k) [33], while the background sounds are obtained from the SINS dataset (activity class "other") [34] and TUT scenes 2016 development dataset [35]. The event co-occurences are computed on a set of strong annotations from Audioset [14]. More information regarding the generation of the DESED dataset can be found in Ronchini et al. [24].

[4] https://project.inria.fr/desed/

[5] For a detailed description of the DESED dataset and how it is generated the reader is referred to the original DESED article [31] and DCASE 2021 task 4 webpage: `http://dcase.community/challenge2021`

	PSDS-1 ↑		PSDS-2 ↑	
	dev-test	eval	dev-test	eval
Baseline	0.336	0.315	0.536	0.543
w. AudioSet strong	0.351	**0.345**	0.552	0.540
w. AST-frame	0.313	0.290	**0.722**	**0.678**
w. AST-global	0.205	0.192	0.369	0.305
w. PANNs-frame	0.354	0.304	0.635	0.597
w. PANNs-global	**0.375**	0.308	0.668	0.584

Table 1: Results for the baseline system with additional external data or pre-trained models embeddings. We report PSDS for two application scenarios as described in Section 4.

	kWh ↓		EW-PSDS-1 ↑		EW-PSDS-2 ↑	
	Dev-test	Eval	Dev-test	Eval	Dev-test	Eval
Baseline	**0.030**	**0.617**	**0.336**	**0.315**	**0.536**	**0.543**
w. AST-frame	0.061	0.901	0.149	0.198	0.344	0.464
w. AST-global	0.063	0.873	0.097	0.136	0.181	0.215
w. PANNs-frame	0.045	0.713	0.236	0.263	0.423	0.516
w. PANNs-global	0.045	0.724	0.262	0.285	0.445	0.497

Table 2: Energy consumption (kWh) and Energy Weighted PSDS scores obtained on an Nvidia A100 GPU during inference on dev-test and evaluation.

As evaluation metrics we mainly use polyphonic sound event detection scores (PSDS) [29] and consider two different applications scenarios. The first scenario targets the need of the systems to accurately detect the onset and offset of the sound event, while the second scenario penalizes more confusion between classes, but the temporal localization is less crucial [23].

5. IMPACT OF 2022 CHALLENGE NOVELTIES ON BASELINE SYSTEM

Tables 1 and 2 report the results of this study. In Table 1 we study the impact of pre-trained models and additional strongly labeled data on the challenge baseline SED system described in Section 3. Firstly, we can observe that the addition of strongly-labeled data from Audioset, as expected, improves the PSDS-1 but has practically no effect on PSDS-2. Instead, the use of embeddings from pre-trained models seems to have the opposite trend, it bring benefits mainly to PSDS-2, with AST-frame and PANNs-global coming on top. This is intuitive as PANNs and AST are trained to perform SEC. AST-global seems to perform very poorly. This is explained by the fact that the embeddings in this model are taken directly before the final classification output projection and thus the representation may be too much biased towards SEC. On the other hand, as far as PANNs embeddings are concerned, global embeddings offer the best performance overall but the two are very close. Note that differently from AST, the global embeddings in PANNs are not taken from the layer just before the final linear classification projection but rather after the mean pooling layer.

The takeaway from these experiments is that downstream performance can heavily depend on the choice of the layer at which the pre-trained model embeddings are extracted. Instead of considering just one layer, future works could consider fusing features from many different layers, for example via self-attention.

In Table 2 we compare the energy consumption of the different architectural variations of the baseline system: PANNs versus AST and frame versus global embeddings. In detail we report kWh using CodeCarbon as explained in Section 3.2 on dev-test and evaluation. We also report the relative EW-PSDS scores so that we can take into account each model efficiency. We can notice that AST models consistently require more energy in inference than PANNs. This is in accordance with the number of parameters 80M for PANNs and 88.1M for AST, and with the fact that AST are based on self-attention which is more computationally expensive than convolution. Secondly we can notice that there is little to no difference between frame and global embeddings. This is unexpected because we know that the baseline with frame-wise embeddings should be slightly more computational demanding due to the biGRU encoder as illustrated in Figure 1. This is likely due to the fact that the pre-trained model embedding extraction likely lead the energy consumption figure and CodeCarbon is simply not sensitive enough to pick up this subtle architectural difference. A more fair measure would be floating points operations (FLOP) for each frame prediction in output, however this has also the drawback that it is not easy to derive in a scalable way for multiple different models as current open source toolkits lack support for many operations.

Regarding the energy-weighted PSDS measures we can see that overall the most efficient system seems to be the "plain" baseline. This is because the pre-trained models used are much more computational intensive than the baseline, with one order of magnitude more parameters. More sophisticated techniques such as distillation could however mitigate this and drive down such energy consumption in inference. On the other hand, the EW-PSDS definition in Section 3.2 may in fact be too aggressive and too penalizing regarding energy consumption. Nevertheless indeed the simple CRNN baseline still performs fairly good, it is remarkable that, for example in Table 1, the best results for PSDS-1 are obtained without any additional pretrained model.

6. CONCLUSION

In this paper we presented the developments introduced in this year's DCASE Challenge Task 4 and analyzed their impact on the baseline performance. We focused on the possibility to use external datasets, in particular AudioSet with strong annotations, embeddings extracted with pre-trained models and on monitoring the energy consumption of the different systems at inference time. We have shown that using recorded (in-domain) clips from AudioSet with strong annotations together with synthetic soundscapes improves the PSDS-1 performance which focuses on accurately localizing the sound events in time. Using embeddings extracted from pre-trained SEC models (like PANNs or AST) improves the performance in terms of PSDS-2 which focuses on the accurate estimation on the sound event class (with loose constraints on the time localization). Yet, there are still some open questions regarding the possibility to efficiently exploit these external resources. Indeed, the baseline remains competitive with the systems using external data or pre-trained models while being substantially simpler. This aspect also reflect in terms of energy consumption of the different systems. In particular, the EW-PSDS of the models with pre-trained models is consistently lower than that of the baseline. Finally, the way monitoring energy consumption was introduced here remains naive and should be consolidated but the preliminary results open the way to considering SED systems under another angle.

7. REFERENCES

[1] G. Ciaburro and G. Iannace, "Improving smart cities safety using sound events detection based on deep neural network algorithms," in *Informatics*, 2020.

[2] P. Zinemanas, P. Cancela, and M. Rocamora, "Mavd: A dataset for sound event detection in urban environments." *DCASE Workshop*, 2019.

[3] R. Serizel, N. Turpault, H. Eghbal-Zadeh, and A. P. Shah, "Large-scale weakly labeled semi-supervised sound event detection in domestic environments," *arXiv preprint arXiv:1807.10501*, 2018.

[4] J. Ebbers and R. Haeb-Umbach, "Self-trained audio tagging and sound event detection in domestic environments," in *DCASE Workshop*, 2021.

[5] R. M. Alsina-Pagès, J. Navarro, F. Alías, and M. Hervás, "homesound: Real-time audio event detection based on high performance computing for behaviour and surveillance remote monitoring," *Sensors*, 2017.

[6] V. Lostanlen, J. Salamon, and A. e. a. Farnsworth, "Robust sound event detection in bioacoustic sensor networks," *PloS one*, 2019.

[7] A. Hüwel, K. Adiloğlu, and J.-H. Bach, "Hearing aid research data set for acoustic environment recognition," in *Proc. of ICASSP*, 2020.

[8] A. Mesaros, T. Heittola, T. Virtanen, and M. D. Plumbley, "Sound event detection: A tutorial," *IEEE Signal Processing Magazine*, 2021.

[9] A. Tarvainen and H. Valpola, "Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results," in *Advances in neural information processing systems*, 2017.

[10] Q. Kong, Y. Cao, T. Iqbal, and Y. e. a. Wang, "Panns: Large-scale pretrained audio neural networks for audio pattern recognition," *IEEE TASLP*, 2020.

[11] Y. Gong, Y.-A. Chung, and J. Glass, "Ast: Audio spectrogram transformer," *arXiv preprint arXiv:2104.01778*, 2021.

[12] J. F. Gemmeke, D. P. W. Ellis, and D. e. a. Freedman, "Audio set: An ontology and human-labeled dataset for audio events," in *Proc. of ICASSP*, 2017.

[13] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," *Advances in Neural Information Processing Systems*, 2020.

[14] S. Hershey, D. P. Ellis, and E. e. a. Fonseca, "The benefit of temporally-strong labels in audio event classification," in *Proc. of ICASSP*, 2021.

[15] V. Schmidt, K. Goyal, and A. J. et al., "CodeCarbon: Estimate and Track Carbon Emissions from Machine Learning Computing," 2021.

[16] F. Ronchini, S. Cornell, and N. e. a. Turpault, "DCASE 2021 Task 4 Challenge," https://dcase.community/challenge2021, 2021.

[17] S. Wisdom, E. Tzinis, and H. e. a. Erdogan, "Unsupervised sound separation using mixture invariant training," *Advances in Neural Information Processing Systems*, 2020.

[18] B. Thomee, D. A. Shamma, and G. e. a. Friedland, "Yfcc100m: The new data in multimedia research," *Communications of the ACM*, 2016.

[19] S. Hershey, S. Chaudhuri, and D. P. e. a. Ellis, "Cnn architectures for large-scale audio classification," in *Proc. of ICASSP*, 2017.

[20] D. Snyder, G. Chen, and D. Povey, "Musan: A music, speech, and noise corpus," *arXiv preprint arXiv:1510.08484*, 2015.

[21] N. Turpault, R. Serizel, A. Parag Shah, and J. Salamon, "Sound event detection in domestic environments with weakly labeled data and soundscape synthesis," in *DCASE Workshop*, 2019.

[22] R. Serizel, N. Turpault, A. Shah, and J. Salamon, "Sound event detection in synthetic domestic environments," in *Proc. of ICASSP* , 2020.

[23] F. Ronchini and R. Serizel, "A benchmark of state-of-the-art sound event detection systems evaluated on synthetic soundscapes," in *Proc. of ICASSP*, 2022.

[24] F. Ronchini, R. Serizel, N. Turpault, and S. Cornell, "The impact of non-target events in synthetic soundscapes for sound event detection," *arXiv preprint arXiv:2109.14061*, 2021.

[25] P. Henderson, J. Hu, and J. a. a. Romoff, "Towards the systematic reporting of the energy and carbon footprints of machine learning," *Journal of Machine Learning Research*, 2020.

[26] T. Parcollet and M. Ravanelli, "The energy and carbon footprint of training end-to-end speech recognizers," 2021.

[27] L. JiaKai, "Mean teacher convolution system for dcase 2018 task 4," DCASE2018 Challenge, Tech. Rep., 2018.

[28] H. Zhang, M. Cisse, and Y. N. e. a. Dauphin, "mixup: Beyond empirical risk minimization," *arXiv preprint arXiv:1710.09412*, 2017.

[29] Ç. Bilen, G. Ferroni, and F. e. a. Tuveri, "A framework for the robust evaluation of sound event detection," in *Proc. of ICASSP*, 2020.

[30] R. Serizel, N. Turpault, A. Shah, and J. Salamon, "Sound event detection in synthetic domestic environments," in *Proc. of ICASSP*, 2020.

[31] N. Turpault, R. Serizel, A. Parag Shah, and J. Salamon, "Sound event detection in domestic environments with weakly labeled data and soundscape synthesis," in *Detection and Classification of Acoustic Scenes and Events, Workshop, DCASE*, 2019.

[32] J. Salamon, D. MacConnell, M. Cartwright, P. Li, and J. P. Bello, "Scaper: A library for soundscape synthesis and augmentation," in *Proc. of WASPAA*, 2017.

[33] E. Fonseca, X. Favory, J. Pons, and e. a. Font, "Fsd50k: an open dataset of human-labeled sound events," *arXiv preprint arXiv:2010.00475*, 2020.

[34] G. Dekkers, S. Lauwereins, and T. et al., "The sins database for detection of daily activities in a home environment using an acoustic sensor network," in *DCASE Workshop*, 2017.

[35] A. Mesaros, T. Heittola, and T. Virtanen, "Tut database for acoustic scene classification and sound event detection," in *EUSIPCO*, 2016.

SOUND EVENT LOCALIZATION AND DETECTION WITH PRE-TRAINED AUDIO SPECTROGRAM TRANSFORMER AND MULTICHANNEL SEPARATION NETWORK

Robin Scheibler, Tatsuya Komatsu, Yusuke Fujita, Michael Hentschel

LINE Corporation, Tokyo, Japan

ABSTRACT

We propose a sound event localization and detection system based on a CNN-Conformer base network. Our main contribution is to evaluate the use of pre-trained elements in this system. First, a pre-trained multichannel separation network allows to separate overlapping events. Second, a fine-tuned self-supervised audio spectrogram transformer provides a priori classification of sound events in the mixture and separated channels. We propose three different architectures combining these extra features into the base network. We first train on the STARSS22 dataset extended by simulation using events from FSD50K and room impulse responses from previous challenges. To bridge the gap between the simulated dataset and the STARSS22 dataset, we fine-tune the models on the training part of the STARSS22 development dataset only before the final evaluation. Experiments reveal that both the pre-trained separation and classification models enhance the final performance, but the extent depends on the adopted network architecture.

***Index Terms*—** SELD, 3D CNN, Conformer, Audio Spectrogram Transformer, Separation

1. INTRODUCTION

Sound event localization and detection (SELD) combines both sound event detection (SED) and direction of arrival (DOA) estimation from multichannel recordings into a single task [1]. The task has been part of the DCASE challenge[1] since 2019. While both tasks are fairly well understood, their combination is made challenging by event polyphony, moving sources, imbalance in the duration of events, interfering events, and an overall training data scarcity. Two types of multichannel recordings have been made available, both derived from 32 channel recordings done with the Eigenmike rigid spherical microphone array[2]. A tetrahydral subset of four channels of the Eigenmike (MIC), and the first order ambisonics (FOA) coefficients derived from all 32 channels. Both formats have four channels and can be used separately or together in the challenge.

Due to the difficulty of the task, all neural solutions have been broadly adopted in the DCASE challenge submissions. A variety of input features have been proposed: generalized cross-correlation, inter-aural level and time differences, intensity vectors [2], per-channel energy normalization [3], SALSA-lite [4], to mention a few. For a detailed list, see [5]. SELD is a data-poor task and augmentations have been a crucial component of past winning systems. One successful strategy is to create new data by convolution of sound event samples with real [6] and simulated [7] impulse responses. To prevent the network overfitting to some directions, the symmetries of the recording system have been used to increase the diversity of DOA angles by artificially rotating the data [8]. A variety of network architectures have been proposed. Many are derived from the convolutional recurrent networks used in the challenge baselines [5]. Many recent solutions have replaced or complemented the recurrent layers by self-attention [9]. The event-independent network architecture [10] proposes to decouple the SED and DOA tasks using separate networks with soft stitching between layers. The 2022 challenge is characterized by the new STARSS22 dataset of real sound scenes played by actors [11]. Due to the high quality of the data set, only $4.9\,\mathrm{h}$ of recordings are available. One of the avowed objective of this year's challenge is to explore the use of external ressources to compensate the lack of training data.

Our submission to the DCASE Challenge Task 3 relies on two unrelated advancements in the field. First, high quality pre-trained and self-supervised audio representation have become recently widely available. As examples, see wav2vec [12], PANN [13], and the self supervised audio spectrogram transformer (SSAST) [14]. Second, multichannel source separation based on independent vector analysis (IVA) has been shown to improve sound event detection [15]. Our proposed solution uses the FOA format as it is free of spatial aliasing up to $9\,\mathrm{kHz}$. We use a multichannel separation model trained in advance to coarsely separate the input signal by directions. The separation algorithm is independent vector analysis with a neural source model [16]. Then, we use a pre-trained SSAST [14] fine-tuned on the Task 3 dataset to predict events in the FOA omni channel and the four separation output channels. However, these predictions lack spatial information so we combine them with a dedicated CNN-Conformer network. The inputs of this network are the log-mel spectrograms of the FOA channels and the intensity vector [2]. The CNN creates useful feature maps which are further processed by an eight layer conformer-encoder. We explore three different architectures introducing the SSAST predictions at different points of this network. We find that it is most effective to introduce the SSAST features both at the input and output of the conformer block. The proposed systems are illustrated in Fig. 1.

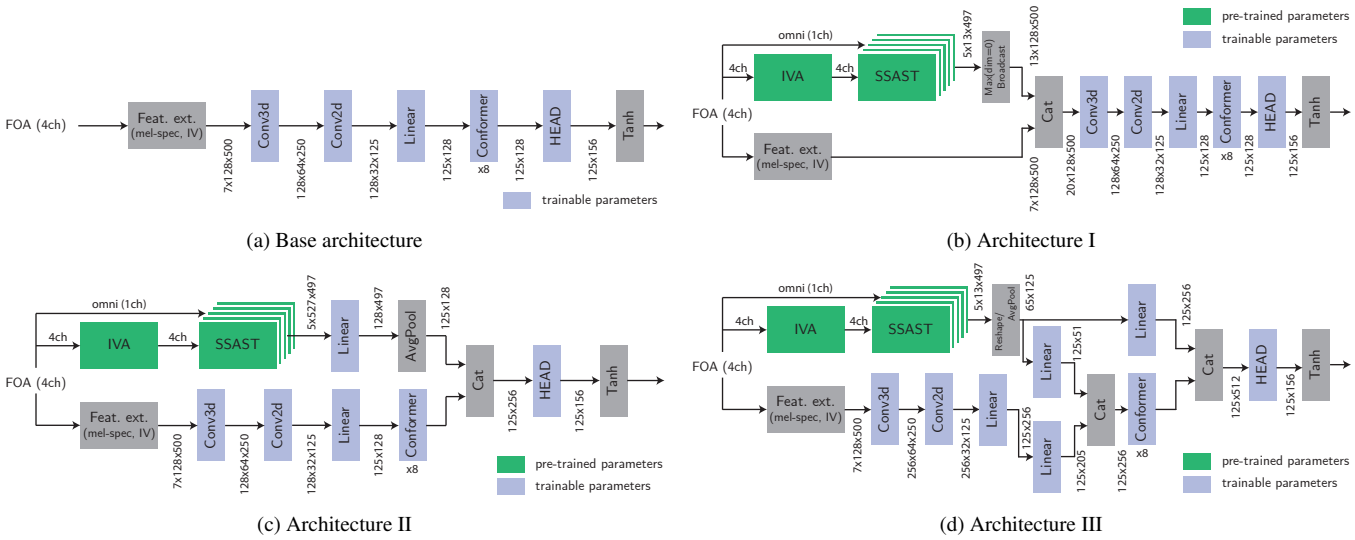
2. PROPOSED SELD NETWORK

2.1. Features

The input data to our SELD network are the four channels first order ambisonics (FOA) signals. First, to help with recognition of events, we run the FOA into a separation network that roughly separates the different events. The separation network is based on independent vector analysis [17] with a neural source model [16] described in Section 2.1.1. We obtain four tracks out of the separation network. Second, these four tracks as well as the omni channel of the FOA are run through a fine-tuned Self-Supervised Audio Spectrogram

[1] https://dcase.community

[2] https://mhacoustics.com/products



(a) Base architecture

(b) Architecture I

(c) Architecture II

(d) Architecture III

Figure 1: Structure of the proposed systems. Blue blocks have trainable parameters. Green blocks have been pre-trained. Gray blocks are not trainable. The (a) base system and the three extensions proposed (b), (c), and (d). HEAD is either a linear or MLP layer.

Transformer (SSAST) described in Section 2.1.2. Third, the log-mel-spectrograms of the four FOA channels as well as the intensity vector (IV) [2], as used in many SELD systems, provide the spatial information needed for DOA estimation. We use 128 bands for the mel-spectrogram analysis.

2.1.1. *Separation Network*

The multichannel separation network consists of a blind dereverberation part using weighted prediction error (WPE) [18], followed by independent vector analysis (IVA) [19, 20]. For WPE, the STFT uses an FFT size of 512 with ¾-overlap and a Hann window. The number of iterations, delays, and taps is 3, 3, and 10, respectively. For IVA, the STFT uses an FFT size of 2048 with ¾-overlap and a Hann window. The IVA algorithm used is iterative source steering [17] with a neural source model [16]. The number of IVA iterations is 20 and we use demixing matrix checkpointing [21] to save memory. The neural source model uses three 1D convolutional layers with GLU non-linearities and batch normalization with four groups. The hidden dimension is 128 which we map back to the STFT size by a 1D transposed convolution layer. Finally, a sigmoid non-linearity produces a mask-like signal from the network's output. A system description of the IVA separation and neural source models are shown in Fig. 2.

Since we do not have access to the ground-truth separated signals for the SELD datasets, we cannot use the conventional source separation loss functions, e.g., SI-SDR or CI-SDR. However, we have access to the direction of arrival of the events so that we can use a recently proposed spatial loss [22]. To train the network, we cut the input data into blocks of 5 s and use the median DOA as target because IVA assumes the sources to be static in this interval.

2.1.2. *Self-Supervised Audio Spectrogram Transformer*

The Self-Supervised Audio Spectrogram Transformer (SSAST) [14] is an all-attention model that has been extensively pre-trained by self-supervision on Audioset [23]. We fine-tune a pre-trained version of SSAST [24] on the STARSS22 dataset and the baseline extended dataset prepared by the organizers of Task 3. The fine tuning is done for the SED part of the task only. To this end, the DOA information is stripped from the targets and multiple events of the same class are merged together when they appear simultaneously. The SSAST model operates on 5 s blocks and produces class presence prediction vectors (13-dimensional) for each of the 497 frames (approx. 10 ms per frame).

2.2. SELD Network

Our proposed SELD system combines a base network with the extra predictions obtained from the separation network and the SSAST. The different system variants are shown in Fig. 1 and their number of parameters given in Table 1.

Base Network: The base network is a fairly conventional CNN-Conformer network for SELD using FOA features. We feed the log-mel-spectrograms of the four FOA channels and the IV channels into a convolutional network with two layers (total of 7 channels). The first is a 3D convolutional layer where the three dimensions are channels, mel-frequency bands, and time, respectively. We expect that such 3D filters can better capture the directional information present in the input signal. The kernels are of size $7 \times 3 \times 3$ and the padding is $(0, 1, 1)$, which results in a 2D output signal. Thus, the second layer is a 2D convolutional layer with 3×3 kernels. Strides of size 2 are used in the frequency and time dimension to reduce the size of the input signal. The number of channels after the 3D convolution is 128. Group normalization with four groups and ReLU activations are used after each layer. After the two strided convolutions, the remaining 32 frequency dimensions are merged with the 128 channels and projected to dimension 128 by a linear layer before the output. The output of this stage is an embedding signal with 128 dimensions and a frame interval of 40 ms This output is fed into a conformer-encoder [25] with eight layers and convolution kernel size 7. The embedding vectors so created are then projected by an output head. We explore both a simple linear projection and a two layer multi-layer perceptron (MLP). The MLP uses a GeLU non-

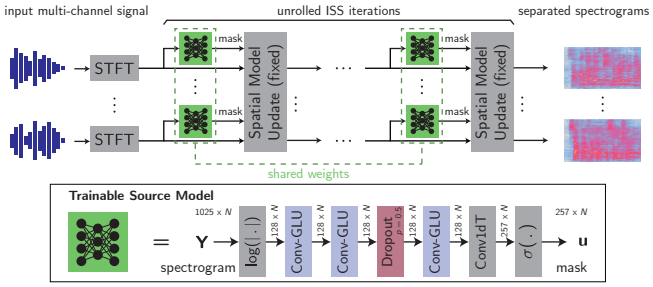


Figure 2: The structure of the separation network used to obtain the input features.

Model	SSAST	IVA	Base	I	II	III
Params.	87.2 M	2.4 M	3.8 M	3.8 M	3.9 M	15.3 M

Table 1: List of models and their number of parameters. Numbers reported include the MLP output head.

linearity. The final non-linearity is a hyperbolic tangent to limit the output to the $[-1, 1]$ range. The output is in the Multi-ACCDOA format [26] with 4 tracks, thus the output size is 4 tracks $\times$ 3 dimension of Cartesian DOA vectors $\times$ 13 classes, a total of 156 outputs per time frame. The event presence probability is given by the length of the 3D vector for each track/class slot. The base network is shown in Fig. 1a.

Architecture I: Our first variant combines all the features at the input of the CNN (Fig. 1b). The 5 channels of SSAST predictions are aggregated by taking the max. Then, they are broadcasted to match the 128 bands of the log-mel spectrogram and IV features to which they are concatenated. Thus, the input of the CNN has 20 channels and 128 mel-frequency bands. The rest of the network is similar to the base network.

Architecture II: Here we concatenate the SSAST predictions to the output of the conformer, before the output head (Fig. 1c). We project the SSAST prediction vectors of the omni FOA channel and the 4 IVA output channels (see Section 2.1.1) from 13 to 128 dimensions by a linear projection followed by ReLU activations. After this, these five channels are averaged into one. The frame rate is adjusted to that of the spatial feature extraction network by average pooling of size four along the time axis. The embedding obtained is concatenated to the output of the conformer to obtain an embedding of size 256. Finally, a linear layer projects this concatenated embedding to the output size.

Architecture III: This variant combines the SSAST predictions to both the input and the output of the conformer (Fig. 1d). The 5 channels and 13 classes dimensions are reshaped to size 65 and projected to size 51 by a linear layer. The output of the CNN is linearly projected to size 205 and concatenated to obtain an embedding of size 256 which is fed to the conformer-encoder. Another linear layer projects the 65-dimensional pre-trained features to size 256, which is concatenated to the output of the conformer layers to obtain a 512-dimensional embedding fed to the output head. This variant uses an embedding size of 256 both for the CNN and conformer (unlike 128 for all other networks), resulting in a larger network.

Name	Ref	Type	Ov.	Inter.	Train	Val.
STARSS22	[11]	Rec.	5	✓	2.9 h	2.0 h
Synth1	[27]	Sim.	2	0	20 h	—
Synth2		Sim.	4	1	20 h	—

Table 2: The datasets used. Columns "Ov." and "Inter." indicate the maximum number of overlapping event, and the number of interfering out-of-classes events. Rec. and Sim. stand for "recorded" and "simulated", respectively.

Name	Ref.	Type
STARSS22	[11]	DCASE2022 task 3 dataset
FSD50K	[28, 29]	audio dataset
TAU-SRIR DB	[6, 30]	RIR dataset
SSAST	[14, 24]	pre-trained pytorch model

Table 3: List of external datasets and models used

2.3. Post-processing

The post-processing works in two steps. Let $\boldsymbol{q}_{ntc}$ be the output of the nth frame, tth track, and cth class. The event probability is taken to be $p_{ntc} = \|\boldsymbol{q}_{ntc}\|$. First, events are detected if $p_{ntc} \geq \sigma_c$ at the output framerate of the network. We run a de-duplication procedure to remove duplicate events produced by the Multi-ACCDOA format. Events from different tracks of the same class with directions closer than θ_c, a class specific threshold, are merged together. Second, all the events from the same output frame are aggregated together. Because the output frames of the network are 40 ms and the target frames are 100 ms, there are 2 or 3 events per output frame, track, and class. For every output frame and class, we find the event with largest p_{ntc} and count all events within θ_c. If the count is larger than η_c, we declare an event with direction given by the average of all aggregated events, weighted by their probability. By default, we use $\sigma_c = 1/2$, $\theta_c = 15°$, and $\eta_c = 1$. To maximize performance, we use a post-processing calibration procedure where σ_c, θ_c, and η_c are chosen per class to minimize the SELD score on the validation dataset of STARSS22.

2.4. Differences with the Challenge Submission

Our challenge submission was based on Architecture II with a linear output head. However, after the end of the challenge, we found a mistake in our use of the pre-trained SSAST, namely, the output layers were not initialized to the correct size (527 instead of 13 dimensional vectors) and weights. Furthermore the weights used on the development and evaluation datasets were different. We have corrected and retrained all the architectures for this paper.

3. DATASET AND TRAINING

3.1. Datasets

We use the three datasets described in Table 2 with a total of 42.9 h and 2.0 h of training and validation data, respectively. From the DCASE2022 task 3 dataset, STARSS22 [11], fold3 (2.9 h) is used for training and fold4 (2.0 h) for validation, as suggested. Since this is not sufficient, we also use the baseline training synthetic dataset

(Synth1) provided by the task organizers [27]. This dataset is created by remixing sound events from the FSD50K dataset [28, 29] with the measured RIR from the TAU-SRIR database [30, 6]. However, the dataset Synth1 only contains up to two overlapping events, and no interfering events. Thus, we use the original recipe provided for Synth1 [31] to create an extended training set, Synth2. We change the recipe in the following ways. First, increase the maximum number of overlapping events from 2 to 4. Second, we add interfering sound events not included in the classification task. For the interference, we select clips from the following categories of FSD50K: `Cutlery, silverware`, `Computer, keyboard`, `Chewing, mastication`, `Buzz`, `Crumpling, crinkling`, `Typing`, `Clock`, `Meow`, `Breathing`, `Glass`, `Writing`, `Chink, clink`. The base external datasets and pre-trained models used are summarized in Table 3 and the training datasets in Table 2, respectively.

3.2. Data Augmentations

SpecAugment: We apply SpecAugment [32] using the same mask to all FOA channels prior to computation of mel-spectrogram and IV during training. The maximum time masking is $2\,\%$ of the total length, while frequency masking is up to $10\,\%$.

Random Rotations: To avoid the network over-fitting to specific directions, we apply random rotations to the FOA input, as has been successfully used for SELD networks in previous challenges [8]. By applying the same rotation to the targets, we are able to simulate large spatial variations in the input dataset. This augmentation is applied to input examples with probability ½.

3.3. Training

We train the network with the recently proposed Multi-ACCDOA loss [26]. The optimizer is Adam [33] with learning rate 0.001. We do learning rate warm-up over the first 10000 steps. The network is trained for 1000 epochs on STARSS22, Synth1, and Synth2 datasets. The progress of the optimization is monitored on the validation set of STARSS22 using the SELD score,

$$\mathrm{SELD} = 0.25\left(\mathrm{ER} + (1-\mathrm{F}) + \mathrm{LE}/180 + (1-\mathrm{LR})\right), \quad (1)$$

where ER, F, LE, LR, are the official SELD metrics [1]. After training finishes, we fine-tune the network on the training part (fold3) of STARSS22 only. We freeze all layers except the output head. Training is restarted from the average of the 10 checkpoints with lowest SELD score with learning rate 0.0001. We proceed for 1000 epochs. Finally, we select the 10 checkpoints with the lowest validation score and average their weights.

4. EXPERIMENTS

We do an ablation study to assess the contribution of the different components. Our reference is the base network with linear output head, trained without fine tuning (Fig. 1a). For each architecture we add in order SSAST predictions of FOA omni channel (+AST), and four separation outputs (+IVA), MLP output head (+MLP), fine-tuning (+FINE), and post-processing calibration (+POST).

Table 4 shows the results on the validation part of the development set of STARSS22 (fold4) compared to that of the baseline system [34]. Our first observation is that all the proposed architectures improve significantly over the baseline [34]. In addition, we see that how we insert the SSAST/IVA features into the network matters. Architecture I concatenates the extra features at the input of the network, which might make training the entire network more difficult. The MLP head is important here to obtain the best performance. For architecture II, where the features are only inserted at the end, the MLP is also required to fully take advantage of the extra information. Architecture III performs best by considering features both in the conformer, and again in the output head. There, the benefits of SSAST and IVA features are clearly visible. Of the three, it performs best, but is also the largest model. However, the better performance is not due to size only since it does not outperform I and II for +AST. In all cases, fine-tuning and post-processing calibration are necessary to maximize performance.

Model	ER↓	F↑	LE↓	LR↑	SELD ↓
Baseline (FOA) [34]					
	0.71	0.21	29.3	0.46	0.5507
Base Network					
	0.578	0.421	19.083	0.602	0.4154
+MLP	0.594	0.412	17.015	0.608	0.4174
+FINE	0.561	0.451	16.314	0.563	0.4094
+POST	0.535	0.464	**15.869**	0.562	0.3994
Architecture I					
+AST	0.575	0.423	18.752	0.591	0.4164
+IVA	0.574	0.418	17.809	0.582	0.4182
+MLP	0.584	0.455	17.331	0.606	0.4050
+FINE	0.562	0.469	16.881	0.616	0.3928
+POST	0.519	0.480	16.375	0.598	0.3830
Architecture II					
+AST	0.572	0.424	18.130	0.604	0.4111
+IVA	0.589	0.414	18.016	0.611	0.4160
+MLP	0.592	0.445	18.156	**0.641**	0.4020
+FINE	0.534	0.478	17.163	0.595	0.3891
+POST	0.516	0.497	16.551	0.603	0.3768
Architecture III					
+AST	0.579	0.417	18.785	0.607	0.4147
+IVA	0.572	0.437	17.957	0.621	0.4037
+MLP	0.567	0.460	18.294	0.616	0.3980
+FINE	0.551	0.493	17.505	0.639	0.3792
+POST	**0.500**	**0.514**	17.131	0.624	**0.3644**

Table 4: SELD metrics of proposed architectures and baseline [34].

5. CONCLUSION

We have presented three ways of using a pre-trained SSAST and separation system to improve SELD. An ablation study demonstrated the effectivness of the different components. We found that inserting the pre-trained predictions both before and after the conformer encoder, combined with an MLP outptut classification head is most effective.

6. REFERENCES

[1] A. Politis, A. Mesaros, S. Adavanne, T. Heittola, and T. Virtanen, "Overview and evaluation of sound event localization and detection in dcase 2019," *IEEE/ACM Trans. Audio Speech Lang. Process.*, vol. 29, pp. 684–698, 2020.

[2] K. Lopatka, J. Kotus, and A. Czyzewski, "Detection, classification and localization of acoustic events in the presence of background noise for acoustic surveillance of hazardous situations," *Multimedia Tools and Applications*, vol. 75, no. 17, pp. 10 407–10 439, 2016.

[3] V. Lostanlen et al., "Per-Channel Energy Normalization: Why and How," *IEEE Signal Process. Lett.*, vol. 26, no. 1, pp. 39–43, Jan. 2019.

[4] T. N. Tho Nguyen, D. L. Jones, K. N. Watcharasupat, H. Phan, and W.-S. Gan, "SALSA-Lite: A fast and effective feature for polyphonic sound event localization and detection with microphone arrays," in *Proc. IEEE ICASSP*, Singapore, SG, May 2022.

[5] S. Adavanne, A. Politis, J. Nikunen, and T. Virtanen, "Sound event localization and detection of overlapping sources using convolutional recurrent neural networks," *IEEE J. Sel. Top. Signal Process.*, vol. 13, no. 1, pp. 34–48, Apr. 2019.

[6] A. Politis, S. Adavanne, and T. Virtanen, "A dataset of reverberant spatial sound scenes with moving sources for sound event localization and detection," in *Proc. DCASE*, Tokyo, JP, Nov. 2020.

[7] Y. Koyama et al., "Spatial data augmentation with simulated room impulse responses for sound event localization and detection," in *Proc. IEEE ICASSP*, Singapore, SG, May 2022, pp. 8872–8876.

[8] F. Ronchini, D. Arteaga, and A. Pérez-López, "Sound event localization and detection based on crnn using rectangular filters and channel rotation data augmentation," in *Proc. DCASE2020*, Tokyo, JP, Nov. 2020.

[9] K. Shimada et al., "Ensemble of ACCDOA- and EINV2-based Systems with D3Nets and Impulse Response Simulation for Sound Event Localization and Detection," DCASE Challenge, Tech. Rep., June 2021.

[10] Y. Cao, T. Iqbal, Q. Kong, F. An, W. Wang, and M. D. Plumbley, "An Improved Event-Independent Network for Polyphonic Sound Event Localization and Detection," in *Proc. IEEE ICASSP*, Toronto, CA, June 2021, pp. 885–889.

[11] A. Politis et al., "STARSS22: A dataset of spatial recordings of real scenes with spatiotemporal annotations of sound events," *arXiv preprint arXiv:2206.01948*, 2022.

[12] S. Schneider, A. Baevski, R. Collobert, and M. Auli, "wav2vec: Unsupervised pre-training for speech recognition," in *Proc. Interspeech*, Graz, AU, Sept. 2019, pp. 3465–3469.

[13] Q. Kong, Y. Cao, T. Iqbal, Y. Wang, W. Wang, and M. D. Plumbley, "PANNs: Large-scale pretrained audio neural networks for audio pattern recognition," *IEEE/ACM Trans. Audio Speech Lang. Process.*, vol. 28, pp. 2880–2894, Oct. 2020.

[14] Y. Gong, C.-I. J. Lai, Y.-A. Chung, and J. Glass, "SSAST: Self-supervised audio spectrogram transformer," *arXiv preprint arXiv:2110.09784*, 2021.

[15] R. Scheibler, T. Komatsu, and M. Togami, "Multichannel separation and classification of sound events," in *Proc. EUSIPCO*, Dublin, IE, Aug. 2021, pp. 1035–1039.

[16] R. Scheibler and M. Togami, "Surrogate source model learning for determined source separation," in *Proc. IEEE ICASSP*, Toronto, CA, June 2021, pp. 176–180.

[17] R. Scheibler and N. Ono, "Fast and stable blind source separation with rank-1 updates," in *Proc. IEEE ICASSP*, Barcelona, ES, May 2020, pp. 236–240.

[18] T. Yoshioka and T. Nakatani, "Generalization of Multi-Channel Linear Prediction Methods for Blind MIMO Impulse Response Shortening," *IEEE Trans. Audio Speech Lang. Process.*, vol. 20, no. 10, pp. 2707–2720, 2012.

[19] A. Hiroe, "Solution of permutation problem in frequency domain ica, using multivariate probability density functions," in *ASIACRYPT 2016*. Springer Berlin Heidelberg, Jan. 2006, vol. 3889, pp. 601–608.

[20] T. Kim, T. Eltoft, and T.-W. Lee, "Independent vector analysis: An extension of ICA to multivariate components," in *ASIACRYPT 2016*. Springer Berlin Heidelberg, Jan. 2006, vol. 3889, pp. 165–172.

[21] K. Saijo and R. Scheibler, "Independence-based joint dereverberation and separation with neural source model," in *Proc. Interspeech*, Incheon, KR, Sept. 2022.

[22] ——, "Spatial loss for unsupervised multi-channel source separation," in *Proc. Interspeech*, Incheon, KR, Sept. 2022.

[23] J. F. Gemmeke et al., "Audio Set: An ontology and human-labeled dataset for audio events," in *Proc. IEEE ICASSP*, New Orleans, LA, USA, Mar. 2017, pp. 776–780.

[24] Y. Gong et al., "Yuangongnd/ssast." [Online]. Available: https://github.com/YuanGongND/ssast

[25] A. Gulati et al., "Conformer: Convolution-augmented transformer for speech recognition," in *Proc. Interspeech*, 2020, pp. 5036–5040.

[26] K. Shimada et al., "MULTI-ACCDOA: Localizing and detecting overlapping sounds from the same class with auxiliary duplicating permutation invariant training," in *Proc. IEEE ICASSP*, Singapore, SG, pp. 316–320.

[27] A. Politis, "[DCASE2022 Task 3] Synthetic SELD mixtures for baseline training," Apr. 2022. [Online]. Available: https://doi.org/10.5281/zenodo.6406873

[28] E. Fonseca et al., "FSD50K: an open dataset of human-labeled sound events," *IEEE/ACM Trans. Audio Speech Lang. Process.*, vol. 30, pp. 829–852, 2022.

[29] E. Fonseca, X. Favory, J. Pons, F. Font, and X. Serra, "FSD50K," Oct. 2020. [Online]. Available: https://doi.org/10.5281/zenodo.4060432

[30] A. Politis, S. Adavanne, and T. Virtanen, "TAU Spatial Room Impulse Response Database (TAU- SRIR DB)," Apr. 2022. [Online]. Available: https://doi.org/10.5281/zenodo.6408611

[31] D. Krause and A. Politis, "danielkrause/dcase2022-data-generator." [Online]. Available: https://github.com/danielkrause/DCASE2022-data-generator

[32] D. S. Park et al., "SpecAugment: A simple data augmentation method for automatic speech recognition," in *Proc. Interspeech*. ISCA, Sept. 2019, pp. 2613–2617.

[33] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. ICLR*, San Diego, CA, USA, May 2015.

[34] S. Adavanne, "sharathadavanne/seld-dcase2022." [Online]. Available: https://github.com/sharathadavanne/seld-dcase2022

KNOWLEDGE DISTILLATION FROM TRANSFORMERS FOR LOW-COMPLEXITY ACOUSTIC SCENE CLASSIFICATION

Florian Schmid[1,2]*, Shahed Masoudian*[2]*, Khaled Koutini*[2]*, Gerhard Widmer*[1,2]

[1]Institute of Computational Perception (CP-JKU),[2]LIT Artificial Intelligence Lab,
Johannes Kepler University Linz, Austria
florian.schmid@jku.at, shahed.masoudian@jku.at, khaled.koutini@jku.at

ABSTRACT

Knowledge Distillation (KD) is known for its ability to compress large models into low-complexity solutions while preserving high predictive performance. In Acoustic Scene Classification (ASC), this ability has recently been exploited successfully, as underlined by three of the top four systems in the low-complexity ASC task of the DCASE'21 challenge [1] relying on KD. Current KD solutions for ASC mainly use large-scale CNNs or specialist ensembles to derive superior teacher predictions. In this work, we use the *Audio Spectrogram Transformer* model *PaSST*, pre-trained on Audioset, as a teacher model. We show how the pre-trained *PaSST* model can be properly trained downstream on the *TAU Urban Acoustic Scenes 2022 Mobile development dataset* [2] and how to distill the knowledge into a low-complexity CNN student. We study the effect of using teacher ensembles, using teacher predictions on extended audio sequences, and using Audioset as an additional dataset for knowledge transfer. Additionally, we compare the effectiveness of Mixup and Freq-MixStyle to improve performance and enhance device generalization. The described system achieved rank 1 in the Low-complexity ASC Task of the DCASE'22 challenge [3][1].

Index Terms— Patchout Spectrogram Transformer, Mixup, Freq-MixStyle, Knowledge Distillation

1. INTRODUCTION

Acoustic Scene Classification (ASC) aims to build a model that accurately predicts a scene label given an audio recording. Convolutional Neural Networks (CNNs) are well-established models that dominate the field of ASC [1, 4–6]. Low-complexity solutions are of increasing interest, making the ASC systems runnable on edge devices. Model compression techniques to reduce complexity include *Parameter Pruning* [7–9], *designing efficient network architectures* [10, 11] and *Knowledge Distillation (KD)* [12–14]. KD recently showed promising results for ASC, with three out of the top four systems applying KD in the low-complexity ASC task of the DCASE'21 challenge [1].

This paper focuses on distilling knowledge from Patchout faSt Spectrogram Transformers (PaSST) [15] into low-complexity CNNs, the winning approach of the DCASE'22 low-complexity ASC task [3]. The *TAU Urban Acoustic Scenes 2022 Mobile development dataset* [2] and the low-complexity constraints of the DCASE'22 challenge define an ASC scenario involving various challenges:

C1 Limits on the number of parameters and the number of multiply-accumulate operations.

C2 Scenes are recorded using different recording devices, resulting in a domain shift.

C3 1-second audio snippets are provided compared to 10-second snippets in the ASC task of the DCASE'21 challenge [1]. Shorter audio files contain less information and are more challenging to classify.

The system studied in the following tackles C1 by using KD with PaSST [15] models as a teacher and a low-complexity CNN as a student. Regarding C2, the domain shift caused by different recording devices is counteracted by *Freq-MixStyle* [16], a modified version of *MixStyle* [17], operating on frequency statistics. The 1-second audio snippets described in C3 result from splitting the 10-second snippets of the *TAU Urban Acoustic Scenes 2020 Mobile development dataset* [2] into ten pieces. The provided segment identifiers allow for reassembling the complete 10-second files. We train all models on randomly cropped 1-second pieces from the reassembled 10-second files to increase the diversity in the training data. Reassembling the 10-second files additionally allows for obtaining superior teacher predictions for KD, as described later.

The related work regarding the system's building blocks is described in Section 2. Section 3 introduces student and teacher architectures and the PaSST downstream training. Section 4 describes how student and teacher are combined in a KD setup and sets the stage for the results presented in Section 5. The paper is concluded in Section 6.

2. RELATED WORK

2.1. ASC Architectures

In the past years, Convolutional Neural Networks (CNNs) became the most prominent solution to process spectrograms and dominated previous DCASE challenges [1, 4–6]. Restricting the receptive field of CNNs, known as Receptive Field Regularization, was shown to be particularly well suited for ASC tasks [18, 19].

Inspired by Vision Transformers (ViT) [20], transformers capable of processing spectrograms to solve audio tasks have been proposed recently. In this regard, Audio Spectrogram Transformers (AST) [21], pre-trained on computer vision tasks, have been adapted to the audio domain and achieved state-of-the-art results on Audioset [22]. Image [20, 23] and spectrogram [15, 21] transformers extract overlapping patches with a certain stride from the input image and add a positional encoding. Patchout faSt Spectrogram Transformer (PaSST) [15] disentangles frequency- and time-encodings, simplifying downstream training on shorter audio clips. PaSST additionally introduces *Patchout*, a mechanism that drops parts of the input sequence to improve generalization and reduce

[1]Source Code:https://github.com/CPJKU/cpjku_dcase22

memory and computational complexity in the quadratically scaling attention layers.

2.2. Knowledge Distillation in ASC

In KD [12–14], the aim is to compress the knowledge of a possibly large number of complex teacher models into a low-complexity student model without significant performance loss. The student minimizes a weighted sum of hard label loss and distillation loss, as shown in Eq. 1. The distillation loss matches student and teacher predictions based on the Kullback-Leibler divergence between soft targets: $L_{\mathrm{DIST}} = D_{\mathrm{KL}}(q_{\mathrm{teacher}}||q_{\mathrm{student}})$. By minimizing the distillation loss, the student learns to mimic the teacher.

$$L_{\mathrm{TOTAL}} = L_{\mathrm{LABEL}} + \lambda L_{\mathrm{DIST}} \tag{1}$$

The soft targets q are created from the logits z by computing $q = \mathrm{softmax}(z/T)$ with a specific temperature T. Raising the temperature T creates a softer distribution and allows the student to exploit the rich similarity information between classes predicted by the teacher compared to the hard labels [12].

In the low-complexity task of the DCASE'21 challenge [1], the top two systems included KD based on a pre-trained CNN teacher network [4] and a large two-stage fusion CNN teacher model [5].

Jung et al. [24] showed that learning from soft targets in a teacher-student setup has a beneficial effect as one-hot labels do not reflect the blurred decision boundaries between different acoustic scenes. Teacher superiority is achieved by using multiple audio segments of the same scene recorded at different locations to generate teacher predictions.

Another popular idea to generate superior teacher predictions is to use ensembles of specialist models. In this regard, ensembles of device-experts [25], ensembles of models trained on different audio representations [26], and ensembles of specialist models for confusing pairs of acoustic scenes [27] have been studied and successfully applied.

2.3. Mixup and Freq-MixStyle

Mixup [28] constructs virtual training samples by linearly mixing two existing samples and their labels. In particular, the coefficients of the convex sample and label combinations are drawn randomly from a Beta distribution, its shape specified by a parameter α. Mixup has been shown to improve generalization on ASC tasks before [29].

MixStyle [17] is introduced to enhance domain generalization by mixing channel-wise statistics of images. However, the device-style in spectrograms primarily resides in the frequency-wise statistics. To enhance generalization across recording devices, an adapted version of MixStyle, *Freq-MixStyle*, is proposed in [16]. It proceeds by normalizing each frequency band and denormalizing it with mixed frequency coefficients of two different samples. *Freq-MixStyle* is guided by two parameters: α determines the shape of the Beta distribution used to randomly draw mixing coefficients, and p specifies the probability of whether it is applied to a batch or not.

3. MODEL SPECIFICATIONS

3.1. Student Model: Compact RFR-CNN

The student model is a Receptive Field Regularized [18, 19] Convolutional Neural Network (RFR-CNN) and is based on *CP_ResNet* which performed well in previous editions of the DCASE ASC challenge [1, 2, 6, 29, 30]. The initial width is reduced to $W = 32$ channels, and a grouping of 2 is applied to the penultimate block to obtain a compact model. The max-pooling layers (indicated by P_f) are adapted to perform pooling only over the frequency dimension to account for the shorter audio clips (1-second). This allows us to downscale spectrogram dimensionality while preserving temporal information. Finally, $C = 36$ channels are cut from the final residual block of the network to conform to the low-complexity limitations. Table 1 summarizes the architecture of the CNN used as the student.

Table 1: Low-complexity Student Model realized by a compact CP-ResNet Architecture.

WIDTH	GROUPING	BLOCK	CONFIG
W		INPUT	5×5, P
W	1	R	3×3, 1×1, P_f
W	1	R	3×3, 3×3, P_f
		LINEAR	$W \rightarrow 2\dot{W}$
$2 \times W$	2	R	3×3, 3×3
		LINEAR	$2W \rightarrow 4\dot{W}$
$4 \times W - C$	1	R	3×3, 1×1
CLASSIFIER $4 \times W - C \rightarrow 10$ CLASSES			
GLOBAL MEAN POOLING			

P: 2×2 MAX POOLING.
P_f: 2×1 MAX POOLING OVER THE FREQUENCY DIMENSION.
R: RESIDUAL, THE INPUT IS ADDED TO THE OUTPUT

3.2. Teacher Model: PaSST

The main criterion for selecting teacher models is the accuracy of the predictions to reflect class similarity structures. The choice of PaSST transformer models as teachers is motivated by their high performance on downstream tasks – for instance, achieving an accuracy of 76.3% [15] on the *TAU Urban Acoustic Scenes 2020 Mobile development dataset* [2] –, and their ability to recognize fine-grained acoustic events after being pre-trained on the 527 classes of Audioset [22]. The PaSST models selected for downstream training on the *TAU Urban Acoustic Scenes 2022 Mobile development dataset* [2] are pre-trained on Audioset and extract patches of size 16x16 with a stride of 10. We use a *structured Patchout* of 6 *only* on the frequency dimension, which means that 6 frequency bands are dropped at random during training. This is an important countermeasure to prevent overfitting on the downstream dataset [3]. Due to the short length of the audio clips, we do not apply Patchout across the time dimension. The experimental setup is the same as provided in the system's technical report [31].

Table 2 compares a PaSST Baseline model with *Mixup* and *Freq-MixStyle* augmentation techniques. *Freq-MixStyle* is only applied to the raw spectrograms. While the effect on real devices is limited, *Mixup* and *Freq-MixStyle* substantially improve over the Baseline on simulated and unseen devices. In particular, on unseen devices, *Freq-MixStyle* significantly outperforms *Mixup*, underlining its superior device generalization capabilities and leading to an overall performance gain compared to *Mixup*. *Ensemble* denotes averaged logits of five PaSST models trained with different Freq-MixStyle configurations. The most significant performance gain of

Method	Real Devices				Simulated Devices				Unseen Devices				Overall
	A	B	C	**Real**	S1	S2	S3	**Sim**	S4	S5	S6	**Unseen**	
PaSST Baseline	72.00	63.29	67.59	67.63	58.13	56.61	58.25	57.66	57.05	57.62	53.67	56.11	60.46
+ Mixup	**72.65**	62.86	**68.04**	**67.85**	59.24	**57.30**	58.81	58.45	57.99	58.24	55.07	57.10	61.13
+ Freq-MixStyle	72.14	**63.55**	67.32	67.68	**59.69**	57.24	**59.99**	**58.97**	**59.03**	**58.65**	**56.98**	**58.22**	**61.64**
Ensemble	73.70	64.07	68.09	68.62	61.30	58.24	60.79	60.11	60.39	60.03	58.76	59.73	62.82

Table 2: PaSST downstream training on the *TAU Urban Acoustic Scenes 2022 Mobile development dataset* [2] following the official test split: Device-wise comparison between Baseline, Mixup ($\alpha = 0.3$), Freq-MixStyle ($\alpha = 0.4, p = 0.4$) and an ensemble of five PaSST models trained with different Freq-MixStyle configurations. The provided accuracies (%) are averaged over three runs and the last 10 epochs of training. The devices are grouped according to real devices (**Real**: A, B, C), the seen, simulated devices (**Sim**: S1, S2, S3) and the unseen, simulated devices (**Unseen**: S4, S5, S6).

the ensemble occurs with unseen devices, underlining the ensemble's robustness.

4. KNOWLEDGE DISTILLATION FROM PASST TO CNN

Knowledge Distillation (KD) [12–14] is the concept used to transfer the knowledge of the well-performing, large PaSST transformer from Section 3.2 to the low-complexity RFR-CNN introduced in Section 3.1. PaSST derives its superiority from its size and the pre-training on Audioset [22]. Hence, in a KD setup, PaSST implicitly passes its semantically rich understanding of acoustic scenes gained from the 527 classes of Audioset to the low-complexity student model. We limit the knowledge transfer to the logits and leave experiments on mimicking embeddings as an interesting future research direction. In Section 5, we experiment with KD and probe for a positive performance impact of the following three variations:

- **Teacher Ensemble**: We ensemble five PaSST models with different Freq-MixStyle configurations by averaging their logits. Freq-MixStyle trained models with different values for α and p tend to have different strengths on real, simulated, and unseen devices, leading to a robust and well-performing ensemble, as shown in Table 2.
- **Superior Teacher**: Given that the full 10-second audio files can be reassembled from the 1-second pieces, the distillation loss can be additionally based on teacher predictions for the full 10 seconds. This way, the student has to match superior teacher predictions while having only access to one-tenth of the input sequence. Eq. 2 presents the adapted loss calculation, adding the new distillation loss $L_{\text{DIST\_SUP}}$ with its corresponding weight λ_{SUP}.

$$L_{\text{TOTAL}} = L_{\text{LABEL}} + \lambda L_{\text{DIST}} + \lambda_{\text{SUP}} L_{\text{DIST\_SUP}} \quad (2)$$

- **Distillation on Out-of-Domain Dataset**: In addition to KD on the *TAU Urban Acoustic Scenes 2022 Mobile development dataset* [2], we experiment with transferring the knowledge using Audioset [22]. For each batch, we sample a batch of Audioset samples of the same size, generate teacher predictions, and calculate the distillation loss. The procedure results in a total loss calculated as shown in Eq. 3.

$$L_{\text{TOTAL}} = L_{\text{LABEL}} + \lambda(L_{\text{DIST}} + L_{\text{DIST\_AUDIOSET}}) \quad (3)$$

When computing student and teacher predictions in the context of Mixup, the same audio snippets are mixed using the same mixing coefficients. Freq-Mixstyle is applied to student and teacher spectrograms independently, which forces the student to match the teacher's soft targets in the context of different device-styles.

4.1. Experimental Setup

Preprocessing: The raw audio signal is down-sampled using a sampling rate of 32 kHz. Spectrograms are generated by applying Short Time Fourier Transformation with a window size of 2048 and an overlap of 744 (approximately 36%) in case of the students and a window size of 800 with an overlap of 320 (40%) for the teachers. A Mel-scaled filter bank is applied to create spectrograms with 256 and 128 mel bins for students and teachers, respectively. The applied preprocessing matches the teacher's pre-training, while, for the student, a higher frequency resolution proved beneficial.
Training: All students in the KD framework are trained with a batch size of 64 for a total amount of 750 epochs, where the models only process one-tenth of the available data each epoch because of the random 1-second cropping. Adam optimizer with a specific learning rate schedule is applied. The learning rate is exponentially increasing to 1×10^{-3} until epoch 150 and linearly decreasing from epoch 250 until epoch 650, dropping to a value of 5×10^{-6}.

5. RESULTS

A summary of the results for the RFR-CNN student model, as presented in Section 3.1, is shown in Table 3. The results are categorized into *Student Baseline* (no KD), *KD Baseline* (*PaSST + Freq-MixStyle* as teacher) and the three KD variations presented in Section 4. In the following, we describe the effect of Freq-MixStyle compared to Mixup, the effect of KD, and the impact of the KD variations.

5.1. Mixup vs. Freq-MixStyle

We investigated different Mixup and Freq-MixStyle configurations and observed that Mixup with $\alpha = 0.4$ and Freq-MixStyle with $\alpha = 0.3$ and $p = 0.4$ yield robust results across a variety of configurations. For *Student Baseline*, Freq-MixStyle outperforms Mixup significantly, achieving the highest performance gains on the unseen device category. In combination with KD, Freq-MixStyle is still slightly superior to Mixup in terms of accuracy but Mixup leads to lower log losses. Freq-MixStyle generalizes much better to unseen devices than Mixup but weakens the performance on real devices.

5.2. Effectiveness of KD

We experiment with three different temperature configurations and adapt the distillation loss weight λ for each temperature. We select the best of the three settings **High (H)** (T=8, λ=800), **Medium (M)** (T=3, λ=100) and **Low (L)** (T=1, λ=50) to be listed in each row of Table 3. KD with Mixup requires high temperatures, while Freq-MixStyle tends to favour low temperature settings.

Method	Configuration					Test Accuracy (%)				Log Loss
	Mixup	Freq-MixStyle	Temp	Teach. Type	AS	Real	Sim	Unseen	Overall	Overall
Student Baseline	✗	✗	-	No	✗	61.97	50.10	40.71	50.92	1.5822
	✓	✗	-	No	✗	62.70	52.48	42.99	52.72	1.4161
	✗	✓	-	No	✗	63.89	56.00	49.98	56.62	1.2344
KD Baseline	✗	✗	H	Single	✗	66.21	57.35	50.14	57.89	1.1316
	✓	✗	H	Single	✗	66.43	58.31	51.32	58.68	1.1063
	✗	✓	L	Single	✗	64.36	58.36	55.12	59.28	1.1431
KD Ensemble	✓	✗	H	Ensemble	✗	66.30	58.65	52.06	59.00	1.0888
	✗	✓	L	Ensemble	✗	64.74	58.59	55.14	59.49	1.1322
KD Superior Teacher	✓	✗	H	Superior	✗	66.53	58.54	51.89	58.98	1.1033
	✗	✓	L	Superior	✗	64.73	58.60	**55.15**	59.49	1.1313
KD Audioset	✓	✗	M	Single	✓	**66.54**	59.09	52.49	59.37	1.0906
	✗	✓	M	Single	✓	64.99	58.50	54.43	59.30	1.0939
	✓	✗	M	Ensemble	✓	66.35	**59.95**	52.99	**59.76**	**1.0794**

Table 3: Results of the low-complexity RFR-CNN student model on the official test split: Accuracies (%) for the device groups **Real** (A, B, C), **Sim** (S1, S2, S3) and **Unseen** (S4, S5, S6), overall accuracy and overall log loss are compared between a baseline using no KD, a baseline using KD and KD variations. Configuration **Temp** is referring to **High (H)** (T=8), **Medium (M)** (T=3) and **Low (L)** (T=1) temperature when calculating teacher and student soft targets, and **AS** indicates the use of Audioset [22] for KD. All results presented are averages of 3 independent runs averaged over the last 10 epochs of training.

KD Baseline outperforms *Student Baseline*, leading to an overall accuracy improvement of 2.66% when comparing the Freq-MixStyle configurations. While the accuracy on real devices improves only slightly, the accuracy on unseen devices increases by 5.14%.

5.3. Effectiveness of KD variations

Using the PaSST ensemble from Table 2 as the teacher improves the results slightly but consistently for both Freq-MixStyle and Mixup, in terms of overall accuracies and log losses. However, the performance gain of more than 1 percentage point in terms of overall accuracy that the ensemble gives compared to a single PaSST model cannot be transferred to the student.

We investigate a range of λ_{SUP} values for each temperature setting for the superior teacher variation. The superior teacher predictions on the 10-second snippets are computed offline. In case of Mixup, the superior predictions are mixed accordingly and no Freq-MixStyle is applied to infer the predictions. *KD Superior Teacher* performs similar to *KD Ensemble* using $\lambda_{\text{SUP}} = 3.0$ for the Mixup configuration and $\lambda_{\text{SUP}} = 1.0$ for the Freq-MixStyle configuration.

With *KD Audioset*, Freq-MixStyle is applied to samples from both datasets, while Mixup is not applied to the samples from Audioset since no hard labels are available. The Mixup configuration shows the best results in terms of accuracy across all Mixup settings, while the Freq-MixStyle configuration only leads to a minor accuracy improvement over *KD Baseline*. However, the Freq-MixStyle configuration achieves the lowest log loss across all Freq-MixStyle experiments. Combining Audioset with *KD Ensemble* and using Mixup leads to the highest overall accuracy and the lowest log loss.

As a final investigation, Figure 1 compares the impact of Freq-MixStyle, Mixup and the KD variations on the overall and the unseen device accuracies. The dominating factor for enhanced generalization to unseen devices is Freq-MixStyle, which clearly outperforms Mixup. The KD variations slightly improve over the KD baseline in terms of overall accuracy but have no significant impact on unseen device performance.

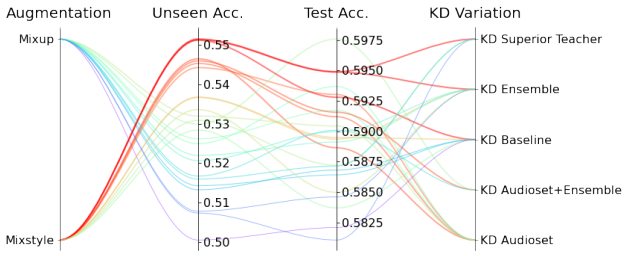


Figure 1: Comparison of the different KD variations and augmentation techniques and their effect on the overall accuracy and the performance on unseen devices. Each line depicts an average over three runs and is colored according to the performance on unseen devices. In addition to the configurations shown in the figure, experiments differ in terms of temperature setting (**High, Medium, Low**).

6. CONCLUSION

In this paper, we distilled the knowledge of a PaSST transformer model into a low-complexity CNN. We showed how the pre-trained PaSST model can be effectively adapted to a downstream task. CNN students that are taught by PaSST models perform significantly better than CNNs learning only from the hard class labels. Based on this, we experiment with three KD variations, including a PaSST teacher ensemble, a superior teacher and KD on Audioset, that show promising performance compared to the KD baseline. To enhance generalization to unseen devices, we compared Mixup with Freq-MixStyle and observed that Freq-MixStyle leads to high accuracy improvements on unseen devices for both PaSST teacher and CNN student models.

7. ACKNOWLEDGMENT

The LIT AI Lab is financed by the Federal State of Upper Austria.

8. REFERENCES

[1] I. Martín-Morató, T. Heittola, A. Mesaros, and T. Virtanen, "Low-complexity acoustic scene classification for multi-device audio: Analysis of DCASE 2021 challenge systems," in *DCASE 2021 Workshop*, 2021.

[2] T. Heittola, A. Mesaros, and T. Virtanen, "Acoustic scene classification in dcase 2020 challenge: generalization across devices and low complexity solutions," in *DCASE 2020 Workshop*, 2020, pp. 56–60.

[3] I. Martín-Morató, F. Paissan, A. Ancilotto, T. Heittola, A. Mesaros, E. Farella, A. Brutti, and T. Virtanen, "Low-complexity acoustic scene classification in dcase 2022 challenge," 2022.

[4] B. Kim, S. Yang, J. Kim, and S. Chang, "QTI submission to DCASE 2021: Residual normalization for device-imbalanced acoustic scene classification with efficient design," DCASE2021 Challenge, Tech. Rep., 2021.

[5] C.-H. H. Yang, H. Hu, S. M. Siniscalchi, Q. Wang, W. Yuyang, X. Xia, Y. Zhao, Y. Wu, Y. Wang, J. Du, and C.-H. Lee, "A lottery ticket hypothesis framework for low-complexity device-robust neural acoustic scene classification," DCASE2021 Challenge, Tech. Rep., 2021.

[6] K. Koutini, F. Henkel, H. Eghbal-zadeh, and G. Widmer, "CP-JKU Submissions to DCASE'20: Low-Complexity Cross-Device Acoustic Scene Classification with RF-Regularized CNNs," DCASE2020 Challenge, Tech. Rep., 2020.

[7] S. A. Janowsky, "Pruning versus clipping in neural networks," *Physical Review A*, vol. 39, no. 12, p. 6600, 1989.

[8] J. Frankle, G. K. Dziugaite, D. Roy, and M. Carbin, "Pruning neural networks at initialization: Why are we missing the mark?" in *ICLR*, 2021.

[9] C. Liu, Z. Zhang, and D. Wang, "Pruning deep neural networks by optimal brain damage," in *INTERSPEECH*, 2014, pp. 1092–1095.

[10] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *CoRR*, vol. abs/1704.04861, 2017.

[11] M. Tan and Q. V. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *ICML*, vol. 97, 2019, pp. 6105–6114.

[12] G. E. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *CoRR*, vol. abs/1503.02531, 2015.

[13] J. Ba and R. Caruana, "Do deep nets really need to be deep?" in *NIPS*, 2014, pp. 2654–2662.

[14] C. Bucila, R. Caruana, and A. Niculescu-Mizil, "Model compression," in *ACM SIGKDD*, 2006, pp. 535–541.

[15] K. Koutini, J. Schlüter, H. Eghbal-zadeh, and G. Widmer, "Efficient training of audio transformers with patchout," *CoRR*, vol. abs/2110.05069, 2021.

[16] B. Kim, S. Yang, J. Kim, H. Park, J. Lee, and S. Chang, "Domain generalization with relaxed instance frequency-wise normalization for multi-device acoustic scene classification," *CoRR*, vol. abs/2206.12513, 2022.

[17] K. Zhou, Y. Yang, Y. Qiao, and T. Xiang, "Domain generalization with mixstyle," in *ICLR*, 2021.

[18] K. Koutini, H. Eghbal-zadeh, and G. Widmer, "Receptive field regularization techniques for audio classification and tagging with deep convolutional neural networks," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 1987–2000, 2021.

[19] K. Koutini, H. Eghbal-zadeh, M. Dorfer, and G. Widmer, "The Receptive Field as a Regularizer in Deep Convolutional Neural Networks for Acoustic Scene Classification," in *EUSIPCO*, 2019.

[20] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," in *ICLR*, 2021.

[21] Y. Gong, Y. Chung, and J. R. Glass, "AST: audio spectrogram transformer," in *Interspeech*, 2021, pp. 571–575.

[22] J. F. Gemmeke, D. P. W. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio set: An ontology and human-labeled dataset for audio events," in *ICASSP*, 2017.

[23] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou, "Training data-efficient image transformers & distillation through attention," in *ICML 2021*, vol. 139, 2021, pp. 10 347–10 357.

[24] H. Heo, J. Jung, H. Shim, and H. Yu, "Acoustic scene classification using teacher-student learning with soft-labels," in *Interspeech*, 2019, pp. 614–618.

[25] S. Takeyama, T. Komatsu, K. Miyazaki, M. Togami, and S. Ono, "Robust acoustic scene classification to multiple devices using maximum classifier discrepancy and knowledge distillation," in *EUSIPCO*, 2020, pp. 36–40.

[26] L. Gao, K. Xu, H. Wang, and Y. Peng, "Multi-representation knowledge distillation for audio classification," *Multim. Tools Appl.*, vol. 81, no. 4, pp. 5089–5112, 2022.

[27] J. Jung, H. Heo, H. Shim, and H. Yu, "Knowledge distillation in acoustic scene classification," *IEEE Access*, vol. 8, pp. 166 870–166 879, 2020.

[28] H. Zhang, M. Cissé, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," in *ICLR*, 2018.

[29] K. Koutini, H. Eghbal-zadeh, and G. Widmer, "CP-JKU submissions to DCASE'19: Acoustic scene classification and audio tagging with receptive-field-regularized CNNs," DCASE2019 Challenge, Tech. Rep., 2019.

[30] K. Koutini, S. Jan, and G. Widmer, "CPJKU Submission to DCASE21: Cross-Device Audio Scene Classification with Wide Sparse Frequency-Damped CNNs," DCASE2021 Challenge, Tech. Rep., 2021.

[31] F. Schmid, S. Masoudian, K. Koutini, and G. Widmer, "CP-JKU submission to dcase22: Distilling knowledge for low-complexity convolutional neural networks from a patchout audio transformer," DCASE2022 Challenge, Tech. Rep., 2022.

FEATURE SELECTION USING ALTERNATING DIRECTION METHOD OF MULTIPLIER FOR LOW-COMPLEXITY ACOUSTIC SCENE CLASSIFICATION

Lorenz P. Schmidt, Beran Kiliç, Nils Peters

Friedrich-Alexander-Universität Erlangen-Nürnberg, Erlangen, Germany
International Audio Laboratories, Erlangen, Germany
{lopa.schmidt, beran.kilic, nils.peters}@fau.de

ABSTRACT

Acoustic Scene Classification (ASC) is a common task for many resource-constrained devices, e.g., mobile phones or hearing aids. Limiting the complexity and memory footprint of the classifier is crucial. The number of input features directly relates to these two metrics. In this contribution, we evaluate a feature selection algorithm which we also used in this year's challenge [1].

We propose binary search with hard constraints on the feature set and solve the optimization problem with Alternating Direction Method of Multipliers (ADMM). With minimal impact on accuracy and log loss, results show the model complexity is halved by masking 50% of the Mel input features. Further, we found that training convergence is more stable across random seeds. This also facilitates the hyperparameter search. Finally, the remaining Mel features provide an insight into the properties of the DCASE ASC data set.

Index Terms— ASC, feature selection, hard thresholding

1. INTRODUCTION

An essential part of the training and inference in machine learning is feature extraction. For Acoustic Scene Classification (ASC), feature extraction is often combined with domain knowledge to make features invariant to pitch changes, amplitude differences, and other modifications [2]. For instance, a Mel-scaled filterbank with additional log transformation and normalization is commonly used. In our DCASE 2022 challenge contribution [1], we saw benefits of using automated features selection to meet the low-complexity model constraint [3]. Here, we explain and further examine this approach.

Studying features and performing subset selection [4] is an experimental step that happens before the actual model training. Several benefits make feature selection interesting to a model developers: First, redundant features increase the computational complexity of the model without benefit to the classification accuracy. For example, in convolutional models, the number of MACs (Multiply-Accumulate) operations scale linearly with the number of features [5]. Second, feature selection makes the model more amenable to pruning as the dimensionality of data is reduced. Third, noisy or weakly correlated features can make the training unstable and increase overfitting. Reducing the number of features avoids modelling the noise of the input and therefore decreases variance in the generalization error. This means that a good model can be found with fewer runs of different seeds. Finally, from a research perspective, selecting features allow us to draw conclusions on the generation process and to interpret patterns of the feature sets. This may give a clue about deficiencies and better feature parametrization.

Sparse estimation methods and the concept of parsimonious representation [6] is an established field for linear models and found recent application for DNNs [7, 8]. Unstructured pruning and feature selection is expressed in linear models as zeroing out model parameters, which also removes correlation between feature and target. In (non-linear) DNNs, this relation is more complex because parameters' magnitudes are not linearly connected to outcomes. Further solutions may differ for each run, making comparisons and drawing conclusions on feature importance difficult. Applying a proximal operator to the iterate directly gives magnitude pruning [9], often seen in DNN methods such as deep compression [10] or lottery ticket [11]. But an abundance of methods exist and are used for compression, including reweighted l_2 penalized techniques [6], log barriers [12], active sets [13] and proximal methods [9].

In Section 2, we state the problem of optimizing with feature cardinality constraint C and derive an algorithm with the Alternating Direction Method of Multipliers (ADMM) [14] and proximal operators [9]. We also explain how binary search is used for finding a bound on C adhering to a threshold on performance loss. We conduct experiments in Section 3, discuss results in Sec. 4, and conclude this study with an outlook in Sec. 5.

1.1. Study findings

- Masking half of the features results in minimal loss of accuracy; discarding them saves half of the DNN model complexity
- Feature selection reduces training variance and improves convergence speed; enabling efficient single-run hyperparameter searches
- Different model architectures vary in selected feature sets
- Mel features above 19 kHz seem to provide unexpected benefits to ASC, with fewer information contained between 9 kHz and 19 kHz

2. FEATURE SELECTION PROBLEM

In this section, we will derive a feature selection method using ADMM. Instead of using an explicit penalty (e.g. l_1 norm on the feature support), we chose to have hard constraint on the feature cardinality. We do this to make the application applicable to binary search as we are interested in the feature sub-set and the amount of features we can mask. The proximal operator makes our method applicable to a number of different constraints, including matrix constraints [15] or mixed ℓ_1/ℓ_2 with elastic net regularization [9].

The International Audio Laboratories Erlangen are a joint institution of the Friedrich-Alexander-Universität Erlangen-Nürnberg and Fraunhofer IIS.

2.1. Feature selection using ADMM

We split the DNN loss function and non-smooth constraint with dual decomposition and solve the sub-problem of constraining the feature support with an proximal operator.

Our objective is to minimize the loss function of a model while fulfilling constraints on the support of the feature set. The optimization problems reads then as

$$\min_{\theta,\mathbf{w}} f_\theta(\mathbf{w} \circ \mathbf{X}) \text{ s.t. } \text{card}(\mathbf{w}) \leq C, \tag{1}$$

where $\mathbf{X}$ is the input spectrogram, $\mathbf{w}$ the feature support vector (masking by rows with $\circ$), θ the model weights, $\text{card}(\mathbf{w})$ the number of non-zero elements, and C is the hard constraint on the feature set. We move the hard constraint into the loss function with indicator

$$I_C(\mathbf{w}) = \begin{cases} 0 & \text{if } \text{card}(\mathbf{w}) \leq C \\ +\infty & \text{otherwise} \end{cases} \tag{2}$$

and split along the feature mask $\mathbf{w}$ by introducing equality constraints

$$\min_{\theta,\mathbf{w}} f_\theta(\mathbf{w} \circ \mathbf{X}) + I_C(\hat{\mathbf{w}}) \quad \text{s.t.} \quad \mathbf{w} = \hat{\mathbf{w}}. \tag{3}$$

The optimization problem contains the smooth and differentiable term of the DNN loss function and the non-smooth indicator function $I_C(\hat{\mathbf{w}})$. While this cannot be solved directly via gradient descent, we can relax the problem by introducing a Lagrangian multiplier for the equality constraint $\mathbf{w} = \hat{\mathbf{w}}$ and optimizing those via gradient ascent. We can derive the augmented Lagrangian as

$$\mathcal{L}_p(\theta, \mathbf{w}, \hat{\mathbf{w}}, \mu) = f_\theta(\mathbf{w} \circ \mathbf{X}) + I_C(\hat{\mathbf{w}}) + \langle \mu, \mathbf{w} - \hat{\mathbf{w}} \rangle + \frac{\rho}{2} \|\mathbf{w} - \hat{\mathbf{w}}\|_2^2 \tag{4}$$

with μ the dual variable and ρ smoothing factor of a Moreau envelope [16]. A stable saddle point does not have to exist for non-linear DNNs. In practise, treating ρ as a hyperparameter, we observe convergence. The scaled version with dual variables $\mathbf{u} = \frac{1}{\rho}\mu$ is then

$$\mathcal{L}_p(\theta, \mathbf{w}, \hat{\mathbf{w}}, \mu) = f_\theta(\mathbf{w} \circ \mathbf{X}) + I_C(\hat{\mathbf{w}}) + \frac{\rho}{2}(\|\mathbf{w} - \hat{\mathbf{w}} + \mathbf{u}\|_2^2 - \|\mathbf{u}\|_2^2) \tag{5}$$

and we can optimize over $(\theta, \mathbf{w})$, $\hat{\mathbf{w}}$ and μ. In each cycle $k = 0, 1, \ldots$ we update the variables separately and repeat until convergence. We use the iteration number k for variables, which are fixed in the following section. For $(\theta, \mathbf{w})$ we can see that

$$\nabla \mathcal{L}_p(\theta, \mathbf{w}, \hat{\mathbf{w}}^k, \mu^k) = \nabla f_\theta(\mathbf{w} \circ \mathbf{X}) + (0, \rho(\mathbf{w} - \hat{\mathbf{w}} + \mathbf{u})) \tag{6}$$

and therefore the gradient of θ is unchanged, but $\mathbf{w}$ is influenced by the regularization term. In practice the PyTorch loss function [17] is extended with the regularization term, while derivatives are generated with automatic differentiation.

The sub-problem of $\hat{\mathbf{w}}$ has a closed-form solution. The closest projection onto the support constraint

$$\min_{\hat{\mathbf{w}}} I_C(\hat{\mathbf{w}}) + \frac{\rho}{2} \|\mathbf{w} - \hat{\mathbf{w}} + \mathbf{u}\|_2^2, \tag{7}$$

gives a closed-form solution $\hat{\mathbf{w}} = \Pi_C(\mathbf{w} + \mathbf{u})$ [9]. Basically, we keep the C largest elements of $\mathbf{w} + \mathbf{u}$, while setting the rest to zero. Finally, optimizing for $\mathbf{u}$ results in gradient ascent of the dual variable and in total we have

$$\begin{aligned} (\theta^{k+1}, \mathbf{w}^{k+1}) &= (\theta^k, \mathbf{w}^k) - \gamma_k \nabla f_\theta(\mathbf{w} \circ \mathbf{X}) - (0, \rho(\mathbf{w} - \hat{\mathbf{w}}^k + \mathbf{u}^k)) \\ \hat{\mathbf{w}}^{k+1} &= \Pi_C(\mathbf{w}^{k+1} + \mathbf{u}^k) \\ \mathbf{u}^{k+1} &= \mathbf{u}^k + \mathbf{w}^{k+1} - \hat{\mathbf{w}}^{k+1}, \end{aligned} \tag{8}$$

with γ_k the learning rate of gradient descent. Observe that we keep the penalty ρ fixed, which in turn means that we have a fixed learning rate for the duals.

2.2. Learning phase

In Section 2.1, we use ADMM to derive an optimization routine for our hard constraint. Because we make no assumption about the model and we cannot ensure that a saddle point even exists for our Lagrangian, we have to take care of the optimization process. First, we pre-train the model with all features and use those weights as initialization for our feature search. Second, we then apply Eq. 8 for a number of epochs. The learning rate of gradient ascent is fixed, while that of gradient descent anneals to zero over the learning cycle. This ensures that the model loss dominates at the beginning, reducing to zero those features unnecessary for good performance. At the end of the training cycle, the penalty of the ADMM algorithm dominates that of the model, forcing the constraint on feature support to be fulfilled, even though the performance may suffer. Finally, after finding the optimal feature mask, we prune unused features and re-train for a small number of epochs (less than 10) to fine-tune the model.

3. EXPERIMENTS

Our experiments use the DCASE 2022 Task 1 [3] split with samples for training (139619), validation/development (29680), and testing (29680). Each sample belongs to one of 10 different acoustic scenes, captured at $44.1\,\text{kHz}$ in one of 12 European cities by 3 real (A, B, C) or 6 simulated recording devices (S1-S6).

3.1. BC-ResNet model

As a state-of-the-art ASC model, we use a BC-ResNet [18] architecture, specifically the parametrization of BC-ResNet Mod-8 [19] which performed excellent in the DCASE 2021 challenge.

3.1.1. Pre-training

Mel-scaled spectrograms with 512 bands are extracted from full-band audio input. We use a frame length of $93\,\text{ms}$ (4096 samples), an overlap of $30\,\text{ms}$ (1323 samples) and Hamming weighting. We apply a logarithmic transformation and residual normalization [19] with $\lambda = 0.1$ to generalize across different devices.

The input features are augmented to avoid overfitting. We use a random roll of 40% of the signal length. We also use Specaugment [20] in frequency domain with mask parameter of 20 and Mixup [21] with $\alpha = 0.2$. We apply stochastic gradient descent (SGD) to the model and train for 60 epochs. The learning rate increases to 0.01 in a warmup phase (3 epochs) and then decreased to 0.0001 in the remaining 57 epochs. We use momentum of 0.9, weight decay of 0.001, and a mini-batch size of 64.

We train the BC-ResNet with eight different random seeds. The accuracy during training is depicted in Fig. 1 and the final performance can be seen in Table 1. With 512 Mel bands and $1\,\text{sec}$ audio

per sample the model has a computational complexity of 300 k parameters and 102 MMACs. We select the best performing model (in terms of the log loss on the testing data) as the initial model weights to use in the following experiments.

3.1.2. *Optimizing with feature constraints*

After pre-training a BC-ResNet model, we are applying Eq. 8 with a given feature constrain C. The primal $\mathbf{w}^0$ is initialized with ones and projected; and dual variables $\hat{\mathbf{w}}^0, \mathbf{u}^0$ with zeros. This penalizes all features in the first step; the second step proceeds normally by masking smallest magnitudes. We update the model weights with an initial learning rate of 4×10^{-3} and cosine annealing for five epochs. After finding a set of features and masking them, we fine-tune the model for another five epochs, resetting the optimizer to learning rate 4×10^{-3} first.

The convergence of ADMM highly depends on choice of ρ. We treat ρ as a hyperparameter and found $\rho = 3 \times 10^{-4}$ to be a good trade-off between fast convergence and stability. Unfortunately, the convergence of the variables depends not only on ρ but also on the number of masked features. As illustrated in Fig. 1, we observe that when masking more features, the initial convergence seems to be faster. When masking more features than realizable (without losing performance), the convergence continues only once the model's learning rate is annealed enough. As we use cosine annealing, this happens in the later phase of training, where the regularization term introduced by ADMM dominates the update step of θ and $\mathbf{w}$. This ensures that the cardinality constraint in Eq. 1 is fulfilled, though we may lose performance. We finally see no difference in convergence of primal and dual variables: both converge in similar rates and to similar residuals (see Fig. 2).

3.1.3. *Finding a good constraint on the feature number*

Choosing an appropriate C prior running feature selection is difficult. If too high, then the model remains too complex and training speed suffers; if too low, then additional error is introduced

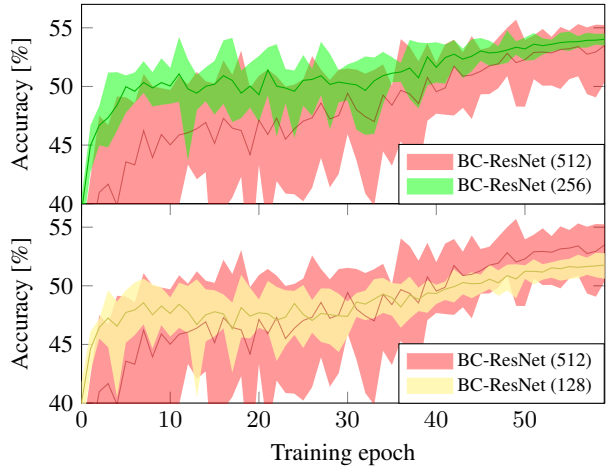


Figure 1: Variance and error during training for validation accuracy. Max, min, and geometric means are depicted for eight runs with different random seeds are shown. Reducing number of features from 512 to 256 improves stability and reduces variance, selecting only 128 features affects performance.

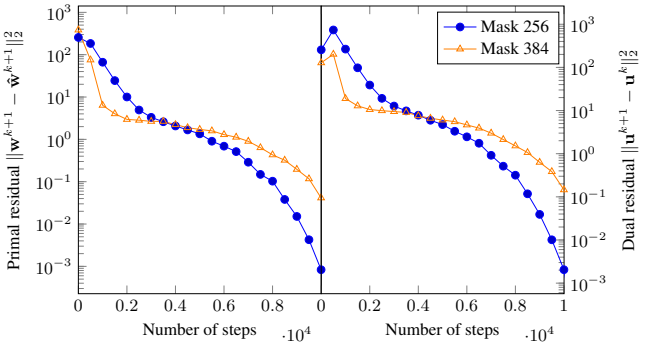


Figure 2: Convergence of primal (left) and dual (right) variables for masking 256 and 384 features. Masking more features initially increases the influence of the regularization term, but requires more iterations to reach low residuals.

(see also Sec. 4.1). Therefore we apply the feature selection in context of a meta heuristic by introducing a maximal degradation $\epsilon = 3 \times 10^{-2}$ of log loss for our pre-trained model. The heuristic performs binary search, halving upwards if the log loss is under our threshold ϵ; downwards, if not. For reliable results, we run the experiment with 8 different random seeds and select the best performance. Once the step size is below 8, we step alternate to stay below the threshold and explore different log loss in that region. Alternatively, (in a more conservative search) the average over all runs could be compared to the absolute threshold. In the end, we conclude with $C = 274$ masked features (see also Fig. 3).

3.2. Linear model

For comparison with the non-linear BC-ResNet model, we also train a linear Support Vector Classifier (SVC) [22]. The SVC performs inference with temporally global features. We reduce the time axis of our Mel spectrogram with moments up to fifth order (mean, variance, skewness, kurtosis, and hyperskewness for each band), resulting in 2560 features per data sample.

Similar to the previous experiment, we apply l_1 regularization during feature selection. The penalty is treated as a hyperparameter. A search found that the model gives the best accuracy on the testing data for $\nu = 100$. We have seen almost no variance for the final

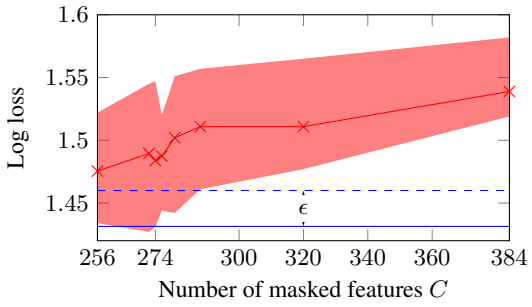


Figure 3: Log loss on validation set during binary search. Max, min, and geometric means of eight runs with different random seeds are shown. The dashed line indicates the threshold $\epsilon = 0.03$ of the log loss to the performance of the pre-trained model (see Sec. 3.1.1).

Model	MMACs	Acc. (%)	Log Loss
Baseline [3]	29.23	42.90	1.575
SVC ($\nu = 100$)	0.026	43.79	1.780
BC-ResNet (512)	101.89	54.24	1.370
BC-ResNet (256)	51.53	54.10	1.496
BC-ResNet (128)	26.35	52.80	1.613

Table 1: MACs and performance on development set for best models. The number of features is indicated in brackets behind the model.

testing results, making the need to do multiple runs unnecessary. Compared to the DCASE 2022 baseline [3], this SVC has a better model accuracy and a worse log loss, but requires significantly fewer MACs (see Table 1).

4. DISCUSSION

4.1. Training variance and model reduction

Having many noisy or weakly correlated features to the targets increases the risk of overfitting. We see this trend in the variance of our validation results. As can be seen in Fig. 1 for the BC-ResNet without feature selection (512 features), the model uses noisy features. Even at the end of training, its accuracy differs by several percentages. Consequently, finding a good solution requires multiple training runs, which is time- and energy-consuming. Halving the number of features to 256 has several effects. First, in the initial training phase, the model converges much faster on average as the variance is reduced drastically. Then the accuracy approximately stays the same until the final phase. All runs with different seeds give a good final result. Thus, hyperparameter searches can be executed much more efficiently. This reduces experimentation time and improves sustainability of our model search.

When reducing the number of Mel features to 128, variance slightly decreases between epochs 30 to 50, but the final model performance was compromised. We decrease variance, but underfit the dataset by using too few features. This behavior is also supported by our binary search, which suggested 278 masked features, i.e., 234 remaining features (see Fig. 3).

The BC-ResNet model uses only convolutional layers [18]. Consequently, the total number of MACs scales linearly with the number of features. By selecting only 50% of the features, we also halve the number of operations from 101.89 MMACs to 51.53 MMACs without significant change in performance. When further reducing the feature count, the log loss increases significantly (see Table 1).

4.2. Interpretation of feature selection

Selection allows making interpretation about the feature set. This is different to feature extraction, where all features are retained and only their representation is changed. Fig. 4 depicts the histogram of the selected feature in our experiments. For the BC-ResNet models, most of the information is contained below 8 kHz. This matches with the literature that 16 kHz sampling rate (leading to a 8 kHz lowpass filtering) gives good ASC results. However, we also tested pre-training of the BC-ResNet with a naïve selection of all features below 8 kHz, but model performance was below that of BC-ResNet (128).

The feature importance around ≈19.8 kHz for all BC-ResNet models is unexpected. It may be related to some recording artifacts by some device types. Masking those bands may improve generalization. Other features are more plausible, e.g., we find that features below 500 Hz are always selected (as they relate to low frequency or harmonics) and the region between 2.5 kHz and 8 kHz contain a lot of information as well, e.g., for consonants in speech.

Also visible in Fig. 4, one can see that feature selection subsamples the frequency bins, suggesting that the band width of ≈5.4 Hz is too small. However, features of very low frequencies (below 60 Hz) are fond to be also relevant. With increasing masking, the feature selection in regions above 500 Hz starts to subsample even more. This is most prominent when examining the 128 remaining features. In this case, only a handful of features are selected.

When comparing the BC-ResNet results to the linear SVC model (upper row of Fig. 4), we see that the regions below 500 Hz and between 1.5 kHz to 8 kHz contain most of the important information. On the other hand, the linear model uses higher frequencies above 9 kHz, but not the prominent peak at 19.8 kHz selected by BC-ResNet. This may suggest that the moments of the SVC may model different information than the convolutional layers of BC-ResNet. This indicates that feature selection is partially model-dependent. More work is needed to better understand this relationship.

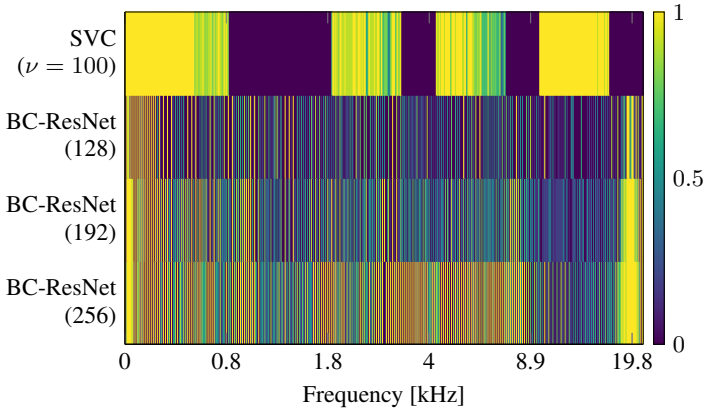


Figure 4: Selected frequency bands for SVC (upper row) and BC-ResNet models across input feature sizes. The selection patterns are averaged across all 8 runs. The color indicates the number of occurrences a feature was selected; with yellow color representing features that are used in all runs. For the SVC model a frequency band counts when at least one moment is selected.

5. CONCLUSION

We showed that feature selection is an important experimental step when developing efficient machine learning models. In addition to providing insights into the working of signal processing, it also improves stability and makes hyperparameter search more efficient. We applied this method in the DCASE 2022 challenge to reduce the total number of MACs [1]. In future, we aim to further examine how features map to different scenes, devices, and cities. This would allow for optimizing methods and models.

6. REFERENCES

[1] L. Schmidt, B. Kiliç, and N. Peters, "Structured filter pruning and feature selection for low complexity acoustic scene classification," DCASE2022 Challenge, Tech. Rep., June 2022.

[2] J. Andén and S. Mallat, "Deep scattering spectrum," *IEEE Transactions on Signal Processing*, vol. 62, 04 2013.

[3] I. Martín-Morató, F. Paissan, A. Ancilotto, T. Heittola, A. Mesaros, E. Farella, A. Brutti, and T. Virtanen, "Low-complexity acoustic scene classification in DCASE 2022 challenge," 2022. [Online]. Available: https://arxiv.org/abs/2206.03835

[4] R. Kohavi and G. H. John, "Wrappers for feature subset selection," *Artificial Intelligence*, vol. 97, no. 1, pp. 273–324, 1997, relevance. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S000437029700043X

[5] M. Taghavi and M. Shoaran, "Hardware complexity analysis of deep neural networks and decision tree ensembles for real-time neural data classification," in *2019 9th International IEEE/EMBS Conference on Neural Engineering (NER)*, 2019, pp. 407–410.

[6] F. Bach, R. Jenatton, J. Mairal, and G. Obozinski, "Optimization with sparsity-inducing penalties," *Found. Trends Mach. Learn.*, vol. 4, no. 1, p. 1–106, jan 2012. [Online]. Available: https://doi.org/10.1561/2200000015

[7] Y. Li, C.-Y. Chen, and W. Wasserman, "Deep feature selection: Theory and application to identify enhancers and promoters," in *2015 Annual International Conference on Research in Computational Molecular Biology (RECOMB)*, 04 2015.

[8] M. Wojtas and K. Chen, "Feature importance ranking for deep learning," in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., vol. 33. Curran Associates, Inc., 2020, pp. 5105–5114. [Online]. Available: https://proceedings.neurips.cc/paper/2020/file/36ac8e558ac7690b6f44e2cb5ef93322-Paper.pdf

[9] N. Parikh, S. Boyd, *et al.*, "Proximal algorithms," *Foundations and trends® in Optimization*, vol. 1, no. 3, pp. 127–239, 2014.

[10] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding," 2015. [Online]. Available: https://arxiv.org/abs/1510.00149

[11] J. Frankle and M. Carbin, "The lottery ticket hypothesis: Finding sparse, trainable neural networks," 2018. [Online]. Available: https://arxiv.org/abs/1803.03635

[12] H. Kervadec, J. Dolz, J. Yuan, C. Desrosiers, E. Granger, and I. B. Ayed, "Constrained deep networks: Lagrangian optimization via log-barrier extensions," 2019. [Online]. Available: https://arxiv.org/abs/1904.04205

[13] J. Kim and H. Park, "Fast active-set-type algorithms for l1-regularized linear regression," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, Y. W. Teh and M. Titterington, Eds., vol. 9. Chia Laguna Resort, Sardinia, Italy: PMLR, 13–15 May 2010, pp. 397–404. [Online]. Available: https://proceedings.mlr.press/v9/kim10a.html

[14] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein, *et al.*, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends® in Machine learning*, vol. 3, no. 1, pp. 1–122, 2011.

[15] J. E. Cohen, "Computing the proximal operator of the ℓ_1 induced matrix norm," 2020. [Online]. Available: https://arxiv.org/abs/2005.06804

[16] C. Planiden and X. Wang, *Proximal Mappings and Moreau Envelopes of Single-Variable Convex Piecewise Cubic Functions and Multivariable Gauge Functions*. Cham: Springer International Publishing, 2019, pp. 89–130. [Online]. Available: https://doi.org/10.1007/978-3-030-11370-4_5

[17] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[18] B. Kim, S. Chang, J. Lee, and D. Sung, "Broadcasted residual learning for efficient keyword spotting," 2021. [Online]. Available: https://arxiv.org/abs/2106.04140

[19] B. Kim, S. Yang, J. Kim, and S. Chang, "QTI submission to DCASE 2021: Residual normalization for device-imbalanced acoustic scene classification with efficient design," DCASE2021 Challenge, Tech. Rep., June 2021.

[20] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "SpecAugment: A simple data augmentation method for automatic speech recognition," in *Interspeech 2019*. ISCA, sep 2019. [Online]. Available: https://doi.org/10.21437%2Finterspeech.2019-2680

[21] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," 2017. [Online]. Available: https://arxiv.org/abs/1710.09412

[22] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

LOW-COMPLEXITY CNNS FOR ACOUSTIC SCENE CLASSIFICATION

Arshdeep Singh, Mark D. Plumbley

Centre for Vision, Speech and Signal Processing (CVSSP)
University of Surrey, UK
Email: {$arshdeep.singh, m.plumbley$}@surrey.ac.uk

ABSTRACT

This paper presents a low-complexity framework for acoustic scene classification (ASC). Most of the frameworks designed for ASC use convolutional neural networks (CNNs) due to their learning ability and improved performance compared to hand-engineered features. However, CNNs are resource hungry due to their large size and high computational complexity. Therefore, CNNs are difficult to deploy on resource constrained devices. This paper addresses the problem of reducing the computational complexity and memory requirement in CNNs. We propose a low-complexity CNN architecture, and apply pruning and quantization to further reduce the parameters and memory. We then propose an ensemble framework that combines various low-complexity CNNs to improve the overall performance. An experimental evaluation of the proposed framework is performed on the publicly available DCASE 2022 Task 1 that focuses on ASC. The proposed ensemble framework has approximately 60K parameters, requires 19M multiply-accumulate operations and improves the performance by approximately 2-4 percentage points compared to the DCASE 2022 Task 1 baseline network.

Index Terms— Acoustic scene classification, Low-complexity, Pruning, Quantization. Convolution neural network.

1. INTRODUCTION

Convolutional neural networks (CNNs) have shown state-of-the-art performance in comparison to traditional hand-crafted methods in various domains [1]. However, CNNs are resource hungry due to their large size and computational complexity [2, 3], and hence it is difficult to deploy CNNs on resource constrained devices. For example, Cortex-M4 devices (STM32L496@80MHz or Arduino Nano 33@64MHz) have a maximum allowed limit of 128K parameters and 30M multiply-accumulate operations (MACs) per second during inference. Thus, the issue of reducing the size and the computational cost of CNNs has drawn a significant amount of attention in the detection and classification of acoustic scenes and events (DCASE) research community.

In the literature, some CNN parameters such as filters or weights may be redundant, and contribute to extra memory and computational complexity only [4, 5]. For example, Li et al. [6] found that 64% of the parameters which contribute approximately 34% of computation time are redundant. Removing the redundant parameters from CNNs gives similar performance with an advantage of reduced memory and less computational cost.

The majority of the methods applied to eliminate redundant parameters are on filter pruning [7, 8, 9, 10], where a redundant filter from the network is being eliminated. Filter pruning methods have been widely employed in the computer vision. However, only a few works [5, 11] have applied filter pruning methods in the audio domain, and the issue of designing CNNs for resource constrained devices with constraints on both memory and MACs have not yet been fully explored. Recently, the DCASE challenge Task 1 focuses on designing low-complexity frameworks for ASC with constraints on both memory and MACs. The DCASE challenge 2022 Task 1 [12] provides a baseline CNN, which has 46512 parameters and 29.24M MACs, and achieves an ASC accuracy and log-loss approximately 43% and 1.575 respectively.

This paper aims to design "low-complexity CNNs" which have a maximum number of parameters less than 128K and a maximum number of MACs per seconds less than 30M and performance better than that of the DCASE 2022 Task 1 baseline network for ASC.

The major contributions of the paper is summarized below,

(a) We design a "low-complexity" CNN that has fewer parameters, fewer MACs and better classification performance than the DCASE 2022 Task 1 baseline CNN.

(b) A filter pruning method is applied to compress the "low-complexity CNN" of (a) further. Subsequently, we quantize each parameter from float32 to INT8 data type, reducing networks memory by four times.

(c) An ensemble approach is proposed which combines predictions obtained from several (b) low-complexity CNNs.

(d) Experimental evaluation is undertaken to compare performance of proposed framework with classical methods such as Gaussian Mixture Model (GMMs) and random forest (RF) classifiers and learning based methods such as dictionary learning and a pre-trained high complexity CNN, VGGish with the proposed framework.

The rest of the paper is organized as follows. In Section 2, a brief overview of dataset used and features used for experimentation is described. In Section 3, a procedure to obtain low-complexity CNN and an ensemble framework is described. Section 4 presents experimental analysis. Finally, discussion and conclusion is presented in Section 5 and 6 respectively.

2. EXPERIMENTAL DATASET AND FEATURE EXTRACTION

DCASE 2022 Task 1 uses the TAU Urban Acoustic Scenes 2022 Mobile, development and evaluation datasets [13]. The dataset contains recordings from 12 European cities in 10 different acoustic scenes using 4 different recording devices. Each audio recording has 1 second length. The development dataset is divided in training and validation sets. The training dataset consists of 139620 audio examples and the validation dataset consists of 29680 audio examples. The evaluation dataset consists of only audio examples

without any groundtruth available publicly, and the evaluation is performed by the DCASE challenge community.

Feature extraction: For time-frequency representations, log-mel band energies of size (40 × 51) corresponding to an audio signal of 1 second length are extracted. A Hamming asymmetric window of length 40ms, and a hop length of 20ms is used to extract magnitude spectrogram. Next, log-mel spectrogram is computed using 40 mel bands.

3. OBTAINING LOW-COMPLEXITY CNNS

Low-complexity optimal CNN architecture: We design a simple low-complexity CNN architecture which consists of three convolutional layers (C1, C2 & C3), two pooling layers (P1, P2), a dense layer (D) and a classification layer. We perform an empirical analysis to select an appropriate size of CNN filters and activation functions across the different layers. The filter size for each convolutional layer is chosen from a set, $\{1 \times 1, 3 \times 3, 5 \times 5, 7 \times 7\}$. We choose hyperbolic tangent (tanh) or rectified linear unit (ReLU) activation function for different layers. The low-complexity CNN is trained using the training dataset with a batch size of 64 with an Adam optimizer for 1000 epochs. A categorical cross-entropy loss function is used during the training process. We apply an early stopping criterion to yield the best network that gives the minimum log-loss for the validation dataset.

The optimal architecture obtained after performing empirical analysis is given in Table 1. The details of the experiments are given in Section 4. The proposed architecture requires approximately 5M MACs to produce an output corresponding to an input of size (40 x 51), and has 14886 parameters. The performance of the trained network is measured in terms of accuracy and log-loss, averaged over 5 different iterations.

Reducing redundancy in the low-complexity optimal CNN via filter pruning: To eliminate redundant filters from the low-complexity optimal architecture as given in Table 1, we apply a filter pruning strategy. For each convolutional layer, we identify filter pairs which are similar. Our hypothesis is that similar filters produce similar output or feature maps and hence, contribute to redundancy only. Therefore, one of the similar filters can be eliminated. The similarity between the filters is measured using a cosine distance. We identify the closest filter pairs for each layer separately. A filter from each pair is deemed redundant and eliminated from the network. More information about the similarity based filter pruning method can be found at [14].

The number of redundant filters obtained after performing similarity-based filter pruning for C1 layer is 4 out of 16, C2 layer is 4 out of 16 and C3 layer is 10 out of 32. We obtain 6 different pruned networks that are obtained after pruning C1 layer only, C2 layer only, C3 layer only, C1 and C2 layers, C2 and C3 layers, C1 and C2 and C3 layers. The number of MACs and the number of parameters for each pruned network are given in Table 2.

To regain the loss in performance due to pruning, the pruned networks are fine-tuned in a training similar to that of the unpruned low-complexity optimal network.

Reducing memory requirement via quantization: To reduce size of the network, we perform quantization on parameters of each pruned network using Tensorflow-Lite (TFLite). TFlite [15] is an open-source framework to quantizes a pre-trained full-precision network for embedded devices. We quantize the network parameters from 32-bit floating point to 8-bit integers. This leads to reduce the network size 4x.

Table 1: Low-complexity optimal CNN architecture. Here, tanh is a hyperbolic tangent activation function and ReLU is a rectified linear unit activation function.

Layer name	Description
C1	Convolution 16@(3 × 3) + Batch Normalization + tanh
C2	Convolution 16@(3 × 3) + Batch Normalization + ReLU
P1	Average Pooling (5 × 5)
C3	Convolution 32@(3 × 3) + Batch Normalization + tanh
P2	Average Pooling (4 × 10)
D	Dense (100) + tanh
Classification	softmax (10)

An ensemble framework: Next, the predictions obtained from various pruned networks are aggregated together in an ensemble framework. The total number of parameters in the ensemble framework that aggregates predictions from all 6 pruned networks are 70.97K and the total number of MACs are 23.84M.

The codes and the pruned networks can be found online [1][2].

4. PERFORMANCE ANALYSIS

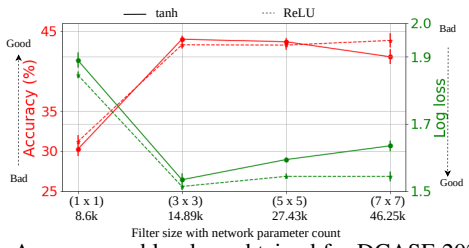


Figure 1: Accuracy and log loss obtained for DCASE 2022 Task 1 development validation dataset using the low-complexity CNNs at different filter size and activation function.

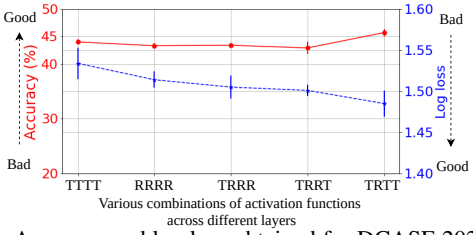


Figure 2: Accuracy and log loss obtained for DCASE 2022 Task 1 development validation dataset using the low-complexity CNN with (3 x 3) filter size when different combinations of activation functions are applied across "C1", "C2", "C3" and "D" layers. Here, "T" represents tanh activation function and "R" represents ReLU activation function.

The accuracy and the log-loss obtained at various filter size and different activation functions in the low-complexity CNN is shown in Figure 1. The low-complexity CNN with (3 x 3) filter size outperforms networks with other filter size.

Using "ReLU" activation function across all layers results in smaller log-loss in comparison to that of the the "tanh" activation

[1] Link: Pruned Quantized models, confusion matrices, evaluation scripts.
[2] Link: Model size and complexity calculation.

Table 2: Various low-complexity CNNs obtained after pruning and applying quantization (INT8).

Sr No.	Network Name	Pruned layer	Architecture (C1-C2-C3-Dense)	Parameters	Size (KB)	MACs (millions)
1	Unpruned optimal low-complexity	NA	16-16-32-100	14886	18.59	5.41
2	Pruned_C1	C1	12-16-32-100	14254	17.86	4.16
3	Pruned_C2	C2	16-12-32-100	13138	16.85	4.13
4	Pruned_C3	C3	16-16-22-100	11396	15.11	5.29
5	Pruned_C12	C1 + C2	12-12-32-100	12650	16.26	3.18
6	Pruned_C23	C2 + C3	16-12-22-100	10008	13.73	4.04
7	Pruned_C123	C1 + C2 + C3	12-12-22-100	9520	13.14	3.08

function. On the other hand, combining "tanh" and "ReLU" activation functions across different layers improves the log-loss further as shown in Figure 2. We find that the "tanh" activation function in the first layer and "ReLU" for other layers improves the performance over that of using "ReLU" for all layers. Finally, we choose the low-complexity optimal CNN architecture which performs better than that of architectures with different combinations of activation function. The optimal low-complexity CNN has "tanh" activation function in all layers except C2 and the filter size is (3 x 3) in each convolutional layers.

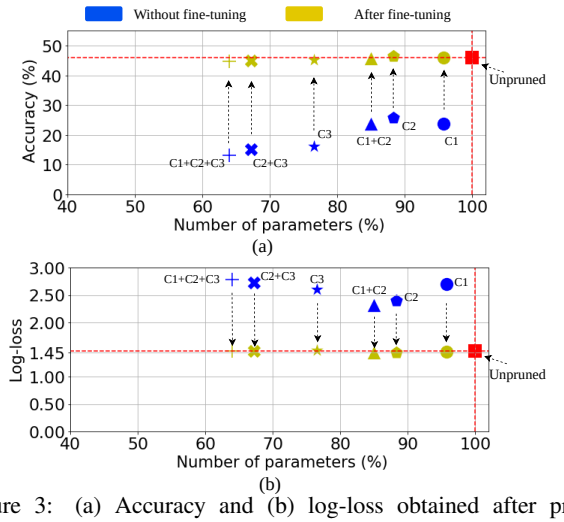


Figure 3: (a) Accuracy and (b) log-loss obtained after pruning different intermediate layers (C1, C2, C3, C1+C2, C2+C3, C1+C2+C3) in the unpruned low-complexity optimal CNN for DCASE 2022 Task 1 development validation dataset. The accuracy and the log-loss is obtained with and without performing the fine-tuning of the pruned network.

Next, we analyse the performance obtained using the low-complexity optimal CNN without performing any pruning and after pruning it. Figure 3 shows the accuracy and the log-loss obtained for the unpruned low-complexity optimal network and its various pruned networks. The unpruned low-complexity optimal network gives 1.475 log-loss and 45.92% accuracy. Eliminating filters from the unpruned network results in a significant reduction in performance, but this is almost entirely restored after fine-tuning.

The performance obtained after aggregating predictions from various pruned networks is given in Table 3. The ensemble framework improves the performance in comparison to that of individual pruned networks.

Performance comparisons: The proposed ensemble framework improves performance as compared to the DCASE 2022 Task 1 baseline network for development and evaluation datasets as given in Table 4. We also compare the performance with the following methods,

- **GMM:** We train Gaussian Mixture Models (GMMs) for each scene class separately, and perform classification using maximum likelihood estimates. The log-mel spectrogram of size (40 x 51) is averaged along temporal dimension to yield (40 x 1) vector, which is given as an input to train the GMMs. The number of mixtures (n) per class are chosen from {5,10,15,20,30,50}.
- **RF:** A random forest (RF) classifier is trained using the log-mel spectrogram averaged along the temporal dimension. The number of estimators are set to 100 and the depth (d) is chosen from {5,20,40,50,100,200}.
- **Dictionary learning**: A dictionary learning framework with structured incoherence and shared features (DLSI) [16] is trained. DLSI framework learns the dictionary for each class by minimizing the reconstruction error and reduces the redundant dictionary atoms in the learning process itself. The number of dictionary atoms per class (k) are chosen from {5,10,15,20,40}.
- **MLP**: A multi-layer perceptron (MLP) network with 2 dense layers each having 100 and 50 units is trained on averaged log-mel spectrogram along the temporal dimension.
- **Pre-trained CNN**: We use pre-trained convolutional layers of VGGish [17, 18] followed by a dense and a classification layer to yield an end-to-end VGGish network. The input to the end-to-end VGGish is log-mel spectrogram of size (40 x 51). We train the end-to-end VGGish for 1000 epochs with similar training settings as used to train the low-complexity CNNs.

The proposed ensemble framework outperforms the other methods as shown in Figure 4. It is interesting to note that the proposed low-complexity optimal CNN performs better than that of the large-scale pre-trained CNN which has 4.5M parameters and has 1077M MACs.

Similarity analysis among various pruned models: We analyse similarity between the different pruned models as given in Table 2. For this, we use 10k audio examples from the validation dataset, and generated outputs from the various filters (feature maps). The similarity is measured using the aggregated mean square error (MSE) across 10k examples, computed between the corresponding feature maps of two different pruned models.

We find that a few of the feature maps generated by the two different pruned models are similar with MSE $\leq 10^{-4}$. To show

Table 3: Performance obtained for DCASE 2022 Task 1 development validation dataset using the ensemble framework that combines various low-complexity optimal CNNs obtained after pruning and quantization.

Sr No.	Ensemble framework	Number of parameters	MACs (millions)	Size (KB)	Accuracy (%)	Log-loss
1	All pruned networks except Pruned_C1	56712	19.72	75.09	47.14	1.394
2	All pruned networks except Pruned_C2	57828	19.75	76.10	47.10	1.396
3	All pruned networks except Pruned_C3	59570	18.60	77.84	47.26	1.392
4	All pruned networks except Pruned_C12	58316	20.70	76.69	47.45	1.394
5	All pruned networks except Pruned_C23	60958	19.84	79.22	47.52	1.389
6	All pruned networks except Pruned_C123	61446	20.80	79.81	47.35	1.392
7	Ensemble on all pruned networks	70966	23.84	92.95	47.45	1.389

Table 4: Performance comparison

Framework	Dataset			
	Development		Evaluation	
	Accuracy (%)	log-loss	Accuracy (%)	log-loss
DCASE baseline network [12]	42.9	1.575	44.2	1.532
Proposed ensemble (#3 in Table 3)	47.26	1.392	45.9	1.492

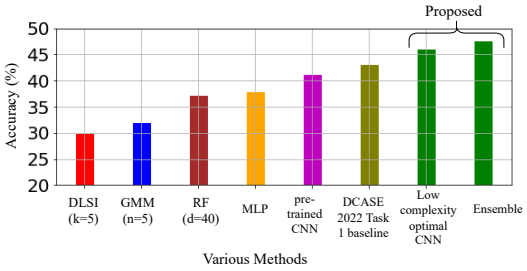


Figure 4: Accuracy comparison of various methods for DCASE 2022 Task 1 development validation set. Here, we show the best accuracy obtained from the DLSI, GMM and RF framework.

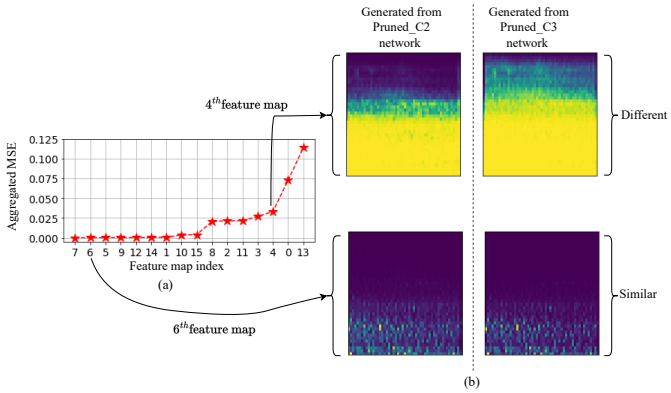


Figure 5: (a) Aggregated MSE computed between the feature maps generated using Pruned_C2 and Pruned_C3 network from the C1 layer. (b) shows the different and the similar feature maps corresponding to 4^{th} and 6^{th} feature map index.

this, the aggregated MSE computed across corresponding feature maps generated by the Pruned_C2 and the Pruned_C3 network is plotted in Figure 5(a). A few of the feature maps have MSE close to zero, and hence these feature maps are similar. We also show the different and the similar feature maps obtained from the Pruned_C2 network and the Pruned_C3 network corresponding to 4^{th} and 6^{th} index for a given input in Figure 5(b). This suggests that similar feature maps across different pruned networks are redundant, and are not required to compute again for the other network.

We find that the Pruned_C1 network shares 7 feature maps with the Pruned_C12 network, and 3 feature maps with the Pruned_C123 network across C1 layer. Similarly, the Pruned_C2 network shares 7 feature maps with the Pruned_C3, network and 6 feature maps with the Pruned_C23.

Ignoring similar feature maps across different pruned models, the MACs could be further reduced by 6M points, and the number of total parameters are reduced by approximately 3.4k in the ensemble framework that combines all pruned networks.

5. DISCUSSION

We observe that small size network obtained after pruning a relatively large size network gives better accuracy compared to that of the similar size network obtained from scratch. For example, the Pruned_C123 network having 9.6k parameters and (3 x 3) filter size gives approximately 16 percentage points more accuracy compared to that of the similar size low-complexity CNN having (1 x 1) filter size and 8.6k parameters as shown in Figure 1.

We find that the ensemble on various CNNs improves the performance as compared to the individual CNN. However, the ensemble on the various CNNs consumes more resources. Therefore, pruning individual CNNs provide an advantage to use the ensemble framework efficiently. To improve the efficiency of the ensemble further, the shared feature maps (similarity) across various CNNs can be ignored.

6. CONCLUSION

This paper focuses on designing a low-complexity system for acoustic scene classification. A filter pruning and quantization is applied to obtain compressed, accelerated, and low-size CNN. Further, various low-size CNNs are combined in the ensemble framework to improve classification performance. The proposed framework shows promising results in terms of reduction in parameters and improved performance. In future, our aim is to improve the performance of the low-complexity framework further.

7. ACKNOWLEDGEMENTS

This work was partly supported by Engineering and Physical Sciences Research Council (EPSRC) Grant EP/T019751/1 "AI for Sound (AI4S)". For the purpose of open access, the authors have applied a Creative Commons Attribution (CC BY) licence to any Author Accepted Manuscript version arising. Thanks to James A. King and Xubo Liu, PhD students at University of Surrey, UK for the discussions.

8. REFERENCES

[1] J. Gu, Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, T. Liu, X. Wang, G. Wang, J. Cai, *et al.*, "Recent advances in convolutional neural networks," *Pattern Recognition*, vol. 77, pp. 354–377, 2018.

[2] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *Proceedings of International Conference on Learning Representations, ICLR (arXiv preprint arXiv:1409.1556)*, 2015.

[3] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems*, pp. 1097–1105, 2012.

[4] Y. Wen and D. Gregg, "Exploiting weight redundancy in CNNs: Beyond pruning and quantization," *arXiv preprint arXiv:2006.11967*, 2020.

[5] A. Singh, P. Rajan, and A. Bhavsar, "SVD-based redundancy removal in 1-D CNNs for acoustic scene classification," *Pattern Recognition Letters*, vol. 131, pp. 383–389, 2020.

[6] H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. P. Graf, "Pruning filters for efficient ConvNets," *Proceedings of International Conference on Learning Representations, ICLR*, 2017.

[7] Z. Liu, J. Li, Z. Shen, G. Huang, S. Yan, and C. Zhang, "Learning efficient convolutional networks through network slimming," *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2736–2744, 2017.

[8] Y. He, X. Zhang, and J. Sun, "Channel pruning for accelerating very deep neural networks," *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1389–1397, 2017.

[9] Y. He, P. Liu, Z. Wang, Z. Hu, and Y. Yang, "Filter pruning via geometric median for deep convolutional neural networks acceleration," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4340–4349, 2019.

[10] M. Lin, R. Ji, Y. Wang, Y. Zhang, B. Zhang, Y. Tian, and L. Shao, "HRank: Filter pruning using high-rank feature map," *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1529–1538, 2020.

[11] B. Kim, S. Yang, J. Kim, and S. Chang, "QTI submission to DCASE 2021: Residual normalization for device imbalanced acoustic scene classification with efficient design," *DCASE2021 Challenge, Tech. Rep*, 2021.

[12] I. Martín-Morató, F. Paissan, A. Ancilotto, T. Heittola, A. Mesaros, E. Farella, A. Brutti, and T. Virtanen, "Low-complexity acoustic scene classification in DCASE 2022 Challenge," *arXiv preprint arXiv:2206.03835*, 2022.

[13] T. Heittola, A. Mesaros, and T. Virtanen, "Acoustic scene classification in DCASE 2020 Challenge: Generalization across devices and low complexity solutions," *Proceedings of the Detection and Classification of Acoustic Scenes and Events (DCASE) Workshop*, 2020.

[14] A. Singh and M. D. Plumbley, "A passive similarity based CNN filter pruning for efficient acoustic scene classification," *Interspeech (arXiv preprint arXiv:2203.15751)*, 2022.

[15] R. David, J. Duke, A. Jain, V. Janapa Reddi, N. Jeffries, J. Li, N. Kreeger, I. Nappier, M. Natraj, T. Wang, *et al.*, "TensorFlow Lite Micro: Embedded machine learning for TinyML systems," *Proceedings of Machine Learning and Systems*, vol. 3, pp. 800–811, 2021.

[16] I. Ramirez, P. Sprechmann, and G. Sapiro, "Classification and clustering via dictionary learning with structured incoherence and shared features," *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 3501–3508, 2010.

[17] J. Cramer, H.-H. Wu, J. Salamon, and J. P. Bello, "Look, listen, and learn more: Design choices for deep audio embeddings," *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3852–3856, 2019.

[18] S. Hershey, S. Chaudhuri, D. P. W. Ellis, J. F. Gemmeke, A. Jansen, C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold, M. Slaney, R. Weiss, and K. Wilson, "CNN architectures for large-scale audio classification," *Proceedings of International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 131–135, 2017.

IMPROVED DOMAIN GENERALIZATION VIA DISENTANGLED MULTI-TASK LEARNING IN UNSUPERVISED ANOMALOUS SOUND DETECTION

Satvik Venkatesh[1,2]*, Gordon Wichern*[1]*, Aswin Subramanian*[1]*, Jonathan Le Roux*[1]

[1]Mitsubishi Electric Research Laboratories (MERL), Cambridge, MA, USA
[2]Interdisciplinary Centre for Computer Music Research, University of Plymouth, UK
satvik.venkatesh@plymouth.ac.uk,{wichern,subramanian,leroux}@merl.com

ABSTRACT

We investigate a novel multi-task learning framework that disentangles domain-shared features and domain-specific features for domain generalization in anomalous sound detection. Disentanglement leads to better latent features and also increases flexibility in post-processing due to the availability of multiple embedding spaces. The framework was at the core of our submissions to the DCASE2022 Challenge Task 2. We ranked 5th out of 32 teams in the competition, obtaining an overall harmonic mean of 67.57% on the blind evaluation set, surpassing the baseline by 13.5% and trailing the top rank by 3.4%. We also explored machine-specific loss functions and domain generalization methods, which showed improvements on the development set, but were less effective on the evaluation set.

***Index Terms*—** Anomaly detection, disentanglement, multi-task learning, domain generalization, representation learning

1. INTRODUCTION

Machine condition monitoring using acoustic sensors is an important topic for industry with applications such as factory automation and predictive maintenance. Automatic detection of anomalous sounds is a particularly important application, however, all possible types of anomalous sounds may not be known in advance, and purposefully damaging machinery to collect anomalous sound recordings is undesirable. Thus, there has been much recent research interest in the field of unsupervised anomalous sound detection, where only data collected under normal operating conditions is available for training machine learning models.

Much of the recent progress in unsupervised anomalous sound detection has been driven by DCASE challenges on the topic [1–3]. Typical approaches include those based on autoencoder-like architectures [4–10], where a model trained only on normal data to reconstruct its input should exhibit large reconstruction error when presented with an anomalous example at inference time. Another class of approaches, which we refer to as surrogate task models, use an alternative supervised training task to learn a model of normality, and then measure deviations from normal to predict anomalies. Example surrogate tasks include outlier exposure [6, 11], predicting metadata (e.g., machine instance) or attributes (e.g., operating load) [12–14], and learning to predict what augmentations (e.g., time-stretching or pitch-shifting) were applied to an audio clip [15].

As in many areas where deep learning-based models have become the predominant approach, unsupervised anomalous sound detection suffers from issues related to robustness. To better tackle such issues, the anomalous sound detection tasks of both the 2021 and 2022 DCASE challenges focused on performance under domain shift, where acoustic conditions differ based on environmental background noise or other machine operating conditions. The goal is to develop methods that should perform equally well in a source domain, where most of the (normal) training data comes from, and in a target domain, where only a few normal examples are available. The 2021 challenge task [2] assumed the domain (source or target) of the audio sample was known at inference time (a configuration referred to as domain adaptation), while the 2022 task [3] assumes the domain is unavailable at inference time (a configuration referred to as domain generalization).

While many well-known techniques exist for domain generalization (see [16] for an overview), we focus our efforts on disentangled representation learning [17], where subsets of learned feature dimensions correspond to specific factors in the dataset. Disentanglement has been successfully applied for music information retrieval in the audio domain [18] and in approaches to domain adaptation for image classification [19]. Specifically, we consider learning feature representations for each normal sound example in the training set, where subsets of features are learned using different surrogate tasks. In the case of the DCASE 2022 Task 2 dataset, we learn a subset of *domain-shared* features, whose surrogate task is to predict the section index regardless of domain (each section is dedicated to a specific type of domain shift, with other conditions being shared across domains), and subsets of *domain-specific* features each associated with a surrogate task consisting of predicting a particular machine attribute (e.g., specific states or environmental conditions of the machine), which are typically different across domains and sections. We demonstrate experimentally that our disentangled model performs better than a multi-task learning model where features are not disentangled, and further show that by weighting individual anomaly scores computed over different disentangled dimensions, we obtain an ensemble-like system using a single model. Furthermore, by examining the anomaly score in specific disentangled dimensions, we can better understand which attribute may have caused the anomaly, improving the explainability of deep learning models.

As discussed in our challenge report [20], we also explored machine-specific variations to the loss function. While they led to significant improvements on the dev set, they ultimately performed worse on the eval set. This paper thus focuses on our best performing disentangled models.

This work was performed while S. Venkatesh was an intern at MERL.

2. DISENTANGLED ANOMALY DETECTOR

In this paper, we investigate an approach that disentangles a learned latent representation into domain-shared and domain-specific features for domain generalization in anomalous sound detection, as illustrated in Fig. 1. In particular, we refer to sections as domain-shared features and to attributes as domain-specific features. For example, in Fan's *section 00*, machine noises occurring in the source domain are of type W and X, while those occurring in the target domain are of type Y and Z. Therefore, *section 00* is common to both domains but the machine noises are different across domains.

2.1. Surrogate Task Training

During training, we have a dataset of N normal training examples for a given machine type, $\mathcal{D} = \{(X^{(n)}, y^{(n)})\}_{n=1}^{N}$, where $X \in \mathbb{R}^{F\times T}$ is a magnitude spectrogram with F frequencies and T time frames, and $y = [y_s, y_{a_1}, \ldots, y_{a_M}] \in \mathbb{N}^{M+1}$ is a vector of categorical surrogate task labels, where y_s represents machine section and y_{a_m} represents the categorical label of the m-th attribute among the M different attributes available for the given machine type. We obtain a domain-shared (section) embedding z_S and a domain-specific (attribute) embedding z_A as:

$$z_S = \Phi^{\text{Sec}}[\text{CNN}(X)] \in \mathbb{R}^{D_S}, \quad z_A = \Phi^{\text{Att}}[\text{CNN}(X)] \in \mathbb{R}^{D_A} \tag{1}$$

where $\text{CNN}(\cdot)$ is a shared convolutional neural network, while Φ^{Sec} and Φ^{Att} represent section and attribute specific linear embedding layers, respectively (implemented as 1×1 convolutions). All parameters are trained by minimizing $\mathcal{L} = \mathcal{L}^{\text{Sec}} + \mathcal{L}^{\text{Att}}$, where

$$\mathcal{L}^{\text{Sec}} = \log \frac{\exp(w_{0,y_s} \cdot z_S + b_{0,y_s})}{\sum_{c=1}^{C} \exp(w_{0,c} \cdot z_S + b_{0,c})}, \tag{2}$$

$$\mathcal{L}^{\text{Att}} = \sum_{m=1}^{M} \log \frac{\exp(w_{m,y_m} \cdot z_A + b_{m,y_m})}{\sum_{c_m=1}^{C_m} \exp(w_{m,c_m} \cdot z_A + b_{m,c_m})} \tag{3}$$

are the cross-entropy losses for section and attributes, respectively, $w_{i,j}$ and $b_{i,j}$ are learned weight vectors and biases of the associated classifiers, c indexes the $C = 6$ sections and c_m indexes the C_m values of the m-th attribute. Because not all attributes are present among all audio examples of a given machine type in the DCASE 2022 Task 2 dataset, the attribute loss in (3) is combined over all attributes in a multi-task learning fashion from the same embedding z_A, rather than learning disentangled feature dimensions for each attribute. If an attribute is unknown for an audio example, the corresponding term in the sum of (3) is ignored.

We note that our formulation of attribute learning in (3) as a multi-task learning problem with a different objective for each attribute differs from [21] where every possible combination of section and attribute corresponded to a different class.

2.2. Inference Approaches

The nearest neighbor (NN) algorithm is a simple and effective approach for anomaly detection [22, 23] given feature vectors of normal samples. As illustrated in Fig. 1, during inference we use the NN distance between a test embedding z_q and all corresponding training set embeddings $z_q^{(j)}$ for computing an anomaly score, i.e.,

$$D_{\text{NN}}(z_q, \mathcal{D}) = \min_{j\in\mathcal{D}} D_{\cos}(z_q, z_q^{(j)}), \tag{4}$$

where $D_{\cos}(\cdot,\cdot)$ is the cosine distance between two embedding vectors. The disentangled model allows us to explore multiple inference approaches depending on which embedding dimensions we use for z_q in (4) as discussed below.

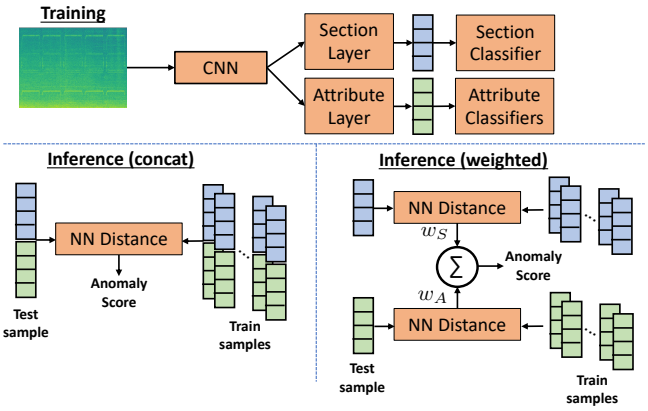


Figure 1: Block diagram of disentangled anomaly detector. In the figure, NN stands for Nearest Neighbor. In the training phase, exclusive latent spaces were assigned to sections and attributes.

Disentangled Concatenated: Use the concatenated embedding $z_C = [z_S^T, z_A^T]^T$ in (4) as shown in the bottom-left of Fig. 1.

Disentangled Weighted: As illustrated in the bottom-right of Fig. 1, we take a weighted average of NN distances separately computed for section embedding z_S and attribute embedding z_A, i.e.,

$$D_{\text{NN}}^{\text{wt}}(z_S, z_A, \mathcal{D}) = w_S D_{\text{NN}}(z_S, \mathcal{D}) + w_A D_{\text{NN}}(z_A, \mathcal{D}) \tag{5}$$

where w_S and w_A are scalar weights, which are optimized after training is complete based on dev set performance. The best weights for each machine are shown in Table 4.

Disentangled Sections: Use only section embedding z_S in (4).

Disentangled Attributes: Use only attribute embedding z_A in (4).

At test time, the section label of the test sample is known, therefore, we limit the training set samples from $\mathcal{D}$ when computing the NN distance to be only those samples belonging to the appropriate section. Furthermore, our CNN architecture, detailed in Section 3.3, operates on spectrogram chunks of $T = 32$ time frames ($\sim$1 s), while each test sample is 10 s long. Using a chunk hop size of one frame, we obtain 282 embedding vectors per 10 s audio file. Following [22], we merged the embedding vectors for each sample by calculating their mean, except for valve where merging based on standard deviation provided significant gains. We then use the merged embedding vectors for computing the anomaly score.

3. EXPERIMENTAL SETUP

3.1. Dataset

There are seven different machine types in the DCASE 2022 Task 2 dataset [3] — ToyCar, ToyTrain, Bearing, Fan, Gearbox, Slider, and Valve. ToyCar and ToyTrain are from the ToyADMOS2 dataset [24], and the five other machines are from the MIMII DG dataset [25]. The data under each machine type is divided into sections, each of which corresponds to a specific type of domain shift. For example, in Fan, *section 00* refers to different machine noise between source and target domains, while *section 01* refers to different factory noise.

For each audio file, information about its section as well as one or more attributes is given. For machines belonging to the MIMII

DG dataset [3], only information on the domain shifting attribute, such as the type of machine noise in Fan's *section 00* and the type of factory noise in Fan's *section 01*, was present. For ToyCar and ToyTrain, which belong to the ToyADMOS2 dataset [24], information on all attributes was present in the filenames, even for those attributes that are not the domain shifting one. For the multi-task attribute learning (3), we make use of all present attributes, and represent them as categorical variables using all possible values found in the training set.

3.2. Audio Features and Training Strategy

The dataset contained 10 s audio files at a sampling rate of 16 kHz. We adopted short-time Fourier transform magnitude spectrograms as features for the neural network. The hop size was set to 32 ms and the window size was 128 ms (2048 samples). While training the neural network, the number of time steps for each audio example was 32 frames. Therefore, the input shape for the network was 1025×32. One epoch is defined as training the network on all 6000 audio files (six sections with 1000 examples in each section). For each audio file, a random chunk of 32 frames is selected for training. The advantages of this technique were reduced RAM usage, less chance of overfitting within epochs, and improved generalization compared with the baseline.

We adopted the Adam optimizer using a batch size of 32. In most cases, the learning rate was set to 10^{-4}. For ToyCar, we found a minor improvement by setting it to 10^{-5}. We trained the models for a maximum of 300 epochs, saved the model's weights every 5 epochs, tested the anomaly detector's performance on the development set, and selected the best performing model for each machine. We were unable to observe a clear relationship between the performances on the surrogate task and detection of anomalies. For instance, an improvement in the classification accuracy of sections (the surrogate task) was not necessarily accompanied by an improvement in anomaly detection. A similar observation has been made by previous studies using autoencoder-based models [10].

3.3. Neural Network Architecture

Morita et al. [22] found that the MobileFaceNet architecture [26] performed better than MobileNetV2 [27] as a feature extractor. We observed a similar improvement in initial experiments, and hence adopted MobileFaceNet. The parameter settings for MobileFaceNet can be found in Table 1. The output of the global depth-wise convolution (GDC) layer is a 512-D embedding vector. This is connected to the linear embedding layers (i.e., 1x1 convolutions) L^{Sec} and L^{Att} defined in Section 2, and associated softmax classification layers. Additionally, we explore minor modifications to the embedding and softmax layers as explained in Section 3.5.

3.4. Evaluation Metrics

We evaluate our models independently for each section and machine type using the three official metrics [3]: area under the ROC curve in the source (AUC (S)) and target (AUC (T)) domains, where the normal test samples are compared against anomalies from both domains, along with the domain agnostic partial AUC (pAUC) computed under low false-alarm-rate conditions.

For threshold-dependent metrics, we followed a similar approach to the baseline [3] and assumed the scores follow a gamma distribution. The parameters of the gamma distribution are estimated from the NN anomaly scores computed on the training set samples independently for each section (excluding self neighbors). For five machines, we set the anomaly detection threshold as the 90th percentile of the gamma distribution. For Fan and Bearing, we observed low sensitivity and hence adopted 60th percentile.

Table 1: MobileFaceNet [26] architecture, where all convolutions are 2D and dw-Conv refers to depth-wise convolution. In the network, Linear Conv 1×1 (sec) is connected to Softmax (sec), and Linear Conv 1×1 (att) is connected to the other softmax layers for attributes. For each layer, we show the expansion factor (**t**), number of channels (**c**), number of repeats (**n**), and stride (**s**). All convolutions excluding the final linear layers use PReLU activations.

Input	Operator	t	c	n	s
1x32x1025	Conv 3x3	-	64	1	2
64x16x513	dw-Conv 3x3	-	64	1	1
64x16x513	Bottleneck	2	64	5	2
64x8x257	Bottleneck	4	128	1	2
128x4x129	Bottleneck	2	128	6	2
128x2x65	Bottleneck	4	128	1	2
128x1x33	Bottleneck	2	128	2	1
128x1x33	Conv 1x1	-	512	1	1
512x1x33	Linear GDC 1x33	-	512	1	1
512x1x1	Linear Conv 1x1 (sec)	-	128	1	1
512x1x1	Linear Conv 1x1 (att)	-	128	1	1
128x1x1	Softmax (sec)	-	6	-	-
128x1x1	Softmax (att_1)	-	C_1	-	-
⋮	⋮		⋮		
128x1x1	Softmax (att_M)	-	C_M	-	-

3.5. Other Models Considered

ArcFace [29] was shown to improve class separability by adding angular margin to the loss. We investigated this technique's advantage by training on section indices. The feature scale and margin parameters were set to 32 and 0.5 respectively. We found ArcFace did not work well in a multi-task learning setting, probably because all attributes were not present in every example.

Conventional Multi-task Learning (MTL): In this framework, the GDC layer from Table 1 is connected to a single 2D convolutional 1×1 layer with 256 channels. In other words, the features are in an entangled latent space.

Machine-Specific Loss (MSL): Although not the focus of this paper, we also compare performance against the other two systems we submitted to the challenge: (S1) MSL as described in [20] and (S2) an ensemble of MSL and the attentive neural process (ANP) approach in [10] as detailed in Table 4.

4. RESULTS

Development Set results are shown in Table 2. Training using only section labels obtains an overall harmonic mean of 72.82%, which is significantly higher than both the baselines. This improvement is attributed to adopting the Nearest Neighbor algorithm during post-processing [22] and to our new training strategy explained in Section 3.2. Adopting ArcFace, which is essentially training on section indices with additive angular margin, improved the overall performance to 73.62%, while the AUC(T) improved from 67.34% to 71.79%. MTL, which trains on sections and attributes, obtained a lower overall performance of 72.47%, but improved the AUC(T) to 70.72%. Note that the MTL model does not use ArcFace. The Disentangled Sections (Disent_Sec) model only considers the section

Table 2: Results of different models on the development test set. We merge the three metrics and all sections to obtain a single number per machine using the harmonic mean. We also report the harmonic mean across machines and sections for each of the three metrics.

System	ToyCar	ToyTrain	Bearing	Fan	Gearbox	Slider	Valve	AUC (S)	AUC (T)	pAUC	Overall
MSL+ANP (S2)	76.43	**59.96**	**73.93**	**68.89**	**85.37**	**85.93**	**95.83**	**87.55**	**73.43**	**70.36**	**76.43**
MSL (S1)	76.43	59.37	73.93	68.85	83.03	85.37	95.63	86.78	73.34	69.68	75.93
Disent_Wt (S4)	**76.95**	59.74	72.07	63.91	81.38	85.14	94.50	86.09	71.65	68.21	74.57
Disent_Cat (S3)	76.43	58.67	67.09	63.18	80.99	85.37	95.01	84.61	70.59	67.02	73.34
Disent_Sec	76.84	56.64	72.07	62.35	81.04	84.84	94.42	86.43	68.64	68.01	73.45
Disent_Att	75.26	59.74	60.82	63.02	78.86	78.88	92.72	82.12	69.31	64.09	71.08
MTL	75.61	59.37	68.24	59.14	80.63	83.51	94.49	81.35	70.72	66.83	72.47
Sec_ArcFace	72.31	58.09	71.30	68.85	79.37	82.50	92.87	86.50	71.19	66.04	73.62
Sec_Softmax	76.20	52.85	**73.93**	64.39	81.43	85.89	90.11	86.05	67.34	67.92	72.82
ANP-Boot	59.84	50.87	55.54	55.31	64.38	64.11	52.63	69.26	50.87	54.24	57.10
AE Baseline	51.06	39.61	54.80	58.54	63.07	57.99	50.59	68.74	41.91	53.76	52.62
MN Baseline	54.23	51.18	59.16	57.21	59.91	50.26	62.42	63.87	50.14	55.69	56.01

Table 3: Official results of different models on the evaluation test set. We were unable to present the results in the same format as Table 2 because we do not have access to all the scores.

System	ToyCar		ToyTrain		Fan		Gearbox		Bearing		Slider		Valve		Overall
	AUC	pAUC	AUC	pAUC	AUC	pAUC	AUC	pAUC	AUC	pAUC	AUC	pAUC	AUC	pAUC	
Top rank [28]	88.45	**81.83**	**70.46**	**61.14**	**57.34**	**57.33**	86.04	64.22	68.85	54.45	**78.26**	**66.39**	**83.87**	**75.22**	**70.97**
Disent_Cat (S3)	**93.88**	78.67	58.23	54.73	48.17	50.34	**86.76**	**79.43**	**72.54**	61.86	73.64	60.70	83.72	62.93	67.57
Disent_Wt (S4)	93.30	75.47	57.30	54.93	46.93	50.33	86.34	78.47	71.96	**64.26**	75.94	64.29	83.05	64.01	67.49
MSL (S1)	**93.88**	78.67	55.53	54.33	44.50	50.84	86.47	68.54	69.94	61.64	73.64	60.70	78.51	66.08	65.66
MSL+ANP (S2)	**93.88**	78.67	54.92	54.22	44.29	50.97	82.37	70.76	69.94	61.64	75.96	62.40	77.69	65.39	65.57
MN Baseline	42.79	53.44	51.22	50.98	50.34	55.22	51.34	48.49	58.23	52.16	62.42	53.07	72.77	65.16	54.02
AE Baseline	61.18	60.21	43.14	49.36	41.16	50.12	61.92	51.95	59.93	53.95	58.95	54.16	54.26	51.30	52.94

Table 4: Detailed model setups. We indicate the best MSL for S1. The ensemble weights (Ens. wt.) of S2 and the disentanglement weights (Disent. wt.) of S4 were calculated via a grid search.

		Ens. wt. (S2)		Disent. wt. (S4)	
Machine	**MSL (S1)**	**MSL**	**ANP**	w_S	w_A
ToyCar	Disent_Cat	0.60	0.40	0.90	0.10
ToyTrain	MTL	0.70	0.30	0.00	1.00
Bearing	Sections only	1.00	0.00	1.00	0.00
Fan	ArcFace	0.95	0.05	0.15	0.85
Gearbox	Adversarial	0.65	0.35	0.80	0.20
Slider	Disent_Cat	0.70	0.30	0.90	0.10
Valve	Disent_Split	0.80	0.20	0.90	0.10

embeddings during inference and obtains an overall performance of 73.45%. Although the overall performance is lower than that of ArcFace, it showed improvements for all machines except Bearing and Fan. The Disentangled Weighted (Disent_Wt) model obtains the highest overall performance for a single model without ensembling and machine-specific losses. The MSL and MSL+ANP systems showed improvements on the development set, but, as discussed next, worsened performance on the evaluation set.

Evaluation Set results on the DCASE challenge blind test set are shown in Table 3. Interestingly, both of our disentanglement models performed better than MSL and MSL+ANP. As indicated in Table 4, the MSL system used multi-task learning for ToyTrain, only sections for Bearing, ArcFace for Fan, adversarial training for Gearbox, and a slightly modified version of disentanglement for valve, where every attribute was assigned a separate embedding space. In all cases, our disentangled models outperformed the MSL systems. These observations convey that disentanglement is an effective technique for domain generalization.

However, optimizing disentanglement weights (Disent_Wt) on the dev set did not lead to improved performance over the simple concatenation approach (Disent_Cat). One hypothesis is that the optimized weights turned out to be slightly dataset-specific, which hurt generalization performance.

We ranked 5th out of 32 teams in the competition, obtaining an overall harmonic mean of 67.57%. We surpassed the baseline by 13.5% and trailed the top rank [28] by 3.4%. We surpassed the top rank for Bearing, Gearbox, and ToyCar(AUC). We believe that [28] adopted a better pipeline to train the feature extractor — (1) they average model weights from multiple epochs (2) they pre-train on all 7 machines and fine-tune for each machine. We hypothesize that our disentangled model can also be improved by incorporating these optimizations in the training pipeline.

5. CONCLUSION

In this study, we presented a disentangled multi-task learning framework for improved domain generalization in anomalous sound detection. We demonstrated that the disentangled model performs better than simple multi-task learning, or only learning based on domain-shared features (e.g., section indices). We also showed that there is increased flexibility in post-processing due to the multiple disentangled embedding spaces. In addition to the NN algorithm, we plan to explore other anomaly detection backends in the future, and to thoroughly evaluate the explainability of our approach.

6. REFERENCES

[1] Y. Koizumi, Y. Kawaguchi, K. Imoto, T. Nakamura, Y. Nikaido, R. Tanabe, H. Purohit, K. Suefusa, T. Endo, M. Yasuda, and N. Harada, "Description and discussion on DCASE2020 challenge task2: Unsupervised anomalous sound detection for machine condition monitoring," in *Proc. DCASE*, Nov. 2020, pp. 81–85.

[2] Y. Kawaguchi, K. Imoto, Y. Koizumi, N. Harada, D. Niizumi, K. Dohi, R. Tanabe, H. Purohit, and T. Endo, "Description and discussion on dcase 2021 challenge task 2: Unsupervised anomalous detection for machine condition monitoring under domain shifted conditions," in *Proc. DCASE*, Nov. 2021, pp. 186–190.

[3] K. Dohi, K. Imoto, N. Harada, D. Niizumi, Y. Koizumi, T. Nishida, H. Purohit, T. Endo, M. Yamamoto, and Y. Kawaguchi, "Description and discussion on DCASE 2022 challenge task 2: Unsupervised anomalous sound detection for machine condition monitoring applying domain generalization techniques," *In arXiv e-prints: 2206.05876*, 2022.

[4] E. Marchi, F. Vesperini, F. Eyben, S. Squartini, and B. Schuller, "A novel approach for automatic acoustic novelty detection using a denoising autoencoder with bidirectional LSTM neural networks," in *Proc. ICASSP*, Apr. 2015.

[5] E. Cakır and T. Virtanen, "Convolutional recurrent neural networks for rare sound event detection," in *Proc. DCASE*, Nov. 2017.

[6] Y. Koizumi, S. Saito, H. Uematsu, Y. Kawachi, and N. Harada, "Unsupervised detection of anomalous sound based on deep learning and the neyman–pearson lemma," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 27, no. 1, pp. 212–224, 2018.

[7] T. Hayashi, T. Komatsu, R. Kondo, T. Toda, and K. Takeda, "Anomalous sound event detection based on wavenet," in *Proc. EUSIPCO*, Sept. 2018, pp. 2494–2498.

[8] K. Suefusa, T. Nishida, H. Purohit, R. Tanabe, T. Endo, and Y. Kawaguchi, "Anomalous sound detection based on interpolation deep neural network," in *Proc. ICASSP*, May 2020, pp. 271–275.

[9] R. Giri, F. Cheng, K. Helwani, S. V. Tenneti, U. Isik, and A. Krishnaswamy, "Group masked autoencoder based density estimator for audio anomaly detection," in *Proc. DCASE*, Nov. 2020, pp. 51–55.

[10] G. Wichern, A. Chakrabarty, Z.-Q. Wang, and J. Le Roux, "Anomalous sound detection using attentive neural processes," in *Proc. WASPAA*, 2021, pp. 186–190.

[11] P. Primus, V. Haunschmid, P. Praher, and G. Widmer, "Anomalous sound detection as a simple binary classification problem with careful selection of proxy outlier examples," in *Proc. DCASE*, Nov. 2020, pp. 170–174.

[12] R. Giri, S. V. Tenneti, F. Cheng, K. Helwani, U. Isik, and A. Krishnaswamy, "Self-supervised classification for detecting anomalous sounds," in *Proc. DCASE*, Nov. 2020, pp. 46–50.

[13] J. A. Lopez, H. Lu, P. Lopez-Meyer, L. Nachman, G. Stemmer, and J. Huang, "A speaker recognition approach to anomaly detection," in *Proc. DCASE*, Nov. 2020, pp. 96–99.

[14] K. Wilkinghoff, "Sub-cluster adacos: Learning representations for anomalous sound detection," in *Proc. IJCNN*, 2021, pp. 1–8.

[15] T. Inoue, P. Vinayavekhin, S. Morikuni, S. Wang, T. H. Trong, D. Wood, M. Tatsubori, and R. Tachibana, "Detection of anomalous sounds for machine condition monitoring using classification confidence," in *Proc. DCASE*, Nov. 2020, pp. 66–70.

[16] J. Wang, C. Lan, C. Liu, Y. Ouyang, T. Qin, W. Lu, Y. Chen, W. Zeng, and P. Yu, "Generalizing to unseen domains: A survey on domain generalization," *IEEE Transactions on Knowledge and Data Engineering*, 2022.

[17] A. Veit, S. Belongie, and T. Karaletsos, "Conditional similarity networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 830–838.

[18] J. Lee, N. J. Bryan, J. Salamon, Z. Jin, and J. Nam, "Metric learning vs classification for disentangled music representation learning," in *Proc. ISMIR*, 2020.

[19] Z. Ding and Y. Fu, "Deep domain generalization with structured low-rank constraint," *IEEE Transactions on Image Processing*, vol. 27, no. 1, pp. 304–313, 2017.

[20] S. Venkatesh, G. Wichern, A. Subramanian, and J. Le Roux, "Disentangled surrogate task learning for improved domain generalization in unsupervised anomalous sound detection," DCASE2022 Challenge, Tech. Rep., July 2022.

[21] K. Wilkinghoff, "Utilizing sub-cluster adacos for anomalous sound detection under domain shifted conditions," DCASE2021 Challenge, Tech. Rep., July 2021.

[22] K. Morita, T. Yano, and K. Tran, "Anomalous sound detection using cnn-based features by self supervised learning," DCASE2021 Challenge, Tech. Rep., July 2021.

[23] M. Jones, D. Nikovski, M. Imamura, and T. Hirata, "Exemplar learning for extremely efficient anomaly detection in real-valued time series," *Data mining and knowledge discovery*, vol. 30, no. 6, pp. 1427–1454, 2016.

[24] N. Harada, D. Niizumi, D. Takeuchi, Y. Ohishi, M. Yasuda, and S. Saito, "ToyADMOS2: another dataset of miniature-machine operating sounds for anomalous sound detection under domain shift conditions," *arXiv preprint arXiv:2106.02369*, 2021.

[25] K. Dohi, T. Nishida, H. Purohit, R. Tanabe, T. Endo, M. Yamamoto, Y. Nikaido, and Y. Kawaguchi, "Mimii dg: Sound dataset for malfunctioning industrial machine investigation and inspection for domain generalization task," *arXiv preprint arXiv:2205.13879*, 2022.

[26] S. Chen, Y. Liu, X. Gao, and Z. Han, "Mobilefacenets: Efficient cnns for accurate real-time face verification on mobile devices," in *Chinese Conference on Biometric Recognition*. Springer, 2018, pp. 428–438.

[27] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4510–4520.

[28] Y. Zeng, H. Liu, L. Xu, Y. Zhou, and L. Gan, "Robust anomaly sound detection framework for machine condition monitoring," DCASE2022 Challenge, Tech. Rep., July 2022.

[29] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, "Arcface: Additive angular margin loss for deep face recognition," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 4690–4699.

DETECT WHAT YOU WANT: TARGET SOUND DETECTION

Helin Wang[1†], *Dongchao Yang*[1†], *Yuexian Zou*[1*], *Fan Cui*[2], *Yujun Wang*[2]

[1]ADSPLAB, School of ECE, Peking University, Shenzhen, China
[2] Xiaomi Corporation, Beijing, China

ABSTRACT

Human beings can perceive a target sound type from a multi-source mixture signal by the selective auditory attention, however, such functionality was hardly ever explored in machine hearing. This paper addresses the target sound detection (TSD) task, which aims to detect the target sound signal from a mixture audio when a target sound's reference audio is given. We present a novel target sound detection network (TSDNet) which consists of two main parts: A conditional network which aims at generating a sound-discriminative conditional embedding vector representing the target sound, and a detection network which takes both the mixture audio and the conditional embedding vector as inputs and produces the detection result of the target sound. These two networks can be jointly optimized with a multi-task learning approach to further improve the performance. In addition, we study both strong-supervised and weakly-supervised strategies to train TSDNet and propose a data augmentation method by mixing two samples. To facilitate this research, we build a target sound detection dataset (*i.e.* URBAN-TSD) based on URBAN-SED and UrbanSound8K datasets, and experimental results indicate our method could get the segment-based F scores of 76.3% and 56.8% on the strongly-labelled and weakly-labelled data respectively.

Index Terms— target sound detection, conditional embedding, weakly supervised, data augmentation

1. INTRODUCTION

Human beings has the ability to focus auditory attention on a particular sound in a multi-source environment, however, there were few studies in this area for machine hearing. In this paper, we initially define a target sound detection (TSD) task, which aims to recognize and localize target sound source within a mixture audio given a reference audio or/and a sound label. For example, the violin sound can be detected within a concert recording and the talking sound can be detected in a noisy cafe environment. TSD has a lots of potential applications, such as noise monitoring for smart cities [1], bioacoustic species and migration monitoring [2] and large-scale multimedia indexing [3]. To the best of our knowledge, this paper is the first attempt that explicitly tackles this problem.

There is one similar task with TSD, *i.e.* sound event detection (SED). SED aims to classify and localize all pre-defined sound events (*e.g.*, train horn, car alarm) within an audio clip, which has been widely studied [4, 5, 6, 7]. Compared to SED, TSD only focuses on detecting the event that we care about and ignores other events. Furthermore, TSD does not require to pre-define categories set, so it can be easy to extend to open domain detection. Other related tasks about extracting the target signal are speaker extraction [8, 9] which extracts the target speech from a mixture speech given a reference utterance of the target speaker, and acoustic events sound selection problems [10]. Different from them, our work focuses on the detection task, which is more suitable for many multimedia retrieval applications and the training data is easier to obtain.

To solve the TSD task, we propose a target sound detection network (*i.e.* TSDNet), and treat TSD as a binary classification problem for each frame of the audio, where the positive class is the sound event of interest, and the negative class is formed by the combination of all foreground and background interfering events and noises. By using reference audio, TSDNet can focus on the target sound and ignore other interference. More specifically, TSDNet is composed of a conditional network which is used to generate a sound-discriminative conditional embedding vector from the reference audio, and a detection network which is applied to obtain binary-classification results at each frame with the conditional embedding vector and mixture audio as inputs. In order to get more sound-discriminative conditional embedding vector for TSDNet, the conditional network is jointly optimized with both the TSD task and a sound event classification task. We further explore a data augmentation method for TSD, which randomly mixes two training samples to form a new training sample. In addition, we explore to train the TSDNet on both strong- and weakly-supervised TSD tasks. Here, weakly-supervised TSD task means the dataset only provides the presence or absence of target sound within the mixture sound, but not any timestamp information, which is more challenging.

Our contributions can be summarized as follows: (1) We propose a novel network for TSD that can be trained with both strong-labelled and weakly-labelled data. (2) We explore the jointly training method to get the robust conditional embedding vector, and propose a data augmentation method for the TSD task. (3) We establish a dataset for TSD, and our method achieves the segment-based F measures of 76.3% and 56.8% on the strongly-labelled and weakly-labelled data respectively.

2. PROPOSED METHOD

The architecture of our proposed network (TSDNet) is shown in Figure 1. TSDNet consists of two components: a conditional network which inputs the reference audio and outputs a conditional embedding vector, and a detection network which uses the conditional embedding vector and the mixture audio to get the detection results. In this section, we will describe the details of the whole network.

---

† Indicates equal contribution.

* Corresponding Author: zouyx@pku.edu.cn

This paper was partially supported by Shenzhen Science & Technology Research Program (No: GXWD20201231165807007-20200814115301001; No: JSGG20191129105421211) and NSFC (No: 62176008).

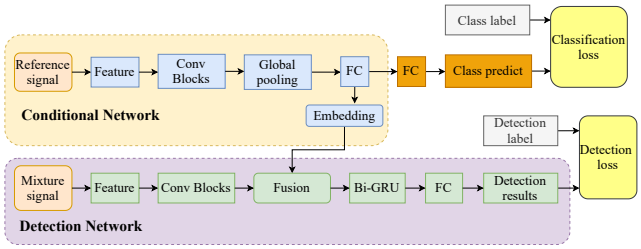


Figure 1: The architecture of our proposed TSDNet. Here, FC denotes the fully-connected layer. Reference signal indicates that the signal from the target class.

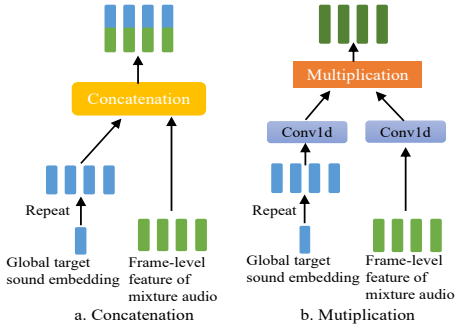


Figure 2: Two fusion modules.

2.1. Conditional Network

The purpose of the conditional network is to produce a global conditional embedding vector to represent the reference information. The input to the conditional network can be either a reference audio or a specific label, or both. In this paper, we focus on a reference audio as the input, for the reason that it is more challenging but easier to be transferred to new classes. Inspired by the powerful ability of extracting time-frequency robust features from audio with convolutional neural networks (CNNs) [11, 12, 13, 14], we apply a VGG-like CNN network for the conditional network, which uses the spectrogram feature as input and consists of 4 convolutional blocks with 64, 128, 256 and 512 output channels, respectively. Each convolutional block contains 2 convolutional layers with kernel size of 3×3, followed by downsampling with average pooling size of 2×2. Batch normalization [15] and ReLU function [16] are applied to all the convolutional layers. Global pooling layer [11] containing a global max-pooling function and a global average-pooling function is applied to get the global feature vector, which is then fed to a fully-connected layer to get the global conditional embedding vector with a fixed dimension of 128.

2.2. Detection network

We have built two types of detection network: strong- and weakly-supervised network. The details are given as follows.

Strong-supervised network The strong-supervised detection network is based on the state-of-the-art work for weakly supervised sound event detection by Dinkel *et al.* [4]. As shown in Figure 1, the network has two inputs: the conditional embedding vector and the mixture audio, which is trained to minimize the difference between the frame-level prediction results and the ground-truth labels. To be more specific, the network is composed of 4 convolutional layers, 1 Bi-GRU layer, and 2 fully-connected layers, each with a LeakyReLU activation except the last layer, which has a sigmoid activation. Given the input feature of the mixture audio $\boldsymbol{x} \in \mathcal{R}^{t \times f}$, where t and f denote the number of frames and the dimension of each frame respectively, the network aims to predict frame-level probabilities $\hat{p}_i = \mathbb{P}(Y = k|X = x_i, \boldsymbol{e}; \phi)$ where ϕ denotes the trainable parameters of detection network, $\boldsymbol{e}$ denotes the embedding from the conditional network and x_i denotes the i-th frame of the mixture audio $\boldsymbol{x}$. Here, the value of k is 0 or 1. Given the ground-truth label $p_i \in \{0, 1\}$ for each frame, the strong-supervised network can be optimized by minimize the binary cross-entropy (BCE) loss function:

$$\mathcal{L}_{sed} = \sum_{i=1}^{t}(-p_i \log \hat{p}_i - (1 - p_i) \log(1 - \hat{p}_i)) \quad (1)$$

where t indicates the number of frames.

Weakly-supervised network The difference between strong-supervised network and weakly-supervised network is that the latter needs a pooling layer to get the clip-level prediction. We add a linear softmax (LinSoft) pooling layer [17] after the last layer of the strong-supervised detection network. It aims to predict a clip-level probability $\hat{P} = f_{LSP}(\hat{p}_1, \hat{p}_2, ..., \hat{p}_t)$ where $f_{LSP}(\cdot)$ denotes the linear softmax pooling function and T denotes the number of frames. Given the clip-level ground-truth label $P \in \{0, 1\}$, BCE loss is also applied as the loss function:

$$\mathcal{L}_{sed}^{'} = -P \log \hat{P} - (1 - P) \log(1 - \hat{P}) \quad (2)$$

2.3. Fusion Module

As Figure 2 shows, we employ two fusion strategies to combine the conditional embedding and the feature of mixture audio. One is repeatedly concatenating the conditional embedding to the feature of mixture audio in each time frame. The other is projecting the conditional embedding and the feature of mixture audio to the same dimension by a 1-D convolutional layer, and then using multiplication operation to fuse them.

2.4. Mixup-TSD

Following the advanced data augmentation methods [18, 19, 20] for audio classification, we propose a data augmentation method based on the widely-used mixup [18] for TSD task. The core idea is to create a new training sample by mixing a pair of two training samples. More specifically, we can generate a new training sample $(M_{new}, R_{new}, y_{new})$ from the data and label pair (M_1, R_1, y_1) and (M_2, R_2, y_2) by the following equation.

$$X_{new} = \lambda X_1 + (1 - \lambda X_2) \quad (3)$$
$$R_{new} = \lambda R_1 + (1 - \lambda R_2) \quad (4)$$
$$y_{new} = \lambda y_1 + (1 - \lambda y_2) \quad (5)$$

where M_1 and M_2 are mixture audios, R_1 and R_2 are reference audios, y_1 and y_2 are the corresponding labels. Such Mixup-TSD works by generating lots of new training samples, and particularly, the target sound turns to be multi-label event instead of single-label event. We argue that this method can perform well at the early training stage, but needs a fine-tuning stage to fit for the single-label test. In practice, we set a ratio α to control whether to use Mixup-TSD while training, which means Mixup-TSD is applied with a probability of α. We set a linear decay for α from 0.3 to 0 during the whole training iterations.

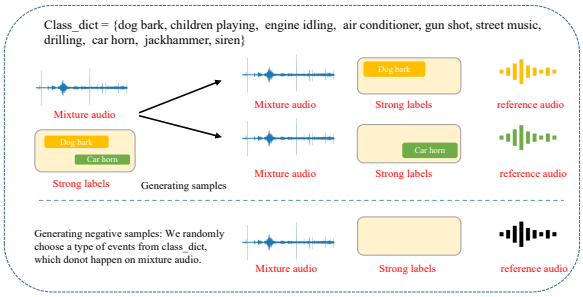


Figure 3: The process of data generation.

2.5. Jointly Training

To get a more sound-discriminative conditional embedding vector for TSDNet, we propose a jointly optimizing method for the conditional network via a multi-task manner. As show in Figure 1, the feature vector after the global pooling layer is fed to a fully-connected layer and a softmax nonlinearity to get the classification results. The conditional network can be optimized by minimizing the cross entropy loss between the predicted label and the ground truth label of the reference audio, along with the original detection task. The whole loss function is defined by formula (6).

$$\mathcal{L}_{total} = \mathcal{L}_{sed} + \mathcal{L}_{cls} \tag{6}$$

where $\mathcal{L}_{sed}$ denotes the loss of the detection task, and $\mathcal{L}_{cls}$ denotes the loss of the classification task, which is the cross-entropy loss between the ground-truth class labels and the predicted class labels.

3. URBAN-TSD DATASET

As there was no dataset for TSD task, we build a dataset called URBAN-TSD based on URBAN-SED [21] and UrbanSound8K datasets [22]. URBAN-SED is a sound event detection dataset with an urban setting, containing 10 event labels. This dataset's source material is the UrbanSound8K dataset containing 27.8 hours of data split into about 4-second clip segments. The URBAN-SED dataset encompasses 10,000 soundscapes generated using the Scaper soundscape synthesis library [21], which have been split into 6000 training, 2000 validation and 2000 test clips. The dataset contains mostly 10-second excerpts with strong labels, whereas each clip contains between one and nine events. We establish two types of dataset: strong-labelled dataset (URBAN-TSD-strong) and weakly-labelled dataset (URBAN-TSD-weak). The details of the number of samples are shown in Table 1.

3.1. URBAN-TSD-strong

We have built two strong datasets: URBAN-TSD-strong and URBAN-TSD-strong+. The details are given as follows.

URBAN-TSD-strong dataset: In this dataset, a sample includes three parts: a mixture audio, a reference audio and the strong label (the timestamp information of target sound). Mixture audios come from URBAN-SED dataset, and reference audio comes from UrbanSound8K. As shown in Figure 3, if there are N sound events in the mixture audio, we can generate N positive samples. For each positive sample, we randomly choose another sample that is in the same class from the UrbanSound8K as the reference audio.

URBAN-TSD-strong+ dataset: To further verify the ability of the model facing all negative frames, we generate samples that do not contain the target sound, which are called as negative samples. We add those samples because the mixture audio may not contain our target sound in the real world. The process of generating negative samples can summarize as: For any mixture audio from URBAN-SED, we randomly choose a reference audio whose sound events do not happen in the mixture audio. The timestamp information (label) of negative samples are all set as 0.

Table 1: The number of strong- and weakly-labelled data.

Type	Strong	Strong+	Weak
Training	23106	29106	41059
Validation	7681	9681	13661
Test	7702	9702	13682

Table 2: The performance comparison between TSDNet and other SED methods. Note that multiplication fusion is applied.

Method	Segment-based F1
CRNN [23]	64.7
CDur [4]	64.8
CTrans [24]	64.51
SEDT-AQ-FT-P3 [25]	65.77
Ours	**73.1**

3.2. URBAN-TSD-weak

We also build a weakly-labelled dataset which is easier to obtain but more challenging. Comparing with strong-labelled dataset, there is no onset or offset time provided. The label is set as 1 (which indicates target sound happens in the mixture audio) or 0 (which indicates target sound does not happen in the mixture audio) for the whole audio clip. Similarly, for any mixture audio from URBAN-SED, if N events happen, we can generate N positive samples. For weakly-labelled dataset, negative samples are very important for training, and we generate the same number of negative samples as positive samples.

4. EXPERIMENTS

4.1. Experimental setups

Training the conditional network: We pre-train the conditional network with the classification task on the UrbanSound8K dataset. All the raw audios are down-sampled to 44.1kHz and applied a Short Time Fourier Transform (STFT) with a window size of 400 samples, followed by a Mel-scaled filter bank on perceptually weighted spectrogram. After that, we get log-mel spectrogram and MFCC feature, and concatenate them as the input, which is the state-of-the-art solution[1]. The Adam optimizer [26] is used for a total of 50 epochs, with an initial learning rate of 1×10^{-3}.

Training the detection network: All the raw audios are down-sampled to 22.05kHz and applied a Short Time Fourier Transform (STFT) with a window size of 2048 samples, followed by a Mel-scaled filter bank on perceptually weighted spectrogram. This results in 64 Mel frequency bins and around 50 frames per second. The Adam optimizer [26] is used for 100 epochs, with an initial learning rate of 1×10^{-3}. Note that we only update the detection

[1]https://www.kaggle.com/adinishad/urbansound-classification-with-pytorch-and-fun/notebook

Table 3: The segment-based F-measure (%) of TSDNet with different fusion strategies. We carry out the experiments for three times and report the mean and standard deviation values.

Method	strong	strong+	weak
Concatenation	73.1±0.61	67.2±0.41	52.8±0.53
Multiplication	**73.1±0.19**	**69.3±0.23**	**53.0±0.32**

Table 4: The segment-based F-measure (%) of different settings for Mixup-TSD. Here, multiplication fusion is applied.

Mixup-TSD setting	strong	strong+	weak
fixed ratio 0	73.1	69.3	53.0
fixed ratio 0.2	74.9	71.1	55.2
fixed ratio 0.5	74.2	70.8	54.7
fixed ratio 0.8	74.0	70.4	53.6
fixed ratio 1.0	73.8	70.2	53.2
linear decay ratio	**75.1**	**71.6**	**55.8**

network with the detection loss.

Mixup-TSD: For all the experiments, we use the Mixup-TSD method on the spectrogram level as we find spectrogram-level method works better than waveform-level method. Under otherwise stated, the linear decay is set for α from 0.3 to 0.

Metrics: We use the segment-based F-measure [27] as the evaluation metric, which is the most commonly used metrics for sound event detection. All the F-scores are macro-averaged.

4.2. The performance comparison between TSDNet and other SED methods

In this part, we conduct experiments on URBAN-TSD-strong dataset to validate the effectiveness of our proposed method (TSDNet). We choose four previous state-of-the-art SED methods on URBAN-SED dataset. As for the baselines, we follow the model architecture in [24] to build the the Transformer-based model, which is referred to as CTrans in this paper. CDur [4] is the backbone of our TSDNet's detection network. Table 2 reports the results. We can see that our TSDNet significantly improve the performance due to introducing the reference audio.

4.3. Experimental results of different fusion strategies

We evaluate our baseline method with different fusion strategies. Table 3 reports the segment-based F-measure on the three datasets. The multiplication gets larger mean values and smaller standard deviation values in all situations, which is a more effective fusion method than the concatenation. In addition, TSDNet performs the worst on the URBAN-TSD-weak dataset, for the reason that no timestamp information is available during training.

4.4. Experimental results of Mixup-TSD

We further evaluate the Mixup-TSD method on three datasets, and report the performance in Table 4. If no augmentation is used, TSDNet can obtain the segment-based F-measures of 73.1%, 69.3% and 53.0% respectively. We can see that the Mixup-TSD method can significantly improve the performance over the baseline under all settings. More specifically, for a fixed ratio α, the performance tends to decrease with the ratio increasing, and the best performance can be achieved when the ratio is 0.2. We argue that the Mixup-TSD method can perform well at the early training stage, but it will change the detection target from single-label event to multi-label event so we need a fine-tuning stage to fit for the single-label test. With a linear decay ratio from 0.3 to 0, we can get the segment-based F-measures of 75.1%, 71.6% and 55.8% respectively.

Table 5: The performance of jointly training. Note that multiplication fusion and Mixup-TSD are applied.

Jointly training	Strong	Strong+	Weak
✗	73.1	69.3	53.0
✓	**76.3**	**72.2**	**56.8**

Table 6: Experimental results of open domain. Note that multiplication fusion is applied.

Jackhammer	Siren	Street_music	Average
40.9	42.2	48.0	43.7

4.5. Experimental results of jointly training

In addition, we explore whether jointly training can further improve the performance of target sound detection. Firstly, we initialize the TSDNet with a pre-trained model which is only trained by the detection task. Secondly, we fine-tune TSDNet by a multi-task manner introduced in Section 2.5. We set a learning rate of 1×10^{-4} for the whole model, and the Adam optimizer [26] is used for a total of 30 epochs. Experimental results are shown in Table 5. We can see that TSDNet with jointly training obtains 76.3%, 72.2% and 56.8% respectively, which brings 4.3%, 4.2% and 7.1% improvement over the baseline respectively. We find that the weak-supervised training still performs much worse than the strong-supervised training. We will study more effective weak-supervised methods in the future.

4.6. Open domain target sound detection

Table 6 shows the results of TSDNet evaluated on the open domain. We choose three events (jackhammer, siren, street_music) as new classes, which does not occur in the training set. Specifically, we first exclude all of the data include the three events in the URBAN-TSD-strong dataset, named URBAN-TSD-strong-7. Then we train TSDNet on URBAN-TSD-strong-7. We can see that TSDNet can be applied to the new class, *e.g.*, the F-score of street_music achieved 48.0% and the average F-score of the three new classes achieved 43.7%. We believe that adding more training data can further improve the performance of new class.

5. CONCLUSIONS

In this paper, we proposed a target sound detection network (TSDNet) which can be trained with both strong-supervised learning and weakly-supervised learning manners. In addition, our proposed Mixup-TSD data augmentation method and jointly training strategy further improve the performance of TSDNet. TSDNet is more applicable to real scenarios because it does not require prior knowledge about the number of sound events and easy to extend to the new events. In the future, we will explore more robust system. The source code and datasets of this work are released.[2]

[2]https://github.com/yangdongchao/Target-sound-event-detection

6. REFERENCES

[1] J. P. Bello, C. Silva, O. Nov, R. DuBois, A. Arora, J. Salamon, C. Mydlarz, and H. Doraiswamy, "Sonyc: A system for the monitoring, analysis and mitigation of urban noise pollution," *arXiv preprint arXiv:1805.00889*, 2018.

[2] D. Stowell and D. Clayton, "Acoustic event detection for multiple overlapping similar sources," in *2015 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*. IEEE, 2015, pp. 1–5.

[3] S. Hershey, S. Chaudhuri, D. Ellis, J. Gemmeke, A. Jansen, R. C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold, *et al.*, "CNN architectures for large-scale audio classification," in *IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2017, pp. 131–135.

[4] H. Dinkel, M. Wu, and K. Yu, "Towards duration robust weakly supervised sound event detection," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 887–900, 2021.

[5] L. Lin, X. Wang, H. Liu, and Y. Qian, "Specialized decision surface and disentangled feature for weakly-supervised polyphonic sound event detection," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 1466–1478, 2020.

[6] Q. Kong, Y. Xu, W. Wang, and M. D. Plumbley, "Sound event detection of weakly labelled data with cnn-transformer and automatic threshold optimization," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 2450–2460, 2020.

[7] Q. Kong, C. Yu, Y. Xu, T. Iqbal, W. Wang, and M. D. Plumbley, "Weakly labelled audioset tagging with attention neural networks," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 27, no. 11, pp. 1791–1802, 2019.

[8] Q. Wang, H. Muckenhirn, K. Wilson, P. Sridhar, Z. Wu, J. R. Hershey, R. A. Saurous, R. J. Weiss, Y. Jia, and I. L. Moreno, "Voicefilter: Targeted voice separation by speaker-conditioned spectrogram masking," *Proc. Interspeech*, pp. 2728–2732, 2019.

[9] M. Ge, C. Xu, L. Wang, E. S. Chng, J. Dang, and H. Li, "Multi-stage speaker extraction with utterance and frame-level reference signals," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 6109–6113.

[10] T. Ochiai, M. Delcroix, Y. Koizumi, H. Ito, K. Kinoshita, and S. Araki, "Listen to what you want: Neural network-based universal sound selector," *Proc. Interspeech*, pp. 1441–1445, 2020.

[11] Q. Kong, Y. Cao, T. Iqbal, Y. Wang, W. Wang, and M. Plumbley, "Panns: Large-scale pretrained audio neural networks for audio pattern recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 2880–2894, 2020.

[12] H. Wang, Y. Zou, D. Chong, and W. Wang, "Environmental sound classification with parallel temporal-spectral attention," in *Proc. Interspeech*, 2020, pp. 821–825.

[13] H. Wang, D. Chong, D. Huang, and Y. Zou, "What affects the performance of convolutional neural networks for audio event classification," in *International Conference on Affective Computing and Intelligent Interaction Workshops and Demos (ACIIW)*. IEEE, 2019, pp. 140–146.

[14] H. Wang, Y. Zou, and D. Chong, "Acoustic scene classification with spectrogram processing strategies," in *Proceedings of 5th the Workshop on Detection and Classification of Acoustic Scenes and Events (DCASE)*, 2020, pp. 210–214.

[15] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*. PMLR, 2015, pp. 448–456.

[16] V. Nair and G. Hinton, "Rectified linear units improve restricted boltzmann machines," in *ICML*, 2010.

[17] Y. Wang, J. Li, and F. Metze, "A comparison of five multiple instance learning pooling functions for sound event detection with weak labeling," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 31–35.

[18] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," *arXiv preprint arXiv:1710.09412*, 2017.

[19] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "SpecAugment: A simple data augmentation method for automatic speech recognition," *Proc. Interspeech*, pp. 2613–2617, 2019.

[20] Y. Tokozume, Y. Ushiku, and T. Harada, "Learning from between-class examples for deep sound recognition," in *International Conference on Learning Representations*, 2018.

[21] J. Salamon, D. MacConnell, M. Cartwright, P. Li, and J. P. Bello, "Scaper: A library for soundscape synthesis and augmentation," in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*. IEEE, 2017, pp. 344–348.

[22] J. Salamon, C. Jacoby, and J. P. Bello, "A dataset and taxonomy for urban sound research," in *Proceedings of the 22nd ACM International Conference on Multimedia*, 2014, pp. 1041–1044.

[23] I. Martín-Morató, A. Mesaros, T. Heittola, T. Virtanen, M. Cobos, and F. Ferri, "Sound event envelope estimation in polyphonic mixtures," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 935–939.

[24] K. Miyazaki, T. Komatsu, T. Hayashi, S. Watanabe, T. Toda, and K. Takeda, "Weakly-supervised sound event detection with self-attention," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 66–70.

[25] Z. Ye, X. Wang, H. Liu, Y. Qian, R. Tao, L. Yan, and K. Ouchi, "Sound event detection transformer: An event-based end-to-end model for sound event detection," *arXiv preprint arXiv:2110.02011*, 2021.

[26] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *ICLR*, 2015.

[27] A. Mesaros, T. Heittola, and T. Virtanen, "Metrics for polyphonic sound event detection," *Applied Sciences*, vol. 6, no. 6, p. 162, 2016.

MATCHING TEXT AND AUDIO EMBEDDINGS: EXPLORING TRANSFER-LEARNING STRATEGIES FOR LANGUAGE-BASED AUDIO RETRIEVAL

Benno Weck[1,2]*, Miguel Pérez Fernández*[1,2]*, Holger Kirchhoff*[1]*, Xavier Serra*[2]

[1] Huawei Technologies, Munich Research Center, Germany
{firstname.lastname}@huawei.com
[2] Universitat Pompeu Fabra, Music Technology Group, Spain
{firstname.lastname}01@estudiant.upf.edu, xavier.serra@upf.edu

ABSTRACT

We present an analysis of large-scale pretrained deep learning models used for cross-modal (text-to-audio) retrieval. We use embeddings extracted by these models in a metric learning framework to connect matching pairs of audio and text. Shallow neural networks map the embeddings to a common dimensionality. Our system, which is an extension of our submission to the Language-based Audio Retrieval Task of the DCASE Challenge 2022, employs the RoBERTa foundation model as the text embedding extractor. A pretrained PANNs model extracts the audio embeddings. To improve the generalisation of our model, we investigate how pretraining with audio and associated noisy text collected from the online platform Freesound improves the performance of our method. Furthermore, our ablation study reveals that the proper choice of the loss function and fine-tuning the pretrained models are essential in training a competitive retrieval system.

1. INTRODUCTION

The *DCASE2022* challenge subtask 6b provides a platform to stimulate research in the underexplored problem domain of language-based audio retrieval [1]. The goal of this task is to find the closest matching audio recordings for a given text query. A possible application for this task is a search engine for audio files in which a user can enter a free-form textual description to retrieve matching recordings. Such systems need to draw a connection between the two modalities: audio and text.

Given the complex nature of both audio and text, we expect that a system can only perform well in this task if it can capitalise on a large amount of training data. Due to the novelty of the task, not many previous studies and systems exist for language-based audio retrieval and training data is still limited. We instead turn to the fields of machine listening, specifically audio tagging, and natural language processing to draw inspiration from related problems and make use of existing resources such as pretrained models. It has become a popular approach to use large-scale pretrained models in a transfer learning setup for tasks where only limited training data is available.

The goal of this work is to study a simple, generic cross-modal alignment system. Our approach should be able to process audio and text independently to be used in a cross-modal retrieval context. Therefore, we leverage the power of pretrained models and a metric learning framework to semantically link the two modalities. We limit the complexity of our approach by employing the pretrained models with fixed weights and only train shallow network architectures to perform the alignment. Additionally, this paper presents an analysis of our submission [2] to the Language-based Audio Retrieval Task of the DCASE2022 Challenge. With an ablation study, we investigate the impact of different training strategies on the performance of our system. This helps us to understand the differences in performance between our system and other submissions to the challenge.

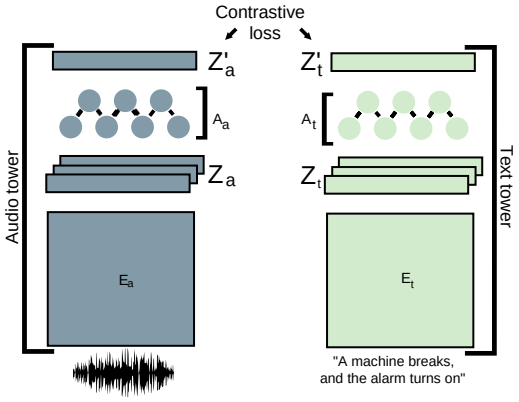


Figure 1: Overview of the architecture of our system. An audio tower and a text tower process the respective input data separately and produce a single embedding.

The remainder of this paper is structured as follows. In the next section, we introduce the methodological framework of our system. Section 3 explains the experiments that lead to our challenge submission and Section 4 presents the results of the submitted systems. The results of additional experiments performed as an ablation study are discussed in Section 5. We summarise our findings in Section 6.

2. METHOD

We adopt a metric learning [3] framework in our approach, which differs from a classification scenario used in related tasks such as audio tagging. In a classification scenario, the outputs of a network are the predictions for the different classes and the features that characterise each of those classes remain in the intermediate layers of the network. However, in metric learning, the goal is to obtain those features directly, so that the output of the network can be used to measure the similarity between two different inputs. The

features learned by the system can be referred to as an 'embedding space'. For each input, a network trained with metric learning will return an embedding $\mathbf{Z} \in \mathbb{R}^F$, where $\mathbb{F}$ is the size of the embedding, which is a hyper-parameter.

Metric learning usually relies on 'positive' and 'negative' examples to teach the networks. Positive examples are pairs of inputs that share some similarities, e.g., two sounds of birds singing. Negative examples, on the other hand, contain dissimilar content, e.g., a recording of a bird singing and a car's ignition system. The positive examples should be 'closer' in the embedded space, while the negative ones should lie in different regions. In our case, positive examples are audios and their corresponding descriptions.

Our system consists of two components – an audio tower and a text tower – to separately process the audio and text input. Each tower is further divided into an encoder, $\mathrm{E}(\cdot)$, and an embeddings' adapter, $\mathrm{A}(\cdot)$. As the audio encoder E_a and the text encoder E_t, we employ pretrained models. An overview of our method is presented in Figure 1.

More specifically, an audio input $\mathbf{X}_a$ or a text input $\mathbf{X}_t$ are processed by E_a and E_t, respectively, as

$$\begin{aligned} \mathbf{Z}_a &= \mathrm{E}_a(\mathbf{X}_a), \\ \mathbf{Z}_t &= \mathrm{E}_t(\mathbf{X}_t), \end{aligned} \tag{1}$$

where $\mathbf{Z}_i \in \mathbb{R}^{T_i \times F_i}, i \in \{a, t\}$ is a sequence of T_i intermediate representations with F_i features provided by the pretrained model (i.e., an embedding sequence). Then, the adapters A_a and A_t will process $\mathbf{Z}_a$ and $\mathbf{Z}_t$ as

$$\begin{aligned} \mathbf{Z}'_a &= \mathrm{A}_a(\mathbf{Z}_a), \\ \mathbf{Z}'_t &= \mathrm{A}_t(\mathbf{Z}_t), \end{aligned} \tag{2}$$

where $\mathbf{Z}'_a, \mathbf{Z}'_t \in \mathbb{R}^{F'}$ are single embeddings and F' denotes their dimensionality. The intermediate embedding sequences $\mathbf{Z}_a$ and $\mathbf{Z}_t$ produced by the audio and text encoder respectively will differ in dimensionality. The main purpose of the adapters is to match the dimensionality of text and audio embeddings in order to enable comparisons. We use the metric learning techniques described above to align the embedded spaces $\mathbf{Z}_a$ and $\mathbf{Z}_t$, so during training the adapters will learn to bring both into a common embedding space.

We experimented with two different losses. The first is the contrastive loss [4], which we used for our submission to the DCASE2022 challenge. Given the cosine similarity s between a pair of embeddings with labels l_1 and l_2, the contrastive loss is defined by:

$$L_{contrastive} = \begin{cases} 1 - s & \text{if } l_1 = l_2 \\ \max(0, s) & \text{otherwise.} \end{cases} \tag{3}$$

The second loss that we use in our experiments is the Normalized Temperature-scaled Cross Entropy (NT-Xent) loss [5], which is used by the leading submissions in the DCASE2022 challenge. For a more concise explanation of this loss, we refer the reader to the technical reports of the top-ranked teams [6, 7].

For the final application as a text-to-audio retrieval system, we compute the embedding of the text query $\mathbf{Z'}_t$ and compares it to all pre-computed embeddings $\mathbf{Z'}_a$ of the audio items in the dataset by means of the cosine similarity. Ranking the audio items by their similarity score in descending order provides the retrieval results.

Description	Tags
"Typing on a mechanical keyboard"	"click", "keyboard", "mechanical", "computer", "typing", "button"
"Pouring liquid in a shot glass, picking it up, drinking & slamming it down (not too hard) on the table."	"slam", "glass", "pour", "drink", "liquid", "alcohol", "shot"
"opening of shower curtain, turning shower on, water running, turning shower off, getting out"	"shower", "water", "bathroom", "bathtub", "human"

Table 1: Hand-picked examples of descriptions and text labels from the metadata of the FSD50k dataset.

3. EXPERIMENTS

3.1. Datasets

As the main dataset in our work, we employ the development dataset provided for this challenge, *Clotho v2* [8], and use its official splits for training, validation, and final evaluation (testing). We posit that the Clotho dataset is relatively small for the training of deep-learning-based retrieval systems and any system might benefit from additional training data. Datasets combining audio and text are scarce, however, and the few that exist besides Clotho are either specific to a certain domain (e.g., urban soundscapes only [9]) or their audio content is not freely accessible [10]. This is why we decided to use weakly aligned text and audio pairs collected from the online platform Freesound [11], which also served as the data source for Clotho. Freesound allows users to upload an audio recording along with a textual description and a set of tags. This type of metadata was used before to extend the training data of Clotho but in the context of an automated audio captioning task [12]. For simplicity and reproducibility, we limit ourselves to the *dev* subset of the *FSD50k* dataset [13]. We assume that the audios in this dataset closely resemble the challenge audio data as the dataset mainly comprises recordings of sound events. Moreover, similarly to Clotho, audio clips are not longer than 30 seconds. The descriptions and tags in the dataset contain rich information about the content of the audio clip as can be seen from the examples given in Table 1. Nevertheless, the text data is noisy and also contains some undesired text.[1] To clean the descriptions we remove all HTML mark-up and limit each text to 500 characters in a pre-processing step. To form a 'sentence' out of the tags, we join them with a single white space in the order given in the dataset. The *dev* split of the FSDK50 dataset contains almost 44100 audio files and we use half of them. By using descriptions and tag sequences, we can extend the training data by 40966 text-audio pairs (more than twice the amount of caption-audio pairs in the training subset of Clotho). We refer to Clotho's data as 'clean' and FSD50k's data as 'noisy'.

3.2. Evaluation & Metrics

We evaluate the ranked retrieval results generated by our systems with the same four metrics as the challenge organisers. Specifically, we report three 'recall at k' metrics (*Recall@1*, *Recall@5*, *Recall@10*) and one 'mean average precision at k' (*mAP@10*), where a score for a given query is computed for the top-k retrieved results and all scores are averaged over the entire set of queries. We direct the reader to [14] for an in-depth explanation of the metrics.

[1] For example: "CAUTION: THIS PACK IS A CHEAP HOME RECORD. (But this one sounds a bit better)"

	Development test set				Challenge test set
	Recall@1	Recall@5	Recall@10	mAP@10	mAP@10
Challenge baseline*	0.03	0.11	0.19	0.07	0.061
ensmbl_5* [6]	0.188	0.447	0.587	0.299	0.276
Mei_Surrey_1* [7]	0.150	0.400	0.530	0.260	0.251
ATAE	0.071 (0.064 - 0.078)	0.217 (0.206 - 0.228)	0.325 (0.312 - 0.337)	0.136 (0.128 - 0.143)	0.114
ATAE-ET	0.064 (0.057 - 0.070)	0.194 (0.184 - 0.205)	0.288 (0.275 - 0.300)	0.121 (0.114 - 0.128)	0.113
ATAE-EP-F	0.067 (0.061 - 0.074)	0.200 (0.189 - 0.210)	0.299 (0.286 - 0.311)	0.127 (0.120 - 0.134)	0.121
ATAE-NP-F	0.072 (0.065 - 0.079)	0.225 (0.214 - 0.236)	0.325 (0.313 - 0.338)	0.139 (0.131 - 0.146)	0.128

Table 2: Retrieval metrics for the four submitted systems, the two leading teams, and the challenge baseline. The 95% confidence intervals computed by jackknife resampling are given in parentheses. Results marked with * were reported by the challenge organisers.

3.3. Implementation details

Our system is implemented by relying on the *PyTorch* [15] framework in connection with the *pytorch-metric-learning* package [16]. For the text processing, we employ the *Transformers* library [17] and use the pretrained *distilroberta-base* model as the text encoder. This model is a compressed version of the original *RoBERTa* model [18] created by a knowledge distillation procedure [19]. It is smaller and faster than the original variant while retaining high performance on downstream tasks. Similar to our previous work on audio captioning [20], we decided to use the penultimate layer as the intermediate embeddings $\mathbf{Z}_t$. The extracted text embeddings have a dimensionality F_t of 768.

For the audio processing, we use a pretrained *PANNs* model [21] as the audio encoder. We follow the authors' suggestion and compute embeddings by taking the post-activation output of the penultimate layer of their *CNN14* model.[2] All audio clips are resampled to a sampling rate of 32 kHz in a preprocessing step. The extracted intermediate audio embeddings $\mathbf{Z}_a$ have a dimensionality F_a of 2048.

We use simple feed-forward neural networks to adapt each embedding sequence to the common dimensionality. Both adapters consist of a two-layer perceptron with a layer size of 512 and a rectified linear unit (ReLU) as activation function after the first layer. We use the average of all embeddings in a sequence as the final representation.

The system is optimised by minimising the contrastive loss with the Adam algorithm [22] ($\alpha = 0.001, \beta_1 = 0.9, \beta_2 = 0.999$, and $\epsilon = 10^{-8}$). We do not fine-tune the encoder models in our approach and only optimise the adapters. To form a minibatch we randomly select 32 audio-text pairs from the training set. We compute the loss for every possible combination of similar and dissimilar samples (including text-to-text and audio-to-audio pairs) and take the mean across all non-zero loss values. Every epoch the mAP@10 metric is computed on the validation dataset. We start training with a learning rate of 0.0001 and reduce it by a factor of 10 if no improvement was found for five epochs. Finally, the training is stopped after ten epochs with no improvement and the model weights are reverted to the checkpoint of the epoch with the highest score.

3.4. Submitted systems

We submit four different configurations of our system. All share the same model hyperparameter configurations but differ in the way the available training data was used to train them. Specifically, we experiment with: 1. adding no external dataset in our training, 2. extending the training data with noisy data from the FSD50k dataset, 3. pretraining with noisy and clean data and later fine-tuning with clean data only, and 4. pretraining exclusively with noisy data and fine-tuning with clean data only.

In every training (also if we refer to it as pretraining or fine-tuning), we follow the optimisation procedure described above.

ATAE: Aligned Text and Audio Embeddings In its standard configuration, our system is trained solely with the challenge development dataset Clotho. We refer to it as 'Aligned Text and Audio Embeddings' or *ATAE* for short.

ATAE-ET: Aligned Text and Audio Embeddings – Extended dataset for Training Next, we want to investigate if adding extra training data helps to improve retrieval performance. To achieve this we combine the noisy FSD50k and the clean Clotho data into a single training dataset.

ATAE-EP-F: Aligned Text and Audio Embeddings – Extended dataset for Pretraining – Fine-tuning To balance out the potential negative effects of the noise in the training data, we fine-tune the trained ATAE-ET model by again training with the clean Clotho dataset.

ATAE-NP-F: Aligned Text and Audio Embeddings – Noisy dataset for Pretraining – Fine-tuning Finally, to be able to better judge the effect of the noisy data for pretraining, we use the datasets in two separate training stages. We first train a model on the noisy data and then fine-tune it on the clean dataset.

4. RESULTS

Table 2 compares the metrics achieved for our four systems with the challenge baseline and two of the leading submissions on the challenge development test set and the challenge test set. We follow the lead of the challenge organisers and report a jackknife approximated 95% confidence interval for each metric [23]. Based on the results on the development test set, we make the following observations. First, our approach produces good quality results even in the standard training setup (ATAE: mAP@10 = 0.136 for the development test set). Second, extending the challenge dataset with additional (noisy) training data significantly degrades retrieval performance (ATAE-ET: mAP@10 = 0.121). Third, even fine-tuning the

[2]Pretrained weights can be found at: `https://doi.org/10.5281/zenodo.3987831`

second system on the clean challenge dataset seems to give worse results (ATAE-EP-F: mAP@10 = 0.127) in comparison with simply training only with the challenge dataset (ATAE). Fourth, our system first pretrained with noisy data only and then fine-tuned on the challenge dataset (ATAE-NP-F: mAP@10 = 0.139) improves on the performance of the first experiment but only slightly. Finally, all of our submitted systems surpass the challenge baseline in each metric by a comfortable margin but are inferior to the best systems in the challenge.

Since the metrics of our best system (ATAE-NP-F) lie within the confidence intervals of our next best system (ATAE) and vice versa, we conclude that no significant difference is measurable between them. These results suggest that no apparent advantage exists for our method in utilising additional noisy training data. However, when comparing the two systems (ATAE & ATAE-NP-F) on the challenge test set the advantage of pretraining with external data is more noticeable. A possible explanation for this might be that the model pretrained with additional external data has better generalisation capabilities and is less affected by a shift in data distribution.

5. ABLATION STUDY

Our approach is similar to the systems of the two top-ranked teams ([6, 7]) in the DCASE2022 challenge, yet we fail to reach the same level of retrieval performance. For example, analogous to us, both teams employ a two-tower architecture and shallow neural networks as adapter layers. Their choice of pretrained models (e.g., PANNs & RoBERTa) is also similar to ours. The most striking differences between our and their submissions are that they decided to: (i) use NT-Xent as a loss function, (ii) fine-tune the encoder models, and (iii) use the AudioCaps dataset [10] in pretraining. In view of this resemblance, we conduct additional experiments to investigate why a large gap in performance exists between our submission and the top-ranked systems.

We test five additive changes in training configuration. The results for each of the configurations are computed from five training runs. First, we employ the NT-Xent loss instead of the contrastive loss. Second, we assess the impact of pair selection for the loss function on the retrieval metrics. Our submission systems were trained considering not only text-audio pairs but also text-text and audio-audio pairs in the loss calculation. Since samples from different training instances (i.e., with different labels) will be considered dissimilar but could contain semantically similar content (e.g., two different recordings of birds), this could harm the training process. Therefore, we compare using only text-audio pairs in the loss calculation with using all possible pairs. Third, we want to test if our approach is restricted by the fixed encoder models and can benefit if they are fine-tuned in the training process. To limit the computational cost, we adopt the idea to only fine-tune the text encoder from a work in computer vision that showed that only fine-tuning the text model can help to train competitive text-to-image alignment models [24]. Fourth, we investigate the potential of pretraining with additional data. As we saw from the results in Section 4, pretraining with extra (noisy) data might help the model generalise better to unseen data. Also, both leading teams adopt pretraining in their training process. This is why we test if adding a pretraining stage relying on the entire *dev* split of the FSDK50 dataset can enhance our system's performance. Finally, we evaluate the benefits of fine-tuning both encoder models instead of only the text encoder similar to the approach in [7].

Figure 2 compares all ablation experiment configurations by

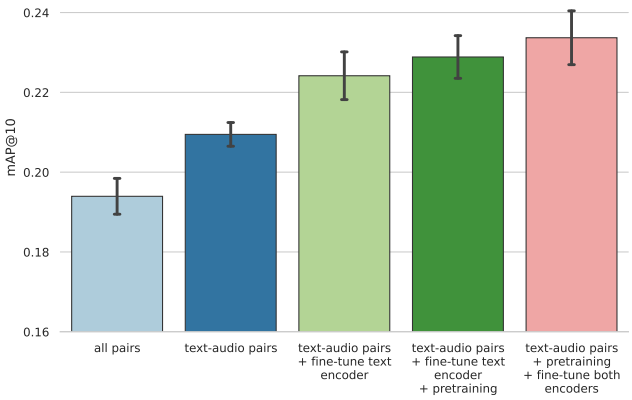


Figure 2: Comparison of the average retrieval results measured in mAP@10 on the development test set for different training configuration settings. The error bars show the standard deviation.

the average mAP@10 achieved on the development test set. What can be clearly seen in this figure is the accumulative increase in mAP@10 with every added change. We find that replacing the contrastive loss with the NT-Xent loss (see 'all pairs' in Fig. 2) already gives improved results in comparison with our challenge submission (mAP@10 = 0.193 compared to ATAE: mAP@10 = 0.136). Only considering text-audio pairs in the NT-Xent loss, however, further improves the retrieval performance to mAP@10 = 0.209. Furthermore, fine-tuning the text encoder model and including a pretraining stage adds to the improvement (mAP@10 = 0.224 and mAP@10 = 0.228, respectively). As the last change, fine-tuning both encoder models results in the best score on average (mAP@10 = 0.233). This comparison points to the conclusion that fine-tuning the encoder models and a pretraining stage are essential to achieve a high retrieval performance with our method. However, with the small sample size, the results must be interpreted with caution as the difference between the last three settings might not be significant.

6. CONCLUSION

We presented an analysis of our submission for the *Language-based Audio Retrieval* subtask of the DCASE2022 challenge. Our approach consists of extracting embeddings for the text and the audio through pretrained encoder models and mapping these embeddings to a shared space with a cross-modal alignment procedure. The best system in our submission is a model that is first pretrained with noisy text-audio data collected from Freesound and later fine-tuned on the challenge dataset. Even though our approach is similar to those of other teams we fall behind in the competition. Through an ablation study, we show that a large part of the performance gap can be attributed to our choice of the loss function and the fact that we keep encoders fixed instead of fine-tuning them. Moreover, we note promising results when pretraining our models with noisy data. Future work should further investigate the use of large quantities of noisy data for pretraining.

7. REFERENCES

[1] H. Xie, S. Lipping, and T. Virtanen, "DCASE 2022 Challenge Task 6B: Language-Based Audio Retrieval," 2022. [Online]. Available: https://arxiv.org/abs/2206.06108

[2] B. Weck, M. Pérez Fernández, H. Kirchhoff, and X. Serra, "Aligning Audio and Text Embeddings for the Language-Based Audio Retrieval Task of the DCASE Challenge 2022," DCASE2022 Challenge, Tech. Rep., July 2022.

[3] K. Mahmut and B. Hasan, "Deep metric learning: A survey," *Symmetry*, vol. 11, no. 9, p. 1066, Aug 2019. [Online]. Available: http://dx.doi.org/10.3390/sym11091066

[4] S. Chopra, R. Hadsell, and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1, 2005, pp. 539–546 vol. 1.

[5] K. Sohn, "Improved deep metric learning with multi-class n-pair loss objective," in *Advances in Neural Information Processing Systems*, D. Lee, M. Sugiyama, U. Luxburg, *et al.*, Eds., vol. 29. Curran Associates, Inc., 2016.

[6] X. Xu, Z. Xie, M. Wu, and K. Yu, "The SJTU System for DCASE2022 Challenge Task 6: Audio Captioning with Audio-Text Retrieval Pre-training," DCASE2022 Challenge, Tech. Rep., July 2022.

[7] X. Mei, X. Liu, H. Liu, *et al.*, "Language-Based Audio Retrieval with Pre-trained Models," DCASE2022 Challenge, Tech. Rep., July 2022.

[8] K. Drossos, S. Lipping, and T. Virtanen, "Clotho: an Audio Captioning Dataset," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Barcelona, Spain: IEEE, May 2020, pp. 736–740. [Online]. Available: https://ieeexplore.ieee.org/document/9052990/

[9] I. Martín-Morató and A. Mesaros, "Diversity and Bias in Audio Captioning Datasets," in *Proceedings of the 6th Workshop on Detection and Classification of Acoustic Scenes and Events 2021 (DCASE 2021), Online, November 15-19, 2021*, F. Font, A. Mesaros, D. P. W. Ellis, *et al.*, Eds., 2021, pp. 90–94. [Online]. Available: https://dcase.community/documents/workshop2021/proceedings/DCASE2021Workshop_Martin_34.pdf

[10] C. D. Kim, B. Kim, H. Lee, and G. Kim, "AudioCaps: Generating Captions for Audios in The Wild," in *NAACL-HLT*, 2019.

[11] F. Font, G. Roma, and X. Serra, "Freesound Technical Demo," in *Proceedings of the 21st ACM International Conference on Multimedia*, ser. MM '13. New York, NY, USA: Association for Computing Machinery, 2013, pp. 411–412, event-place: Barcelona, Spain. [Online]. Available: https://doi.org/10.1145/2502081.2502245

[12] Q. Han, W. Yuan, D. Liu, *et al.*, "Automated Audio Captioning with Weakly Supervised Pre-Training and Word Selection Methods," in *Proceedings of the 6th Workshop on Detection and Classification of Acoustic Scenes and Events 2021 (DCASE 2021), Online, November 15-19, 2021*, F. Font, A. Mesaros, D. P. W. Ellis, *et al.*, Eds., 2021, pp. 6–10. [Online]. Available: https://dcase.community/documents/workshop2021/proceedings/DCASE2021Workshop_Han_9.pdf

[13] E. Fonseca, X. Favory, J. Pons, *et al.*, "FSD50K: an open dataset of human-labeled sound events," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 30, pp. 829–852, 2022, publisher: IEEE.

[14] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to information retrieval*. New York: Cambridge University Press, 2008.

[15] A. Paszke, S. Gross, F. Massa, *et al.*, "PyTorch: An Imperative Style, High-Performance Deep Learning Library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, *et al.*, Eds. Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[16] K. Musgrave, S. Belongie, and S.-N. Lim, "PyTorch Metric Learning," 2020. [Online]. Available: https://arxiv.org/abs/2008.09164

[17] T. Wolf, L. Debut, V. Sanh, *et al.*, "Transformers: State-of-the-Art Natural Language Processing," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Online: Association for Computational Linguistics, Oct. 2020, pp. 38–45. [Online]. Available: https://aclanthology.org/2020.emnlp-demos.6

[18] Y. Liu, M. Ott, N. Goyal, *et al.*, "RoBERTa: A Robustly Optimized BERT Pretraining Approach," 2019. [Online]. Available: https://arxiv.org/abs/1907.11692

[19] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter," 2019. [Online]. Available: https://arxiv.org/abs/1910.01108

[20] B. Weck, X. Favory, K. Drossos, and X. Serra, "Evaluating Off-the-Shelf Machine Listening and Natural Language Models for Automated Audio Captioning," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2021 Workshop (DCASE2021)*, Barcelona, Spain, Nov. 2021, pp. 60–64.

[21] Q. Kong, Y. Cao, T. Iqbal, *et al.*, "PANNs: Large-Scale Pretrained Audio Neural Networks for Audio Pattern Recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 2880–2894, 2020.

[22] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2015. [Online]. Available: http://arxiv.org/abs/1412.6980

[23] A. Mesaros, A. Diment, B. Elizalde, *et al.*, "Sound Event Detection in the DCASE 2017 Challenge," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, pp. 992–1006, 2019.

[24] X. Zhai, X. Wang, B. Mustafa, *et al.*, "LiT: Zero-Shot Transfer with Locked-image Text Tuning," in *IEEE / CVF Computer Vision and Pattern Recognition Conference (CVPR)*, 2022.

CONTINUAL LEARNING FOR ON-DEVICE ENVIRONMENTAL SOUND CLASSIFICATION

Yang Xiao[1,*], *Xubo Liu*[2,*], *James King*[2], *Arshdeep Singh*[2], *Eng Siong Chng*[1], *Mark D. Plumbley*[2], *Wenwu Wang*[2]

[1] School of Computer Science and Engineering, Nanyang Technological University, Singapore
[2] Centre for Vision, Speech and Signal Processing (CVSSP), University of Surrey, UK

ABSTRACT

Continuously learning new classes without catastrophic forgetting is a challenging problem for on-device environmental sound classification given the restrictions on computation resources (e.g., model size, running memory). To address this issue, we propose a simple and efficient continual learning method. Our method selects the historical data for the training by measuring the per-sample classification uncertainty. Specifically, we measure the uncertainty by observing how the classification probability of data fluctuates against the parallel perturbations added to the classifier embedding. In this way, the computation cost can be significantly reduced compared with adding perturbation to the raw data. Experimental results on the DCASE 2019 Task 1 and ESC-50 dataset show that our proposed method outperforms baseline continual learning methods on classification accuracy and computational efficiency, indicating our method can efficiently and incrementally learn new classes without the catastrophic forgetting problem for on-device environmental sound classification.

Index Terms— Continual learning, environmental sound classification, on-device, convolutional neural networks

1. INTRODUCTION

Environmental sound classification aims to categorize audio recordings into pre-defined environmental sound classes [1]. Recently, on-device environmental sound classification [2, 3, 4] has attracted increasing research interest, as shown in Task 1 of Detection and Classification of Acoustic Scenes and Events (DCASE) 2022 Challenge: "Low-Complexity Acoustic Scene Classification" [5]. Such a sound classification system with low computation-complexity can be deployed on mobile and embedded platform for many real-world audio applications, such as acoustic surveillance [6], bio-acoustic monitoring [7] and multimedia indexing [8].

Most existing environment sound classification models [1, 3, 4, 9, 10] are trained with limited sound classes, which cannot directly adapt to new sound classes. When model developers want to expand the categories of environmental sounds to be classified, one way to do this is to fine-tune the model with new classes of data [11, 12]. However, this method may discard previously learned knowledge during the fine-tuning process: this is also known as the catastrophic forgetting problem [13]. Another possible solution is to re-train sound classification models with a mixture of historical and new data. However, this method is resource- and time-consuming in real-world on-device scenarios. As the solution based on re-training is computationally expensive, it is important to design efficient and effective methods to adapt the trained on-device sound classification model to new sound classes.

Continual learning (CL) [14, 15, 16] aims to continuously learn new knowledge over time while retaining and reusing previously learned knowledge. Existing CL methods can be generally divided into two categories: regularization-based methods [17, 18] and replay-based methods [19, 20]. Regularization-based methods use a regularization loss to preserve previously learned model parameters when learning new knowledge. Replay-based methods use a memory update algorithm (MUA) [20, 21, 22] to sample a few informative examples from historical data. The selected examples are used to preserve information about old classes when training new classes. Recently, replay-based CL methods have shown promising results outperforming regularization-based methods in audio tasks such as keywords spotting [23, 24] and sound event detection [25]. However, CL in on-device applications, such as on-device environmental sound classification, has received less attention in the literature, which is the focus in this paper. The on-device scenarios are often associated with restrictions in storage and memory space [3], which can pose challenges to replay-based CL which relied on external memory to restore historical data. As a result, the sound classification models that can be operated on the device may be limited in their capacities, thus prone to forgetting old knowledge when continuously learning new sound classes.

In this work, we investigate the replay-based CL (RCL) methods for on-device environmental sound classification. We first study the performance of existing memory update algorithm (MUA) methods such as *Reservoir* [21], *Prototype* [20] and *Uncertainty* [22] (as described in Section 2.1) on RCL for on-device environmental sound classification. We empirically demonstrate that *Uncertainty* [22] method performs best in our scenario. Furthermore, we propose *Uncertainty*++, a simple yet efficient MUA method based on *Uncertainty* method. Different to the *Uncertainty* method, our proposed *Uncertainty*++ introduces the perturbations to the embedding layer of the classifier. As a result, the computation cost (e.g., running memory and time) can be significantly reduced when measuring the data uncertainty. We evaluate the performance of our method on the DCASE 2019 Task1 [26] and the ESC-50 [27] datasets with on-device model BC-ResNet-Mod (∼86k parameters) [28, 29]. Experimental results show that *uncertainty*++ outperforms the existing MUA methods on classification accuracy, indicating its potential in real-world on-device audio applications. Our proposed method is model-independent and simple to apply. Our code is made available at the GitHub[1].

The remainder of this paper is organized as follows. Section 2 introduces the continual learning method we proposed for on-device environmental sound classification. Section 3 and Section 4 present the experimental settings and the evaluation results. Conclusions and future directions are given in Section 5.

*The first two authors contributed equally to this work.

[1] https://github.com/swagshaw/ASC-CL

2. METHOD

This section first describes replay-based continual learning and four memory update algorithms, and then introduces the proposed *uncertainty++* algorithm.

2.1. Replay-based continual learning

Following the continual learning setting [14, 18, 25] of environmental sound classification, we assume that the model M should identify all classes in a series of tasks $T = \{\tau_0, \ldots, \tau_t\}$ without catastrophic forgetting. For each task $\tau \in T$, we have input pairs (x, y) and classes C, where x denotes audio waveforms and y are classes $c \in C$. We aim to minimize a cross-entropy loss of all classes C present in the current task τ formulated as:

$$L_{CE}(\tau) = \sum_{c \in C} y_c log \frac{exp(M(x)_c)}{\sum_{c \in C} exp(M(x)_c)}, \quad (1)$$

Where $M(x)$ denotes the output of the model M for input x.

The parameters learned from the previous task are potentially overwritten after learning the new class, also known as catastrophic forgetting. To mitigate this issue, we introduce replay-based methods. The replay-based methods utilize a region of the memory which is called 'replay buffer' to temporarily store the historical training samples to maintain the performance.

Re-training sound classification models with the mixture of the whole historical and new data is resource- and time-consuming in real-world on-device scenarios. To mitigate this issue, the replay-based methods access only a subset of the historical data to save the storage space. In this case, how to select the part of samples to the replay buffer by the memory update algorithm is the key.

Specifically, in the training of task τ_t, the replay buffer stores the selected training samples from the previous $t-1$ learned task(s) $\{\tau_0, \tau_1, \ldots, \tau_{t-1}\}$, and builds the training data buffer $\hat{D}_t$ for task τ_t formulated as:

$$\hat{D}_t = g(\hat{D}_{t-1}) \cup D_t, \quad (2)$$

where g is the memory update algorithm [24], $\hat{D}_{t-1}$ is the training data buffer for task τ_{t-1}, and D_t is the incoming data for the new task.

2.1.1. Memory update algorithm (MUA)

We introduce four memory update algorithms in the literature. Generally, we assume that the memory update should select L samples from the training data $\hat{D}_{t-1}$ of the previous task τ_{t-1} for the training of the task τ_t.

Random [30] memory update algorithm selects L new samples $\{(x_1, y_1), (x_2, y_2), \ldots, (x_L, y_L)\}$ for the next task randomly from the candidates $\hat{D}_{t-1}$ into replay buffer.

Reservoir [21] memory update algorithm conducts uniform sampling from $\hat{D}_{t-1}$. Specifically, the reservoir algorithm initializes the replay buffer indexed from 1 to L, containing the first L items $\{(x_1, y_1), (x_2, y_2), \ldots, (x_L, y_L)\}$ of the candidates. When updating replay buffer from the candidates, for each sample, the reservoir algorithm generates a random number m uniformly in $\{1, \ldots, len(\hat{D}_{t-1})\}$. If $m \in \{1, \ldots, L\}$, then the sample with the index m in the replay buffer is replaced with the sample $\hat{D}_{t-1}[m]$.

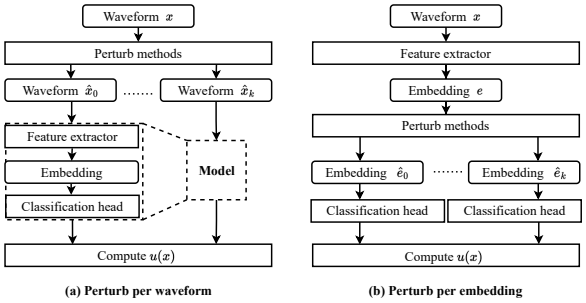


Figure 1: *Block diagram of the native uncertainty approach and our proposed approach. Specifically, the naive approach adds perturbations to x by waveform and generates multiple waveform as $\hat{x}$. Our approach inputs the embedding e and generates perturbed embedding $\hat{e}$ which means we only save the embedding. The output of the backbone of the model is calculated only once. "Compute $u(x)$" is to compute $u(x)$ by Eq. (3). The K refers to the number of the perturbations generated by perturb methods.*

Prototype [20] memory update algorithm selects the samples from $\hat{D}_{t-1}$ where the embedding of the classifier is close to the embedding mean of its own class. Specifically, the algorithm first groups the $\hat{D}_{t-1}$ into subsets as $D_c, c = 1 \ldots N^t$ by unique classes, where N^t denotes the total numbers of unique classes in the $\hat{D}_{t-1}$ set. Then the algorithm uses the current model to extract the embedding of the candidates for each D_c and calculates the class mean by the embedding as the average feature vector. For each class, the algorithm selects the samples of the candidates so that the average feature vector over the replay buffer provides best approximate to the average feature vector over all the samples of the corresponding class.

Uncertainty [22] memory update algorithm selects the sample by the uncertainty of the sample through the inference by the classification model. Specifically, the first step groups the $\hat{D}_{t-1}$ in the same way as the *prototype* algorithm introduced above. The second step estimates the uncertainty of each sample x in D_c. Predictive likelihood captures how well a model fits the data, with larger values indicating better model fit. Uncertainty score can be determined from predictive likelihood [31]. Following the derivation from [31], the predictive likelihood of a sample given by the model can be approximated by the Monte-Carlo (MC) integration [32] method with the model outputs of perturbed samples [24], which is defined as follows:

$$P(y = c \mid x) = \int p(y = c \mid \hat{x}) p(\hat{x} \mid x) d\hat{x}, \quad (3)$$

where x, $\hat{x}$, y denote an audio utterance of one class, the perturbed samples of x, and the label of x. Therefore, the uncertainty of the audio utterance x is formulated as $u(x)$:

$$u(x) \approx 1 - \frac{1}{K} \sum_{k=1}^{K} P(y = c \mid \hat{x}_k), \quad (4)$$

where K presents the number of the perturbations generated by perturb methods such as Audio Shift [33], Audio PitchShift [33] and Audio Colored Noise [34, 35]. A larger $u(x)$ indicates a smaller confidence of the model in predicting the perturbed samples. The third step selects L examples from D_c through descending the uncertainty $u(x)$ with the step size of $len(D_c) * C/L$, where L is the size of the replay buffer.

Previous research [24] demonstrated that the uncertainty memory update algorithm performs better than the other three algorithms on speech tasks such as keyword spotting. However, the computation cost of *Uncertainty* increases linearly with the number of perturbation operations.

2.2. Proposed MUA method (*Uncertainty++*)

As illustrated in Figure 1, the native uncertainty memory update algorithm requires to employ perturbation methods offline for the waveform of each sample to generate the perturbed samples first. In our proposed method, noisy perturbations are added to the pre-classifier embedding of the sample, and not to the waveform, so the output of the backbone of the model is calculated only once. Specifically, we propose a vector-wise perturbation method that adds noise with different intensities according to the variance of classifier's embedding. We denote the perturbed version of the classifier's embedding e as $\hat{e}$, which is computed as follows:

$$\hat{e} = e + U(-\frac{\lambda}{2}, \frac{\lambda}{2}) * std(e), \tag{5}$$

where $std(\cdot)$ stands for standard deviation, the function $U(a, b)$ represents the noise distributed uniformly from a to b, $U(a, b)$ is a vector with the same shape as e, and λ is a hyperparameter that controls the relative noise intensity.

By the vector-wise perturbation method, we generate the perturbed embedding $\hat{e}$ of the embedding e. Finally, we input $\hat{e}$ to the final classification layer of the model and output $P(y = c \mid \hat{e})$ which is used to compute the uncertainty as in Eq. (3). After the uncertainty is estimated, we select examples for replay as native approach. This method saves time by calculating the output of the backbone of the model only once. We also save the memory usage by replacing the perturbed raw data with the classifier's embedding which is of much smaller size as compared with the raw data.

3. EXPERIMENTS

3.1. Environmental sound classification model

For the on-device environmental sound classification model, we use BC-ResNet-Mod [29] which is an adaptation of the BC-ResNet [28] that achieves improved results on acoustic scene classification. The BC-ResNet paradigm works via repeatedly extracting spectral and then temporal features in series. Because these spectral features are of a lower dimension than the input, this model has fewer parameters than one that processes the waveform directly. Feature extraction is channel-wise, and both parameter reductions have negligible impact on performance [28]. For our experiments, we use BC-ResNet-Mod-4, which increases the input channel dimension to 80 before extracting spectral and temporal features.

3.2. Datasets

ESC-50 consists of 2000 five-second environmental audio recordings [27]. Data are balanced between 50 classes, with 40 examples per class, covering animal sounds, natural soundscapes, human sounds (non-speech), and ambient noises. The dataset has been prearranged into five folds for cross-validation.

DCASE 2019 Task 1 is an acoustic scene classification task, with a development set [26] consisting of 10-second audio segments from 10 acoustic scenes: airport, indoor shopping mall, metro station, pedestrian street, public square, the street with a medium level of traffic, traveling by tram, traveling by bus, traveling by an underground metro and urban park. In the development set, there are 9185 and 4185 audio clips for training and validation, respectively.

Table 1: *Accuracy (ACC) and Backward Transfer (BWT) in a comparative study of the proposed memory update algorithm.*

Method	DCASE 2019 Task 1		ESC-50	
	ACC ↑	BWT ↑	ACC ↑	BWT ↑
Finetune	0.205	-0.276	0.181	-0.307
Random	0.473	-0.115	0.225	-0.231
Reservoir	0.568	-0.096	0.430	-0.121
Prototype	0.559	-0.089	0.482	**-0.104**
Uncertainty	0.578	**-0.079**	0.477	-0.111
Uncertainty++	**0.581**	**-0.079**	**0.500**	-0.121

3.3. Experimental setup

Task setting To evaluate the performance of the proposed approach, we split the data into five tasks. Each task includes 2 new unique classes in DCASE 19 Task 1 and 10 new unique classes in ESC-50, which is unseen in previous tasks. To simulate the condition of edge devices, we set the buffer size L of examples as 500, 100 samples in DCASE 19 Task 1 and ESC-50 due to the memory limitation.

Implementation details The original audio clip is converted to 64-dimensional log Mel-spectrogram by using the short-time Fourier transform with a frame size of 1024 samples, a hop size of 320 samples, and a Hanning window. The classification network is optimized by the Adam [36] algorithm with the learning rate 1×10^{-3}. The batch size is set to 32 and the number of epochs is 50.

3.4. Evaluation metrics

We report performances in terms of the accuracy and forgetting metric. Specifically, the *Accuracy* (ACC) reports an accuracy averaged on learned classes after the entire training ends. The *Backward Transfer* (BWT) [37] evaluates accuracy changes on all previous tasks after learning a new task, indicating the forgetting degree. For measuring BWT, we first construct the matrix $R \in \mathbb{R}^{T \times T}$, where $R_{i,j}$ is the test classification accuracy of the model on task τ_j after observing the last sample from task τ_i. After the model finished learning about each task τ_i, we evaluate its BWT on all T tasks, which is formulated as:

$$BWT = \frac{1}{T-1} \sum_{i=1}^{T-1} R_{T,i} - R_{i,i}. \tag{6}$$

There exists negative BWT when learning about some task decreases the performance on some preceding task. A smaller value of BWT indicates a higher catastrophic forgetting.

3.5. Reference baselines

We built five baselines for comparisons. The *Finetune* training strategy adapts the BC-ResNet-Mod model for each new task without any continual learning strategies, as the lower-bound baseline. The four prior memory update algorithms of replay-based continual learning (i.e., *Random*, *Reservoir*, *Prototype*, *Uncertainty*) are introduced in Section 2.1. Specifically, at the perturbation stage of the uncertainty, we use two perturbation methods, namely, '*uncertainty-shift*', which

Table 2: *Accuracy (ACC) and Backward Transfer (BWT) in a comparative study of the proposed perturbation method. The K refers to the number of the perturbations generated by perturbation methods.*

Method	K	DCASE 2019 Task 1		ESC-50	
		ACC ↑	BWT ↑	ACC ↑	BWT ↑
Uncertainty-Shift	2	0.557	-0.101	0.461	**-0.111**
	4	0.575	-0.103	0.476	-0.118
	6	0.567	-0.079	0.477	-0.118
Uncertainty-Noise	2	0.560	-0.100	0.465	-0.118
	4	0.535	-0.104	0.473	-0.118
	6	0.578	-0.079	0.458	-0.120
Uncertainty++	2	0.571	-0.102	**0.500**	-0.121
	4	0.548	-0.103	0.481	-0.114
	6	**0.581**	**-0.079**	0.484	-0.119

includes Audio Shift and Audio PitchShift, and '*uncertainty-noise*' which refers to the Audio Colored Noise perturbation method.

4. RESULTS

4.1. Experiments on MUA methods

Table 1 presents the results on DCASE 2019 Task 1 and ESC-50 test set in terms of ACC and BWT. We compare the proposed *Uncertainty++* MUA method with five baselines. We observe that the *uncertainty* MUA method achieves better performance than the five baselines. Comparing with the best baseline *uncertainty*, we observe that the proposed *uncertainty++* method obtains 58.1% on classification accuracy which outperforms the existing MUA methods. In addition, we observe that the *Finetune* method achieves the worst ACC and BWT performance compared with other baselines, which indicates the issue of catastrophic forgetting.

We further analyze and summarize the performances of the proposed *uncertainty++* method compared with the *uncertainty* MUA method with different numbers of the perturbation methods in terms of ACC and BWT as shown in Table 2. The K refers to the number of the perturbations generated by perturb methods. Even with only two perturbation methods, our proposed method still outperforms other two baselines. We also observe that our method under two perturbations obtains the best performance on the ESC-50 test set. Such performance might be due to the small size of the ESC-50, therefore it is more sensitive to perturbations.

4.2. Comparative experiments on computation time for *Uncertainty* and *Uncertainty++*

We further report the Average Time for the proposed method when there is an increasing number of perturbations. The Average Time measures a relative time increase compared to training time in each task. As shown in Table 3, even with 6 perturbations, the Average Time of the *uncertainty++* is still less than 60s. This can be explained by the fact that our proposed method can limit the growth of the additional training time. We also observe that our proposed method outperforms other baselines in any number of perturbations, which indicates our proposed method is computationally more efficient. In addition, the average time of *uncertainty-shift* is much longer than others. Because the Audio Shift and Audio PitchShift perturbations takes more time than simply adding noise.

Table 3: *Average Time (s) in a comparative study of the proposed uncertainty++ method. The K refers to the number of the perturbations generated by perturbation methods.*

Method	K	Average Time (s) ↓
Uncertainty-Shift	2	1221.7
	4	2205.1
	6	2926.1
Uncertainty-Noise	2	246.2
	4	390.8
	6	506.3
Uncertainty++	2	44.0
	4	48.5
	6	55.1

5. CONCLUSIONS

In this work, we have presented *uncertainty++*, an efficient replay-based continual learning method for on-device environmental sound classification. Our method selects the historical data for the training by measuring the per-sample classification uncertainty on the embedding layer of the classifier. Experimental results on the DCASE 2019 Task 1 and ESC-50 datasets show that our proposed method outperforms the baseline continual learning methods on classification accuracy and computational efficiency. In future work, we plan to apply and adapt our approach to other on-device audio classification tasks such as audio tagging and sound event detection.

6. ACKNOWLEDGMENT

This work is partly supported by UK Engineering and Physical Sciences Research Council (EPSRC) Grant EP/T019751/1 "AI for Sound", a Newton Institutional Links Award from the British Council, titled "Automated Captioning of Image and Audio for Visually and Hearing Impaired" (Grant number 623805725), a Research Scholarship from the China Scholarship Council (CSC), and a PhD studentship from the Engineering and Physical Sciences Research Council (EPSRC) Doctoral Training Partnership EP/T518050/1. For the purpose of open access, the authors have applied a Creative Commons Attribution (CC BY) licence to any Author Accepted Manuscript version arising.

7. REFERENCES

[1] K. J. Piczak, "Environmental sound classification with convolutional neural networks," in *Proc. IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP)*, 2015, pp. 1–6.

[2] A. Singh, J. A King, X. Liu, W. Wang, and M. D. Plumbley, "Low-complexity CNNs for acoustic scene classification," DCASE2022 Challenge, Tech. Rep., June 2022.

[3] A. Singh and M. D. Plumbley, "A passive similarity based cnn filter pruning for efficient acoustic scene classification," *arXiv preprint:2203.15751*, 2022.

[4] K. Choi, M. Kersner, J. Morton, and B. Chang, "Temporal knowledge distillation for on-device audio classification," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022, pp. 486–490.

[5] I. Martín-Morató, F. Paissan, A. Ancilotto, T. Heittola,

A. Mesaros, E. Farella, A. Brutti, and T. Virtanen, "Low-complexity acoustic scene classification in DCASE 2022 Challenge," *arXiv preprint:2206.03835*, 2022.

[6] R. Radhakrishnan, A. Divakaran, and A. Smaragdis, "Audio analysis for surveillance applications," in *Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics, 2005.*, 2005, pp. 158–161.

[7] H. Liu, X. Liu, X. Mei, Q. Kong, W. Wang, and M. D. Plumbley, "Surrey system for DCASE 2022 task 5 : Few-shot bioacoustic event detection with segment-level metric learning technical report," DCASE2022 Challenge, Tech. Rep., June 2022.

[8] S. Kiranyaz, A. F. Qureshi, and M. Gabbouj, "A generic audio classification and segmentation approach for multimedia indexing and retrieval," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14, no. 3, pp. 1062–1081, 2006.

[9] J. Salamon and J. P. Bello, "Deep convolutional neural networks and data augmentation for environmental sound classification," *IEEE Signal processing letters*, vol. 24, no. 3, pp. 279–283, 2017.

[10] J. Sun, X. Liu, X. Mei, J. Zhao, M. D. Plumbley, V. Kılıç, and W. Wang, "Deep neural decision forest for acoustic scene classification," *arXiv preprint:2203.03436*, 2022.

[11] N. Hou, C. Xu, E. S. Chng, and H. Li, "Domain adversarial training for speech enhancement," in *Proc. Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*. IEEE, 2019, pp. 667–672.

[12] N. Hou, C. Xu, J. T. Zhou, E. S. Chng, and H. Li, "Multi-task learning for end-to-end noise-robust bandwidth extension," in *Proc. Interspeech*, 2020, pp. 4069–4073.

[13] M. McCloskey and N. J. Cohen, "Catastrophic interference in connectionist networks: The sequential learning problem," in *Psychology of Learning and Motivation*, 1989, vol. 24, pp. 109–165.

[14] A. Awasthi and S. Sarawagi, "Continual learning with neural networks: A review," in *Proc. the ACM India Joint International Conference on Data Science and Management of Data*, 2019, pp. 362–365.

[15] H. Zhang, M. Shen, Y. Huang, Y. Wen, Y. Luo, G. Gao, and K. Guan, "A serverless cloud-fog platform for DNN-based video analytics with incremental learning," *arXiv preprint:2102.03012*, 2021.

[16] Y. Huang, H. Zhang, Y. Wen, P. Sun, and N. B. D. Ta, "Modelcie: Enabling continual learning in deep learning serving systems," *arXiv preprint:2106.03122*, 2021.

[17] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, *et al.*, "Overcoming catastrophic forgetting in neural networks," *the National Academy of Sciences*, vol. 114, no. 13, pp. 3521–3526, 2017.

[18] F. Zenke, B. Poole, and S. Ganguli, "Continual learning through synaptic intelligence," in *Proc. International Conference on Machine Learning*, 2017, pp. 3987–3995.

[19] Y.-C. Hsu, Y.-C. Liu, A. Ramasamy, and Z. Kira, "Re-evaluating continual learning scenarios: A categorization and case for strong baselines," *arXiv preprint:1810.12488*, 2018.

[20] S.-A. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert, "iCaRL: Incremental classifier and representation learning," in *Proc. the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2001–2010.

[21] J. S. Vitter, "Random sampling with a reservoir," *ACM Transactions on Mathematical Software (TOMS)*, vol. 11, no. 1, pp. 37–57, 1985.

[22] J. Bang, H. Kim, Y. Yoo, J.-W. Ha, and J. Choi, "Rainbow memory: Continual learning with a memory of diverse samples," in *Proc. the IEEE Conference on Computer Vision and Pattern Recognition*, 2021, pp. 8218–8227.

[23] Y. Huang, N. Hou, and N. F. Chen, "Progressive continual learning for spoken keyword spotting," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022, pp. 7552–7556.

[24] Y. Xiao, N. Hou, and E. S. Chng, "Rainbow keywords: Efficient incremental learning for online spoken keyword spotting," *arXiv preprint:2203.16361*, 2022.

[25] Z. Wang, C. Subakan, E. Tzinis, P. Smaragdis, and L. Charlin, "Continual learning of new sound classes using generative replay," in *Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, 2019, pp. 308–312.

[26] T. Heittola, A. Mesaros, and T. Virtanen, "TAU Urban Acoustic Scenes 2019, Development dataset," Mar. 2019. [Online]. Available: https://doi.org/10.5281/zenodo.2589280

[27] K. J. Piczak, "Esc: Dataset for environmental sound classification," in *Proc. the ACM international conference on Multimedia*, 2015, pp. 1015–1018.

[28] B. Kim, S. Chang, J. Lee, and D. Sung, "Broadcasted residual learning for efficient keyword spotting," *arXiv preprint:2106.04140*, 2021.

[29] B. Kim, S. Yang, J. Kim, and S. Chang, "QTI submission to DCASE 2021: Residual normalization for device-imbalanced acoustic scene classification with efficient design," DCASE2021 Challenge, Tech. Rep., June 2021.

[30] Y.-C. Hsu, Y.-C. Liu, A. Ramasamy, and Z. Kira, "Re-evaluating continual learning scenarios: A categorization and case for strong baselines," *arXiv preprint:1810.12488*, 2018.

[31] Y. Gal and Z. Ghahramani, "Dropout as a Bayesian approximation: Representing model uncertainty in deep learning," 2016.

[32] T. Kloek and H. K. Van Dijk, "Bayesian estimates of equation system parameters: An application of integration by monte carlo," *Econometrica: Journal of the Econometric Society*, pp. 1–19, 1978.

[33] T. Ko, V. Peddinti, D. Povey, and S. Khudanpur, "Audio augmentation for speech recognition," in *Proc. Interspeech*, 2015.

[34] I. Jordal, S. ES, H. BREDIN, K. Nishi, F. Lata, H. C. Blum, P. Manuel, akash raj, K. Choi, FrenchKrab, P. Żelasko, amiasato, M. L. Quatra, and E. Schmidbauer, "asteroid-team/torch-audiomentations: v0.11.0," June 2022. [Online]. Available: https://doi.org/10.5281/zenodo.6778064

[35] S. Kamath and P. Loizou, "A multi-band spectral subtraction method for enhancing speech corrupted by colored noise," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2002.

[36] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint:1412.6980*, 2014.

[37] D. Lopez-Paz and M. Ranzato, "Gradient episodic memory for continual learning," in *Proc. Advances in neural information processing systems*, vol. 30, 2017.

LANGUAGE-BASED AUDIO RETRIEVAL TASK IN DCASE 2022 CHALLENGE

Huang Xie, Samuel Lipping, Tuomas Virtanen

Audio Research Group, Tampere University, Tampere, Finland

ABSTRACT

Language-based audio retrieval is a task, where natural language textual captions are used as queries to retrieve audio signals from a dataset. It has been first introduced into DCASE 2022 Challenge as Subtask 6B of task 6, which aims at developing computational systems to model relationships between audio signals and free-form textual descriptions. Compared with audio captioning (Subtask 6A), which is about generating audio captions for audio signals, language-based audio retrieval (Subtask 6B) focuses on ranking audio signals according to their relevance to natural language textual captions. In DCASE 2022 Challenge, the provided baseline system for Subtask 6B was significantly outperformed, with top performance being 0.276 in mAP@10. This paper presents the outcome of Subtask 6B in terms of submitted systems' performance and analysis.

Index Terms— Language-based audio retrieval, DCASE 2022 Challenge, Clotho.

1. INTRODUCTION

With the growth of multimedia content in recent decades, there is a need for retrieval methods that can efficiently organize the data based on its content, and retrieve relevant items when doing searches to datasets. Natural language provides an efficient way to represent complex information about multimedia. It can represent high level information about data that goes beyond any fixed taxonomies. For audio signals, natural language can represent information related to temporal relationships between sound sources, and attributes of sounds and their environment.

Language-based multimedia retrieval has received increasing attention in recent years. The majority of recent works has focused heavily on the visual domain [1, 2]. For example, there are plenty of approaches [3] tackling content-based image retrieval with free-form textual descriptions. In contrast, only a few studies have been conducted on language-based audio retrieval in the existing literature. Early works [4, 5] deal with language-based audio retrieval using multi-word text queries consisting of audio tags or class labels, rather than sentence-like textual descriptions, e.g., captions. Recent studies [6, 7] prompt research in this field by exploring human written captions as queries. In DCASE 2022 Challenge, language-based audio retrieval is introduced into as Subtask 6B, which aims to inspire further research into audio retrieval with unconstrained textual descriptions.

In this paper, we present the task setup and submissions for Subtask 6B of task 6 in DCASE 2022 Challenge. We introduce the datasets for system development and evaluation, describe the baseline system for Subtask 6B, and present the challenge submissions. Evaluation and analysis of submitted systems includes general statistics on systems and performance and system characteristics.

The remainder of this paper is organized as follows. In Section 2, we describe the task setup, including task description, task datasets, and evaluation metrics. Then, we introduce the task baseline system in Section 3. We present the evaluation results and analysis of challenge submissions in Section 4. Finally, we conclude this paper in Section 5.

2. TASK SETUP

In this section, we introduce the task description, datasets for system development and evaluation, and evaluation metrics.

2.1. Task Description

Language-based audio retrieval is concerned with retrieving audio signals using their sound content textual descriptions (i.e., audio captions). With this task, the goal is to evaluate audio retrieval methods, where a retrieval system takes an audio caption as a text query and ranks audio signals in a fixed dataset according to their relevance to the caption. In DCASE 2022 Challenge, human-written natural language audio captions are used as text queries. For each query, the retrieval task is to retrieve 10 audio files from a given evaluation dataset and sort them according to their relevance to the query.

2.2. Development Dataset

The Clotho v2 [8] is provided as the task development dataset, which consists of audio samples of 15 to 30 seconds duration, with each audio sample having five captions of eight to 20 words length. There are 6,974 audio samples with 34,870 captions in total. All audio samples are sourced from the Freesound platform [9], and captions are crowd-sourced using a three-step framework [8].

The Clotho v2 [8] is divided into a training split of 3,839 audio clips with 19,195 captions, a validation split of 1,045 audio clips with 5,225 captions, and a testing split of 1,045 audio clips with 5,225 captions. These splits are created by first constructing the sets of unique words of the captions of each audio clip. These sets of words are combined to form the bag of words of the whole dataset, from which the frequency of a given word can be derived. With the unique words of audio files as classes, multi-label stratification is applied. The data collecting procedure is explained in detail in [8].

2.3. Clotho Retrieval Evaluation Dataset

The task evaluation dataset consists of 1,000 audio samples sourced from the Freesound platform [9], and one human written caption is provided for each audio sample. The audio samples are collected following the procedure described in [8], by optimizing the tag distribution of the selected samples. The samples were selected from the set of files not used in Clotho v2. Captions were gathered using

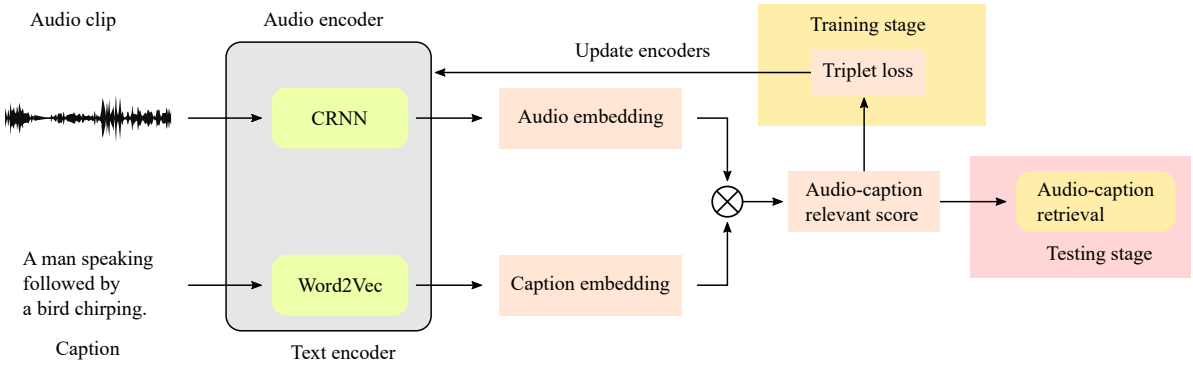


Figure 1: The baseline system for the language-based audio retrieval subtask.

Dataset	Split	#Audio	#Captions
Development	Training	3839	19195
	Validation	1045	5225
	Testing	1045	5225
Evaluation		1000	1000

Table 1: Statistics of the development and evaluation datasets.

the first crowd-sourcing step of Clotho v2 manually screened for typographical errors and speech transcription. Table 1 summarizes the information about the development and evaluation datasets.

2.4. Evaluation Metrics

In the evaluation, the ground truth relevance of audio samples are considered binary (i.e., only the audio samples belong to the caption query are considered relevant, and all the others not relevant). The submissions for this task will be evaluated using mean average precision at top-10 (mAP@10) as the main metric, and recall at k (R@k with $k \in \{1, 5, 10\}$) as the secondary metrics.

The mAP metric has been widely used for evaluating the performance of cross-modal retrieval algorithms [3]. It is a rank-aware metric, which measures the mean of average precisions (AP) over all the queries. The AP for a query is calculated by averaging the precisions at positions, where relevant items are in the retrieved rank list. The more relevant items in the top rank list, the higher mAP value it has. The R@k metric is another standard, rank-unaware retrieval metric [6], which is defined as the proportion of relevant items among the top-K retrieved results to all the relevant items in the evaluation dataset, averaged across all the caption queries. The challenge submissions will be ranked by the mAP@10 metric.

3. BASELINE SYSTEM

In this section, we describe the task baseline system, as illustrated in Figure 1. The baseline system is a simplified version of the audio-text aligning framework presented in [7], which calculates relevant scores between encoded textual descriptions (i.e., encoded captions) and encoded audio signals. It consists of two input encoders: one for audio, and the other for text, as illustrated in Figure 1. These two modality-specific encoders generate vector representations (i.e., audio embeddings and caption embeddings) for audio clips and textual descriptions. Then, the relevance score between an audio clip and a textual description is calculated by the dot product of their vector representations. The baseline system is optimized with a sampling-based triplet loss [19] at the training stage, and then applied to retrieve audio for caption queries at the testing stage.

3.1. Audio Encoder

A convolutional recurrent neural network (CRNN) [20] is used as the audio encoder, which extracts frame-wise acoustic embeddings from audio signals. It consists of five convolution blocks, followed by a bidirectional gated recurrent unit (BiGRU). Each convolution block includes an initial batch normalization, a convolutional layer with padded 3×3 convolutions, and a LeakyReLU activation with a slope of -0.1. After the first, third, and fifth convolution blocks, one L4-Norm subsampling layer is used to reduce the temporal dimension of each block's output by a factor of four. A dropout layer with a rate of 0.3 is placed between the last L4-Norm layer and the BiGRU. Lastly, an up-sampling operation is applied to ensure the final output has the same temporal dimension as the CRNN input.

The CRNN audio encoder takes 64-dimensional log mel-band energies as input. Each audio clip is split into 40 ms Hanning-windowed frames with a hop length of 20 ms. Then, 64 log mel-band coefficients are extracted from each frame. A sequence of 300-dimensional frame-wise acoustic embeddings are generated for each audio clip. The final audio embedding is calculated by averaging the frame-wise acoustic embeddings.

3.2. Text Encoder

Word2Vec (Skip-gram model) [21] is utilized as the text encoder to convert textual descriptions into sequences of word embeddings. It is a two-layer fully-connected neural network, which learns word embeddings that are good at predicting surrounding words in a sentence or a document. For the sake of simplicity, we adopt a publicly available pre-trained Word2Vec [22], which is trained on Google News dataset. It consists of 300-dimensional word embeddings for roughly three million case-sensitive English words and phrases. The Word2Vec text encoder converts textual descriptions into sequences of semantic word embeddings word by word. The final caption embedding is computed by averaging the word embeddings.

Rank	Team	**mAP@10** with 95% CI		**R@1** with 95% CI		**R@5** with 95% CI		**R@10** with 95% CI	
1	Xu et al. [10]	0.276	[0.254, 0.299]	0.176	[0.152, 0.200]	0.416	[0.385, 0.447]	0.536	[0.505, 0.567]
2	Mei et al. [11]	0.251	[0.229, 0.273]	0.153	[0.131, 0.175]	0.387	[0.357, 0.417]	0.504	[0.473, 0.535]
3	Lamort et al. [12]	0.221	[0.200, 0.242]	0.131	[0.110, 0.152]	0.343	[0.314, 0.372]	0.466	[0.435, 0.497]
4	Pellegrini [13]	0.216	[0.195, 0.237]	0.127	[0.106, 0.148]	0.321	[0.292, 0.350]	0.463	[0.432, 0.494]
5	Lai et al. [14]	0.215	[0.194, 0.235]	0.122	[0.102, 0.142]	0.328	[0.299, 0.357]	0.478	[0.447, 0.509]
6	Wu et al. [15]	0.188	[0.168, 0.207]	0.107	[0.088, 0.126]	0.303	[0.275, 0.331]	0.413	[0.382, 0.444]
7	Weck et al. [16]	0.128	[0.111, 0.145]	0.077	[0.060, 0.094]	0.188	[0.164, 0.212]	0.284	[0.256, 0.312]
8	Xiao et al. [17]	0.097	[0.083, 0.111]	0.043	[0.030, 0.056]	0.162	[0.139, 0.185]	0.267	[0.240, 0.294]
9	Park et al. [18]	0.075	[0.063, 0.088]	0.033	[0.022, 0.044]	0.127	[0.106, 0.148]	0.208	[0.183, 0.233]
10	Baseline	0.061	[0.049, 0.072]	0.026	[0.016, 0.036]	0.102	[0.083, 0.121]	0.176	[0.152, 0.200]

Table 2: Evaluation results with 95% confidence intervals for the top system of each team.

3.3. Training Objective

The baseline system is trained by optimizing a ranking-based criterion [19], such that audio clips and captions that belong together are more similar in the embedding space than mismatched audio-caption pairs. Specifically, across a batch of N audio-caption pairs $\{(x_n, y_n)\}_{n=1}^{N}$, where y_n is the caption pertaining to an audio clip x_n, we randomly select an imposter clip $\hat{x}_n$ and an imposter caption $\hat{y}_n$ for each audio-caption pair (x_n, y_n). Then, the widely used sampling-based triplet loss [7, 23] is calculated by

$$loss = \frac{1}{N}\sum_{n=1}^{N}[\max(0, S(x_n, \hat{y}_n) - S(x_n, y_n) + 1) + \max(0, S(\hat{x}_n, y_n) - S(x_n, y_n) + 1)], \quad (1)$$

where S is the audio-caption relevance score.

3.4. Baseline Results

The baseline system is trained with batches of 32 audio-caption pairs in the training split for at most 150 epochs, while monitoring the loss (1) on the validation split during the training process. An Adam optimizer with an initial learning rate of 0.001 is adopted to optimize the training process. The learning rate is reduced by a factor of ten once the validation loss does not improve for five epochs. Training is terminated by early stopping with ten epochs.

As shown in Table 3, the baseline system achieves similar performance in terms of mAP@10 and recall scores on the testing split and the evaluation dataset. Specifically, with the evaluation dataset, the theoretical chance levels are $1/1000 = 0.001$ for R@1, $1/200 = 0.005$ for R@5, and $1/100 = 0.01$ for R@10, respectively. In contrast to the theoretical chance levels, the baseline system obtains better recall scores, with an R@1 / R@5 / R@10 of 0.026 / 0.102 / 0.176. The experimental results show that the baseline system can retrieve audio with their corresponding captions, i.e., perform language-based audio retrieval. On the other hand, since the baseline system employs a simple pipeline (e.g., averaging frame-wise acoustic embeddings and word embeddings), the retrieval performance remains limited.

4. CHALLENGE SUBMISSIONS

In this section, we present the evaluation results and analysis of the submissions for language-based audio retrieval.

Dataset	mAP@10	R@1	R@5	R@10
Testing split	0.068	0.032	0.109	0.188
Evaluation	0.061	0.026	0.102	0.176

Table 3: Baseline results on the testing split and the evaluation dataset.

4.1. Evaluation Results

The task received a total number of 31 submissions from nine teams, with maximum four submissions per team allowed. Table 2 shows the results of evaluation metrics (e.g., mAP@10) with 95% confidence intervals (CIs) for the top system of each team, comparing with the baseline system. The 95% confidence intervals for evaluation metrics are calculated using the Jackknife estimate [24].

All the submitted systems outperformed the baseline system in terms of mAP@10 and R@k with $k \in \{1, 5, 10\}$. Particularly, Xu et al. [10] achieved the best performance, with a mAP@10 of 0.276 and a 95% CI between 0.254 and 0.299. Mei et al. [11] ranked second, with their best system having a mAP@10 of 0.251 (95% CI $[0.229, 0.273]$). The top two teams obtained mAP@10 over 0.250 and R@10 over 0.500, in contrast to those of teams ranked third - fifth, having mAP@10 around 0.220 and R@10 around 0.470. Out of the nine teams, six teams achieved mAP@10 over 0.180, R@5 over 0.300, and R@10 over 0.410. The baseline system ranked last with a significantly lower performance.

4.2. Analysis of Submissions

The following analysis is based on the information reported by participating teams.

System summary. All the 31 submitted systems and the baseline system adopted a bi-encoder architecture, which consisted of an audio encoder and a caption encoder, to associate audio with captions. For audio encoders, pre-trained convolutional neural networks (e.g., PANNs [25]) were the most common choice (27 systems from eight teams) among the nine participating teams. For caption encoders, pre-trained Transformer-based language embedding models (e.g., BERT [26] and Sentence-BERT [34]) were frequently employed (28 systems from seven teams). Then, audio and captions were encoded into a common embedding space, where their relevance was scored with cosine similarity (24 systems from six teams) or dot product (seven systems from three teams) of their embeddings. Most of the submitted systems (23 systems from seven

Rank	Team	Audio Modelling	Caption Modelling	Loss Function
1	Xu et al. [10]	PANNs [25]	BERT [26], RoBERTa [27]	InfoNCE loss [28]
2	Mei et al. [11]	PANNs [25]	BERT [26]	NT-Xent loss [29]
3	Lamort et al. [12]	PANNs [25], OpenL3 [30], VGGSound [31], VGGish [32], BART [33]	Sentence-BERT [34]	Triplet loss [6, 7]
4	Pellegrini [13]	PaSST [35]	Sentence-BERT [34]	Triplet loss [6, 7]
5	Lai et al. [14]	ESResNet [36]	Transformer [37]	CE loss [37]
6	Wu et al. [15]	PANNs [25], HTS-AT [38]	Transformer [37]	CE loss [37]
7	Weck et al. [16]	PANNs [25]	RoBERTa [27]	Contrastive loss [39]
8	Xiao et al. [17]	PANNs [25]	Word2Vec [21]	Triplet loss [6, 7]
9	Park et al. [18]	CRNN [20]	Word2Vec [21]	Triplet loss [6, 7]
10	Baseline	CRNN [20]	Word2Vec [21]	Triplet loss [6, 7]

Table 4: Summary of systems with best performance from all teams.

teams and the baseline) were trained by optimizing some contrastive losses (e.g., InfoNCE loss [28] and triplet loss [6, 7]).

Audio encoders. Among the submitted systems, PANNs [25] (e.g., CNN14, Wavegram-Logmel-CNN14 and CNN10) were the most common choice for encoding audio data. There were in total 21 systems from six teams utilizing PANNs as their audio encoders, including the top three teams [10, 11, 12]. Particularly, the CNN14 model was most preferred (20 systems). Xu et al. [10] experimented with multiple pre-trained audio expert models, including CNN14 and Wavegram-Logmel-CNN14. With ensemble of different models, they achieved the best performance. Mei et al. [11] adopted CNN14 as their audio encoder without model ensembles, having their best system ranked second. Tied on third rank, Lamort et al. [12] experimented with aggregating the most number of pre-trained audio expert models (i.e., five models in total, as shown in Table 4). The audio encoders utilized in other systems were PaSST [35], ESResNet [36], HTS-AT [38], and CRNN [18].

Log-mel energies were frequently taken as input among the submitted systems. A total number of 28 systems from eight teams utilized log-mel energies as standalone audio features or in combination with other features (e.g., raw waveform [10]). Other audio features used included log-magnitude spectrogram [12] and log-power spectrogram [14].

Caption encoders. Most of the participating teams (i.e., seven out of nine) preferred Transformer-based language embedding models (e.g., BERT [26]) over word embedding models (e.g., Word2Vec [21]), to learn caption embeddings. As shown in Table 4, pre-trained Transformer-based language embedding models, including BERT [26], RoBERTa [27], and Sentence-BERT [34], were utilized as caption encoders by seven teams (28 systems in total). The other two teams and the baseline converted captions into embeddings with pre-trained Word2Vec word vectors [21]. All teams with Transformer-based models ranked higher than those with pre-trained Word2Vec [21], which showed that Transformer-based models learned caption embeddings more efficiently.

Loss functions. The loss functions utilized in the submissions can be categorized into three groups: contrastive loss (one team), triplet loss (four teams), and N-pair loss (four teams). Contrastive loss [40] takes pairs of samples as input and measures the similarity between two inputs. Triplet loss [6, 7] takes as input triplets of samples, consisting of one anchor sample along with one positive and one negative sample. Compared to contrastive loss, triplet loss focuses on the difference of (dis-)similarities between positive and negative samples to the anchor sample. N-pair loss [40] shares a similar spirit to triplet loss, but extends to multiple negative samples, i.e., one positive and multiple negative samples for an anchor sample. In practice, several versions of N-pair loss were adopted, including InfoNCE loss [28], NT-Xent loss [29], and symmetric cross-entropy (CE) loss [37].

Data augmentation and external data. Data augmentation was adopted by only three participating teams (i.e., Mei et al. [11] on rank two, Lamort et al. [12] on rank three, and Wu et al. [15] on rank six) during their systems' training. In contrast, external audio and textual data was leveraged for optimizing system performance by most participating teams (i.e., seven out of nine). The submitted systems were usually pre-trained with a large amount of external audio and textual data, and then fine-tuned on the task development dataset. For example, the top two teams (i.e., Xu et al. [10] and Mei et al. [11]) pre-trained their systems with AudioCaps [41]. Additionally, Wu et al. [15] trained their systems by involving various external data, including audio samples and text (e.g., labels, file names and captions) from seven external datasets in addition to Clotho v2 [8].

5. CONCLUSIONS

The DCASE 2022 Challenge has introduced the language-based audio retrieval task, which is about using natural language textual captions as queries to retrieve audio signals from a dataset. This paper describes setups of the task, including task description, datasets for system development and evaluation, evaluation metrics, and a task baseline system. Moreover, this paper reports the final evaluation results of the task submissions and their analysis. The evaluation results show that all submitted systems outperformed the baseline in terms of evaluation metrics, with top performance being 0.276 in mAP@10 and 0.536 in R@10.

One immediate observation about the submitted systems is that all of them adopted a bi-encoder architecture consisting of an audio encoder and a caption encoder. Pre-trained convolutional neural networks (e.g., PANNs [25]) were commonly employed as audio encoders, and pre-trained Transformer-based language embedding models (e.g., BERT [26]) were the preferred caption encoders. In addition to the task development dataset, external audio and textual data (e.g., labels and captions) was also frequently leveraged during system training.

6. REFERENCES

[1] J. Dong, X. Li, C. Xu, X. Yang, G. Yang, X. Wang, and M. Wang, "Dual encoding for video retrieval by text," *IEEE Trans. Pattern Anal. Mach. Intell.*, pp. 1–10, 2021.

[2] L. Qu, M. Liu, J. Wu, Z. Gao, and L. Nie, "Dynamic modality interaction modeling for image-text retrieval," in *Proc. ACM SIGIR Int. Conf. Res. Dev. Inf. Retr.*, 2021, pp. 1104–1113.

[3] P. Kaur, H. S. Pannu, and A. K. Malhi, "Comparative analysis on cross-modal information retrieval: A review," *Comput. Sci. Rev.*, p. 100336, 2021.

[4] G. Chechik, E. Ie, M. Rehn, S. Bengio, and D. Lyon, "Large-scale content-based audio retrieval from text queries," in *Proc. Int. Conf. Multimed. Inf. Retr.*, 2008, pp. 105–112.

[5] B. Elizalde, S. Zarar, and B. Raj, "Cross modal audio search and retrieval with joint embeddings based on text and audio," in *Proc. IEEE Int. Conf. Acoustic., Speech and Signal Process. (ICASSP)*, 2019, pp. 4095–4099.

[6] A. M. Oncescu, A. Sophia Koepke, J. F. Henriques, Z. Akata, and S. Albanie, "Audio retrieval with natural language queries," in *Proc. Annu. Conf. Int. Speech Commun. Assoc. (INTERSPEECH)*, 2021, pp. 2411–2415.

[7] H. Xie, O. Räsänen, K. Drossos, and T. Virtanen, "Unsupervised audio-caption aligning learns correspondences between individual sound events and textual phrases," in *Proc. IEEE Int. Conf. Acoustic., Speech and Signal Process. (ICASSP)*, 2022, pp. 8867–8871.

[8] K. Drossos, S. Lipping, and T. Virtanen, "Clotho: an audio captioning dataset," in *Proc. IEEE Int. Conf. Acoustic., Speech and Signal Process. (ICASSP)*, 2020, pp. 736–740.

[9] F. Font, G. Roma, and X. Serra, "Freesound technical demo," in *Proc. Int. Conf. Multimed.*, 2013, pp. 411–412.

[10] X. Xu, Z. Xie, M. Wu, and K. Yu, "The SJTU system for DCASE2022 challenge task 6: Audio captioning with audio-text retrieval pre-training," DCASE2022 Challenge, Tech. Rep., 2022.

[11] X. Mei, X. Liu, H. Liu, J. Sun, M. D. Plumbley, and W. Wang, "Language-based audio retrieval with pre-trained models," DCASE2022 Challenge, Tech. Rep., 2022.

[12] T. Lamort de Gail and D. Kicinski, "Take it easy: Relaxing contrastive ranking loss with CIDEr," DCASE2022 Challenge, Tech. Rep., 2022.

[13] T. Pellegrini, "IRIT-UPS DCASE 2022 language-based audio retrieval system," DCASE2022 Challenge, Tech. Rep., 2022.

[14] Y. Lai, J. Pan, and B. Chen, "A ResNet-Based clip text-to-audio retrieval system for DCASE challenge 2022 task," DCASE2022 Challenge, Tech. Rep., 2022.

[15] Y. Wu, T. Zhang, and K. Chen, "Text-to-audio retrieval via large-scale contrastive training," DCASE2022 Challenge, Tech. Rep., 2022.

[16] B. Weck, M. Pérez Fernández, H. Kirchhoff, and X. Serra, "Aligning audio and text embeddings for the language-based audio retrieval task of the DCASE challenge 2022," DCASE2022 Challenge, Tech. Rep., 2022.

[17] F. Xiao, J. Guan, H. Lan, Q. Zhu, and W. Wang, "Language-based audio retrieval with pretrained CNN and graph attention," DCASE2022 Challenge, Tech. Rep., 2022.

[18] J. Park, C. Hwang, I.-Y. Kwak, and C. Lim, "CAU submission to DCASE 2022 task6b: Language-based audio retrieval using transfer learning," DCASE2022 Challenge, Tech. Rep., 2022.

[19] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah, "Signature verification using a "siamese" time delay neural network," in *Proc. Int. Conf. Neural Inf. Process. Syst. (NIPS)*, 1993, pp. 737–744.

[20] X. Xu, H. Dinkel, M. Wu, and K. Yu, "A crnn-gru based reinforcement learning approach to audio captioning," in *Proc. Detect. Classif. Acoust. Scenes Events Work. (DCASE)*, 2019, pp. 225–229.

[21] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," in *Proc. Int. Conf. Learn. Representations (ICLR)*, 2013.

[22] "Word2Vec," https://code.google.com/archive/p/word2vec, accessed: 2021-09-27.

[23] D. Harwath, A. Torralba, and J. R. Glass, "Unsupervised learning of spoken language with visual context," in *Proc. Int. Conf. Neural Inf. Process. Syst. (NIPS)*, 2016, pp. 1866–1874.

[24] A. Mesaros, A. Diment, B. Elizalde, T. Heittola, E. Vincent, B. Raj, and T. Virtanen, "Sound event detection in the DCASE 2017 challenge," *IEEE/ACM Trans. Audio Speech Lang. Process.*, pp. 992–1006, 2019.

[25] Q. Kong, Y. Cao, T. Iqbal, Y. Wang, W. Wang, and M. D. Plumbley, "PANNs: Large-scale pretrained audio neural networks for audio pattern recognition," *IEEE/ACM Trans. Audio Speech Lang. Process.*, pp. 2880–2894, 2020.

[26] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. Conf. North Am. Chapter Assoc. Comput. Linguist. Hum. Lang. Technol. (NAACL-HLT)*, 2019, pp. 4171–4186.

[27] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "RoBERTa: A robustly optimized BERT pretraining approach," 2019. [Online]. Available: https://arxiv.org/abs/1907.11692

[28] A. van den Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," 2018. [Online]. Available: https://arxiv.org/abs/1807.03748

[29] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2020.

[30] R. Arandjelovic and A. Zisserman, "Look, listen and learn," in *Proc. Int. Conf. Comput. Vis. (ICCV)*, 2017.

[31] H. Chen, W. Xie, A. Vedaldi, and A. Zisserman, "VGGsound: A large-scale audio-visual dataset," in *Proc. IEEE Int. Conf. Acoustic., Speech and Signal Process. (ICASSP)*, 2020, pp. 721–725.

[32] S. Hershey, S. Chaudhuri, D. P. W. Ellis, J. F. Gemmeke, A. Jansen, R. C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold, M. Slaney, R. J. Weiss, and K. Wilson, "CNN architectures for large-scale audio classification," in *Proc. IEEE Int. Conf. Acoustic., Speech and Signal Process. (ICASSP)*, 2017, pp. 131–135.

[33] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, "BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," in *Proc. Annu. Meet. Assoc. Comput. Linguist. (ACL)*, 2020, pp. 7871–7880.

[34] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence embeddings using Siamese BERT-networks," in *Proc. Empherical Methods Nat. Lang. Process. (EMNLP)*, 2019, pp. 3982–3992.

[35] K. Koutini, J. Schlüter, H. Eghbal-zadeh, and G. Widmer, "Efficient training of audio transformers with patchout," 2021. [Online]. Available: https://arxiv.org/abs/2110.05069

[36] A. Guzhov, F. Raue, J. Hees, and A. Dengel, "ESResNet: Environmental sound classification based on visual domain models," in *Proc. Int. Conf. Pattern Recognit. (ICPR)*, 2021, pp. 4933–4940.

[37] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, "Learning transferable visual models from natural language supervision," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2021, pp. 8748–8763.

[38] K. Chen, X. Du, B. Zhu, Z. Ma, T. Berg-Kirkpatrick, and S. Dubnov, "HTS-AT: A hierarchical token-semantic audio transformer for sound classification and detection," in *Proc. IEEE Int. Conf. Acoustic., Speech and Signal Process. (ICASSP)*, 2022, pp. 646–650.

[39] P. Khosla, P. Teterwak, C. Wang, A. Sarna, Y. Tian, P. Isola, A. Maschinot, C. Liu, and D. Krishnan, "Supervised contrastive learning," in *Proc. Int. Conf. Neural Inf. Process. Syst. (NIPS)*, 2020, pp. 18 661–18 673.

[40] K. Sohn, "Improved deep metric learning with multi-class n-pair loss objective," in *Proc. Int. Conf. Neural Inf. Process. Syst. (NIPS)*, 2016, pp. 1857–1865.

[41] C. D. Kim, B. Kim, H. Lee, and G. Kim, "AudioCaps: Generating captions for audios in the wild," in *Proc. Conf. North Am. Chapter Assoc. Comput. Linguist. Hum. Lang. Technol. (HLT-NAACL)*, 2019, pp. 119–132.

A MIXED SUPERVISED LEARNING FRAMEWORK FOR TARGET SOUND DETECTION

Dongchao Yang[1], *Helin Wang*[1], *Wenwu Wang*[2], *Yuexian Zou*[1]*

[1]ADSPLAB, School of ECE, Peking University, Shenzhen, China
[2]Center for Vision, Speech and Signal Processing, University of Surrey, UK

ABSTRACT

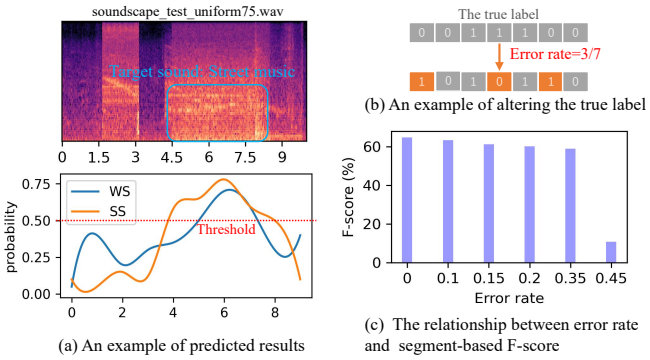
Target sound detection (TSD) aims to detect the target sound from mixture audio given the reference information. Previous works have shown that TSD models can be trained on fully-annotated (frame-level label) or weakly-annotated (clip-level label) data. However, there are some clear evidences show that the performance of the model trained on weakly-annotated data is worse than that trained on fully-annotated data. To fill this gap, we provide a mixed supervision perspective, in which learning novel categories (target domain) using weak annotations with the help of full annotations of existing base categories (source domain). To realize this, a mixed supervised learning framework is proposed, which contains two mutually-helping student models (*f_student* and *w_student*) that learn from fully-annotated and weakly-annotated data, respectively. The motivation is that *f_student* learned from fully-annotated data has a better ability to capture detailed information than *w_student*. Thus, we first let *f_student* guide *w_student* to learn the ability to capture details, so *w_student* can perform better in the target domain. Then we let *w_student* guide *f_student* to fine-tune on the target domain. The process can be repeated several times so that the two students perform very well in the target domain. To evaluate our method, we built three TSD datasets based on UrbanSound and Audioset. Experimental results show that our methods offer about 8% improvement in event-based F-score as compared with a recent baseline.

Index Terms— Target sound detection, audioset, weakly supervised, mixed supervised learning

1. INTRODUCTION

In target sound detection (TSD) [1], one aims to recognize and localize the target sound source within a mixture audio given a reference audio, e.g. detecting the dog bark sound within a street. TSD can be applied to numerous potential fields [2–4], such as species migration monitoring and large-scale multimedia indexing. Sound event detection (SED) [5] is a similar task with TSD, and a lot of works have been done for SED [6–11]. However, SED aims to classify and localize all pre-defined events (*e.g.*, dog barking, man speaking) within an audio clip, which significantly limits the flexibility to detect unseen classes. Different with SED, TSD only focuses on detecting the event that we interest. TSD does not require pre-defined set of categories, therefore, it can be easily extended to sound detection from an open set. In a recent work, a target sound detection network (TSDNet) [1] is presented, where a conditional network is used to generate sound-discriminative embedding which is then used as the reference information to guide a detection network for the detection of the target sound from the mixture audio. TSDNet provides a good detection performance when training data is fully-annotated, e.g. the onset and offset time of the target sound are provided in the annotations. However, collecting large-scale fully-annotated data is time-consuming and labor-intensive. Weakly supervised TSD is an effective method to reduce the reliance on fully-annotated data, but the performance tends to degrade significantly [1].

In this paper, we consider TSD with mixed supervision, which learns novel sound categories (target domain) using weak annotations with the help of full annotations of the existing base sound categories (source domain). Under this setting, we can use a small-scale fully-annotated dataset (e.g. URBAN-SED [12]) to complement a large-scale weakly-annotated dataset (e.g. Audioset [13]). To achieve this, we propose a novel mixed supervised learning framework, which includes two mutually-helping student models (*f_student* and *w_student*), which are trained by fully- and weakly-annotated data, respectively. The proposed method involves three novel aspects. Firstly, the *f_student* learned from fully-annotated data has better ability in capturing detailed information than the *w_student*. As Figure 1 (a) shows, the model trained on weakly-annotated data fails to locate the event boundary: it only focuses on the most distinct part and misses the boundary information. Thus, we propose a frame-level knowledge distillation (KD) strategy to transfer the knowledge from *f_student* to *w_student*, which makes *w_student* able to capture more details, *e.g.* boundary information.



(a) An example of predicted results

(b) An example of altering the true label

(c) The relationship between error rate and segment-based F-score

Figure 1: (a) shows the predicted results generated by weakly (WS) and strongly supervised (SS) learning. (b) shows an example of altering the true label to false label. (c) shows the influence of the error rate of the frame-level label when we train TSDNet model [1] on the URBAN-TSD dataset.

* Corresponding Author: zouyx@pku.edu.cn

This paper was partially supported by Shenzhen Science & Technology Research Program (No: GXWD20201231165807007-20200814115301001; No: JSGG20191129105421211) and NSFC (No: 62176008).

However, it is hard to transfer all of the knowledge from *f_student* to *w_student*. Thus, we propose to directly apply *f_student* to the target domain with the guidance of *w_student*. Specifically, *w_student* is used to produce frame-level pseudo labels for *f_student*. This strategy inspired by an interesting phenomenon that even if there are some errors in the frame-level labels, the detection performance remains stable or decreases only slightly (when the error rate lower than 0.35), as Figure 1 (c) shows. The process of mutually helping can be repeated several times so that the two students perform very well in the target domain. Lastly, we found that the mismatch between source and target data distribution tends to affect significantly the performance of transfer learning, *e.g.* URBAN-SED [12] and Audioset [14]. Thus, we propose an adversarial training strategy to solve the domain mismatch problem. To evaluate our method, we built two small-scale fully-annotated datasets and a large-scale weakly-annotated dataset based on URBAN-SED and Audioset. Experimental results show that two small-scale fully annotated datasets could significantly improve the performance on large-scale weakly-annotated dataset.

2. RELATED WORK

Many methods [15–22] have been proposed to utilize both fully- and weakly-annotated data to train the SED model. However, previous methods assume that fully- and weakly- annotated data belong to the same set of pre-defined categories. Our method aims to use the existing base categories with full annotations to facilitate the recognition of novel categories with weak labels. This setting is more realistic for the reason that a small-scale fully-annotated dataset with few categories is, in practice, easier to create, as compared with a large-scale fully-annotated dataset of many categories.

3. TSD DATASETS

We built two TSD datasets based on Audioset [14], which includes 94126 training clips and 16118 test clips, from 456 different classes. One is the large-scaled TSD dataset, named as L-TSD dataset, by choosing 192 different classes from Audioset. The other is the small-scaled fully-annotated TSD dataset (S-TSD), by choosing 51 different classes from Audioset. L-TSD includes two types of annotated samples, that is, fully-annotated (*i.e.* L-TSD-strong) and weakly-annotated (*i.e.* L-TSD-weak) samples. There is no common class between L-TSD and S-TSD datasets. The process of building the datasets is similar to the one described in [1]. The mixture audio signals come from Audioset. If there are N sound events in the mixture audio, we can generate N positive samples. We also generate $N/2$ negative samples, which do not contain the target sound. To prepare reference audio, we select the audio clips for each category directly from the Audioset training set i.e. those that do not contain interference from other events. In total, L-TSD includes 490,336 training and 83,334 test clips, while S-TSD contains 26,247 training and 5113 test clips. In addition, we choose the URBAN-TSD dataset [1] as another small-scale fully-annotated dataset.

4. PROPOSED METHOD

Figure 2 shows our proposed mixed supervised learning framework. The core idea of the framework is that the two students can teach each other iteratively. One of the student models is trained on fully-annotated data, we name it as *f_student*. The other model is trained on weakly-annotated data, and we name it as *w_student*. The two students have the same structure, while the only difference is that *w_student* has a linear softmax pooling layer [10].

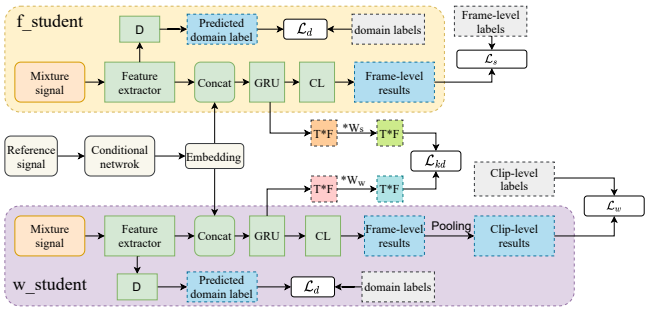


Figure 2: The architecture of the mixed supervised learning framework. CL denotes classification layers, which includes two fully-connected layers and one softmax function. D denotes the discriminator, which consists of three convolutional layers and one fully-connected layer.

4.1. Network Structure

Conditional network. The conditional network aims to extract a sound-discriminative embedding vector from the reference audio. Similar to the previous work [1], we adopt a VGG-like convolutional neural network (CNN) model [23] for the conditional network.

Detection network. Similar to the previous work [1], the network is composed of 5 convolutional layers, 1 Bi-GRU layer, and 2 fully-connected layers. Given the mel-spectrogram of the mixture audio $\boldsymbol{x} \in \mathcal{R}^{T \times F}$, where T and F denote the number of frames and the dimension of frame. The detection network aims to predict frame-level probabilities

$$\hat{p}_i = \mathbb{P}(Y = k | X = x_i, \boldsymbol{e}; \phi) \tag{1}$$

where ϕ denotes the trainable parameters of the detection network, $\boldsymbol{e}$ denotes the embedding obtained from the conditional network and x_i denotes the i-th frame of the mixture audio $\boldsymbol{x}$. For *f_student*, given the ground-truth label $p_i \in \{0, 1\}$ for each frame, which can be optimized by minimizing the binary cross entropy (BCE) loss:

$$\mathcal{L}_s = \sum_{i=1}^{t} (-p_i \log \hat{p}_i - (1 - p_i) \log(1 - \hat{p}_i)) \tag{2}$$

where t indicates the number of frames. The difference between *f_student* and *w_student* is that the latter needs a pooling layer to get the clip-level prediction. Thus, a LinSoft pooling layer [10] is added after the last layer of the *f_student*. *w_student* aims to predict a clip-level probability $\hat{P} = f_{LSP}(\hat{p}_1, \hat{p}_2, ..., \hat{p}_t)$ where $f_{LSP}(\cdot)$ denotes the LinSoft pooling function. Given the clip-level ground-truth label $P \in \{0, 1\}$, the BCE loss is applied as the loss function:

$$\mathcal{L}_w = -P \log \hat{P} - (1 - P) \log(1 - \hat{P}) \tag{3}$$

4.2. Two-student Learning

In this part, we introduce the details of how to enable *w_student* and *f_student* to help each other.

Frame-level knowledge distillation. According to formula (3), we

can see that *w_student* makes a decision on the whole audio clip. Compared to *f_student*, *w_student* is limited in capturing the detailed information of the sound events. To address this issue, we propose to first train *f_student* on a small-scale fully-annotated dataset (*i.e.* source domain), and then transfer its knowledge to *w_student*, so that *w_student* can get better performance on weakly-annotated dataset (*i.e.* target domain). Specifically, we first train the *f_student* model on the source dataset with strong labels. After that, we train the *w_student* model on the source data with weak labels. We then treat the trained *f_student* model as a teacher, to generate a frame-level feature representation. As a result, *w_student* may capture more detailed information, due to the frame-level class-agnostic knowledge distillation. More specifically, we can train *w_student* with the following objective function,

$$\mathcal{L}_{w\_kd} = \mathcal{L}_w + \mathcal{L}_{kd} \tag{4}$$

$$\mathcal{L}_{kd} = ||\boldsymbol{F}_s \cdot \boldsymbol{W}_s - \boldsymbol{F}_w \cdot \boldsymbol{W}_w||_2 \tag{5}$$

where $\boldsymbol{F}_s$ and $\boldsymbol{F}_w$ denote the feature map of the GRU layer of the two models, and $\boldsymbol{W}_s$ and $\boldsymbol{W}_w$ denote the transformation matrix.

Pseudo Supervised Training. The idea of pseudo supervised training strategy is motivated by an interesting observation, i.e. even if there are some errors in the frame-level labels, the detection performance remains stable or decreases only slightly. This means that we could use the noisy frame-level labels as our training target. In this paper, we propose to use *w_student* to produce noisy frame-level labels (i.e. pseudo labels), and then use the pseudo labels to re-train *f_student*, as follows

$$\hat{p}_i^w = \mathbb{P}(Y = k|X = x_i, \boldsymbol{e}; \phi_w), \hat{p}_i^s = \mathbb{P}(Y = k|X = x_i, \boldsymbol{e}; \phi_s) \tag{6}$$

$$\mathcal{L}_{re\_s} = \sum_{i=1}^{t}(-\hat{p}_i^w \log \hat{p}_i^s - (1 - \hat{p}_i^w) \log(1 - \hat{p}_i^s)) \tag{7}$$

Adversarial Training. In our experiments, we found that the mismatch between source and target data distribution could significantly degrade the performance of the mixed supervised learning framework. For example, if we choose the URBAN-TSD as the source dataset, and the L-TSD-weak as the target dataset, the performance will decrease substantially. This is because there is domain mismatch between URBAN-TSD and L-TSD-weak datasets [24, 25], i.e. *f_student* and *w_student* are first trained on the URBAN-TSD dataset but tested on L-TSD-weak dataset. To solve the domain mismatch problem, we propose a domain adversarial training strategy that aims to learn a common subspace shared by both the source and target domains, which enables all domains to have the same data distribution in the feature space. Specifically, inspired by GAN [26] and DANN [27], we make use of the adversarial relationship between modules Feature extractor (F) and Discriminator (D) to learn domain-invariant features in the feature space. To achieve this, we add an adversarial loss when we train *f_student* and *w_student*, as follows

$$\mathcal{L}_d = ||D(\boldsymbol{z}) - \boldsymbol{d}||_2^2 \tag{8}$$

$$\mathcal{L}_{d\_tsd} = \mathcal{L}_{tsd} - \lambda_d * \mathcal{L}_d \tag{9}$$

where $\boldsymbol{z}$ denotes the intermediate feature produced by F, and $\boldsymbol{d}$ denotes the domain label. The domain label is defined as $\boldsymbol{d} = [1, 0]^T$ (which stands for the source domain) or $\boldsymbol{d} = [0, 1]^T$ (target domain). $\mathcal{L}_d$ denotes the domain classification loss of the discriminator, $\mathcal{L}_{tsd}$ denotes the detection loss. $\mathcal{L}_{d\_tsd}$ denotes the training objective function, which minimizes the detection loss and meanwhile maximizes the domain classification loss. The parameter λ_d controls the trade-off between $\mathcal{L}_{tsd}$ and $\mathcal{L}_d$. In our experiments, λ_d is set to 0.2 empirically based on the validation set.

Iterative Training Strategy. According to the previous description, we can use frame-level KD to transfer the knowledge from *f_student* to *w_student*, and use the pseudo supervised training to transfer the knowledge from *w_student* to *f_student*. Intuitively, the process can be repeated several times. Thus, an iterative training strategy is proposed. The whole algorithm is summarized in Algorithm 1.

Algorithm 1 Two-Student Learning

Input:
The source dataset D_s and the target dataset D_t
Output: *f_student* and *w_student* model
1: Training *f_student* on D_s using formula (2) and (9)
2: Training *w_student* on D_s using formula (4) and (9)
3: Retraining *w_student* on D_t using formula (3) and (9)
4: Retraining *f_student* on D_t using formula (7) and (9)
5: While True:
Retraining *w_student* on D_t, using formula (4)
Retraining *f_student* on D_t using formula (7)
If no improvement: break;
6: **return** *f_student* and *w_student*;

5. EXPERIMENTS

5.1. Datasets

In this section, we introduce the source dataset and target dataset used in our experiments. The source (*resp.*, target) dataset is fully-annotated (*resp.*, weakly-annotated).

Source Dataset. We first use the S-TSD dataset as the source dataset, which is a small-scale fully-annotated dataset based on Audioset [14]. In addition, we choose 10-category URBAN-TSD [1] as another source dataset, which includes two similar categories as in the L-TSD dataset: *dog_bark* and *gun_shot*.

Target Dataset. We take the L-TSD-weak dataset as the target dataset. The details were given in Section 3.

5.2. Experimental Setups

Conditional Network. We use the pre-trained PANNs [23] model to initialize the conditional network, and then fix it in the training process.

Detection network. All the raw audios are down-sampled to 22.05kHz and then Short Time Fourier Transform (STFT) with a window size of 2048 samples are applied, followed by a Mel-scaled filter bank on perceptually weighted spectrogram. This results in 64 Mel frequency bins and around 50 frames per second. When training *f_student* and *w_student* for the first time, the Adam optimizer [28] is used for 100 epochs, with an initial learning rate of 1×10^{-3}. When they are re-trained, the learning rate is set as 1×10^{-4}.

Metrics. We use the segment-based F-score and event-based F-score [29] as the evaluation metrics, which are the most commonly used metrics for detection task.

Table 1: F-score comparison with different supervision strategy on L-TSD test set. SS, WS and MS represent strong, weak and mixed supervision, respectively. F-scores are macro-averaged.

Method	Source dataset	Segment-F score	Event-F score
SS [1]	-	58.57	50.4
WS [1]	-	49.39	39.07
MS (ours)	S-TSD	50.95	47.19
	URBAN-TSD	51.31	47.56

Table 2: Ablation studies on different strategies on L-TSD test set.

Model	KD	PS	AD	Segment-F score	Event-F score
w_student	✗	✗	✗	49.39	39.07
	✗	✗	✓	49.34	39.11
	✓	✗	✗	37.55	39.74
	✓	✗	✓	50.34	41.35
f_student	✗	✓	✓	48.09	43.47
	✓	✓	✗	37.67	45.42
	✓	✓	✓	**51.31**	**47.56**

5.3. Experiments on S-TSD and L-TSD Datasets

In this section, we use S-TSD as the source dataset and L-TSD-weak as the target dataset. We compare our method with strongly supervised (SS) and weakly supervised (WS) methods. For SS-TSD, we directly train *f_student* on the L-TSD-strong dataset with the strong labels. For WS-TSD, we directly train *w_student* on the L-TSD-weak dataset with the weak labels. Note that for mixed supervised (MS) method, we only report the results obtained by *f_student* for the reason that *f_student* performs better than *w_student*. The experimental results are given in Table 1. From this table, we can see that our proposed MS method performs significantly better than the WS method, and performs similarly to the SS method.

5.4. Experiments on URBAN-TSD and L-TSD Datasets

We also conduct experiments by using URBAN-TSD as the source dataset and L-TSD-weak as the target dataset. Table 1 shows the experimental results, and we can see that using URBAN-TSD as the source dataset significantly improves the performance compared with the WS method. Furthermore, by comparing rows 3 and 4, we can find that using URBAN-TSD as the source dataset obtains better performance than using S-TSD as the source dataset. One of the reasons is the categories of URBAN-TSD and L-TSD-weak datasets have overlaps.

5.5. Ablation Studies

By taking URBAN-TSD as the source dataset, we conduct ablation studies to investigate the effectiveness of knowledge distillation (KD), pseudo supervised training (PS) and adversarial (AD) training, with the results shown in Table 2. For the *w_student* model: (1) The first row shows the results for directly training *w_student* on L-TSD-weak (without using any strategy). (2) The second row shows the results of directly training *w_student* on L-TSD-weak while using the AD strategy. We can see that only using the AD strategy does not give improvements, because it only aims to align the data distribution. (3) By comparing rows 2 and 4, we can see that the KD strategy can improve the *w_student*'s ability in capturing detailed information. (4) By comparing rows 3 and 4, we can see the effectiveness of the AD strategy when there is mismatch between the source and target datasets.

Table 3: Ablation study on the effect of the number of iterations on iterative training strategy.

Iterations	Model	Segment-F score	Event-F score
1	w_student	50.39	40.88
	f_student	50.21	46.99
2	w_student	50.14	41.60
	f_student	51.10	46.96
3	w_student	51.33	44.70
	f_student	50.95	47.19

For the *f_student* model: (1) By comparing rows 5 and 7, we can see that the KD strategy can improve the performance of *f_student* for the reason that it can improve the performance of *w_student*, and the better *w_student* leads to better *f_student*. (2) By comparing rows 6 and 7, we can find the improvements given by the AD strategy. Lastly, we can see that *f_student* has better performance than *w_student*. A reason could be that the ability of *f_student* in capturing the detailed information is only partially transferred to *w_student* through the KD strategy.

Influence of the number of iterations. By taking S-TSD as the source dataset, we conduct ablation studies to investigate the influence of the number of iterations on the iterative training strategy, and the results are shown in Table 3. In this work, considering the cost of training time, we only tested three stages. For *f_student*, the segment- and event-based F-scores are increased from 50.21% and 46.99% to 50.95% and 47.19%, respectively. Furthermore, the event-based F-score of *w_student* is increased from 40.88% to 44.7%. It means that the two students can help each other.

Why does pseudo supervised training work? To explain why the pseudo supervised training strategy is effective, we calculate the error rate between the true label and pseudo label (produced by *w_student*) on the L-TSD-strong training set. We found that the average error rate is 27.03%, over the 490,336 audio clips. This is consistent with our empirical results as shown earlier in Figure 1: if the error rate is smaller than 0.35, the performance is similar to that of the case where the true label is taken as the target.

6. CONCLUSIONS

In this paper, we have presented a novel mixed supervised learning framework, which effectively improves the performance of novel categories with the help of a small-scale fully-annotated base categories dataset. In the future, we will apply our method to other tasks, such as SED and object detection. The source code and dataset of this work have been released[1].

7. REFERENCES

[1] D. Yang, H. Wang, Y. Zou, and C. Weng, "Detect what you want: Target sound detection," *arXiv preprint arXiv:2112.10153*, 2021.

[2] J. P. Bello, C. Silva, O. Nov, R. DuBois, A. Arora, J. Salamon, C. Mydlarz, and H. Doraiswamy, "Sonyc: A system for the

[1] https://github.com/yangdongchao/weakly-target-sound-detection

monitoring, analysis and mitigation of urban noise pollution," *arXiv preprint arXiv:1805.00889*, 2018.

[3] D. Stowell and D. Clayton, "Acoustic event detection for multiple overlapping similar sources," in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*. IEEE, 2015, pp. 1–5.

[4] S. Hershey, S. Chaudhuri, D. Ellis, J. Gemmeke, A. Jansen, R. C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold, *et al.*, "CNN architectures for large-scale audio classification," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 131–135.

[5] Q. Kong, Y. Xu, I. Sobieraj, W. Wang, and M. D. Plumbley, "Sound event detection and time-frequency segmentation from weakly labelled data," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 27, pp. 777–787, 2019.

[6] H. Dinkel, M. Wu, and K. Yu, "Towards duration robust weakly supervised sound event detection," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 887–900, 2021.

[7] L. Lin, X. Wang, H. Liu, and Y. Qian, "Specialized decision surface and disentangled feature for weakly-supervised polyphonic sound event detection," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 1466–1478, 2020.

[8] Q. Kong, Y. Xu, W. Wang, and M. D. Plumbley, "Sound event detection of weakly labelled data with cnn-transformer and automatic threshold optimization," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 2450–2460, 2020.

[9] A. Mesaros, T. Heittola, T. Virtanen, and M. D. Plumbley, "Sound event detection: A tutorial," *IEEE Signal Processing Magazine*, vol. 38, no. 5, pp. 67–83, 2021.

[10] Y. Wang, J. Li, and F. Metze, "A comparison of five multiple instance learning pooling functions for sound event detection with weak labeling," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 31–35.

[11] I. Martín-Morató, A. Mesaros, T. Heittola, T. Virtanen, M. Cobos, and F. Ferri, "Sound event envelope estimation in polyphonic mixtures," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 935–939.

[12] J. Salamon, D. MacConnell, M. Cartwright, P. Li, and J. P. Bello, "Scaper: A library for soundscape synthesis and augmentation," in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*. IEEE, 2017, pp. 344–348.

[13] J. Gemmeke, D. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio set: An ontology and human-labeled dataset for audio events," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 776–780.

[14] S. Hershey, D. P. Ellis, E. Fonseca, A. Jansen, C. Liu, R. C. Moore, and M. Plakal, "The benefit of temporally-strong labels in audio event classification," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 366–370.

[15] A. Kumar and B. Raj, "Audio event and scene recognition: A unified approach using strongly and weakly labeled data," in *International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2017, pp. 3475–3482.

[16] Y. Liang, Y. Long, Y. Li, and J. Liang, "Joint weakly supervised AT and AED using deep feature distillation and adaptive focal loss," *arXiv preprint arXiv:2103.12388*, 2021.

[17] L. Lin, X. Wang, H. Liu, and Y. Qian, "Guided learning for weakly-labeled semi-supervised sound event detection," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 626–630.

[18] Z. Shi, L. Liu, H. Lin, R. Liu, and A. Shi, "Hodgepodge: Sound event detection based on ensemble of semi-supervised learning methods," *arXiv preprint arXiv:1907.07398*, 2019.

[19] T. K. Chan, C. S. Chin, and Y. Li, "Non-negative matrix factorization-convolutional neural network (nmf-cnn) for sound event detection," *arXiv preprint arXiv:2001.07874*, 2020.

[20] X. Zheng, Y. Song, I. McLoughlin, L. Liu, and L.-R. Dai, "An improved mean teacher based method for large scale weakly labeled semi-supervised sound event detection," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 356–360.

[21] L. Cances and T. Pellegrini, "Comparison of deep co-training and mean-teacher approaches for semi-supervised audio tagging," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 361–365.

[22] Y. Guan, J. Xue, G. Zheng, and J. Han, "Sparse self-attention for semi-supervised sound event detection," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022, pp. 821–825.

[23] Q. Kong, Y. Cao, T. Iqbal, Y. Wang, W. Wang, and M. Plumbley, "Panns: Large-scale pretrained audio neural networks for audio pattern recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 2880–2894, 2020.

[24] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, 2009.

[25] S. Ben-David, J. Blitzer, K. Crammer, and F. Pereira, "Analysis of representations for domain adaptation," *Advances in Neural Information Processing Systems*, vol. 19, 2006.

[26] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *arXiv preprint arXiv:1406.2661*, 2014.

[27] R. Wang, M. Wang, X.-L. Zhang, and S. Rahardja, "Domain adaptation neural network for acoustic scene classification in mismatched conditions," in *Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*. IEEE, 2019, pp. 1501–1505.

[28] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *ICLR*, 2015.

[29] A. Mesaros, T. Heittola, and T. Virtanen, "Metrics for polyphonic sound event detection," *Applied Sciences*, vol. 6, no. 6, p. 162, 2016.