End-to-End and Self-Supervised Learning for ComParE 2022 Stuttering Sub-Challenge

Shakeel A Sheikh, Md Sahidullah, Fabrice Hirsch, Slim Ouni

End-to-End and Self-Supervised Learning for ComParE 2022 Stuttering Sub-Challenge

Shakeel A. Sheikh[1], Md Sahidullah[1], Fabrice Hirsch[2], Slim Ouni[1]
Universite de Lorraine, CNRS, Inria, LORIA, F-54000, Nancy, France[1]
Universite Paul-Valery Montpellier, CNRS, Praxiling, Montpellier, France[2]

ABSTRACT

In this paper, we present end-to-end and speech embedding based systems trained in a self-supervised fashion to participate in the ACM Multimedia 2022 ComParE Challenge, specifically the stuttering sub-challenge. In particular, we exploit the embeddings from the pre-trained Wav2Vec2.0 model for stuttering detection (SD) on the KSoF dataset. After embedding extraction, we benchmark with several methods for SD. Our proposed self-supervised based SD system achieves a UAR of 36.9% and 41.0% on validation and test sets respectively, which is 31.32% (validation set) and 1.49% (test set) higher than the best (DeepSpectrum) challenge baseline (CBL). Moreover, we show that concatenating layer embeddings with Mel-frequency cepstral coefficients (MFCCs) features further improves the UAR of 33.81% and 5.45% on validation and test sets respectively over the CBL. Finally, we demonstrate that the summing information across all the layers of Wav2Vec2.0 surpasses the CBL by a relative margin of 45.91% and 5.69% on validation and test sets respectively.

CCS CONCEPTS

• **Speech disorders → Stuttering**; *Disfluency, Stuttering Detection.*

KEYWORDS

Speech disorders, ComParE stuttering-sub challenge

ACM Reference Format:
Shakeel A. Sheikh[1], Md Sahidullah[1], Fabrice Hirsch[2], Slim Ouni[1]. 2022. End-to-End and Self-Supervised Learning for ComParE 2022 Stuttering Sub-Challenge. In *Proceedings of the 30th ACM International Conference on Multimedia October 10–14, 2022, Lisbon, Portugal ('MM 22').* ACM, New York, NY, USA, 5 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Stuttering is a speech impairment that impairs a person's ability to communicate and is a neuro-developmental speech disorder mostly characterized by involuntarily blocks/stop gaps, repetitions, prolongations, and interjections [7, 34]. These uncontrolled utterances are usually accompanied by psychological and linguistic variabilities [9, 13]. In addition, unique behaviors such as head shaking, lip tremors, fast eye blinks, and unusual lip shapes are commonly associated with these uncontrolled utterances [9]. These unique abnormal behaviors make it difficult for people who stutter (PWS) to communicate properly and, as a result, have a detrimental impact on their lives [3]. Speech therapy is frequently used by PWS to cope with their problem, where stuttering is identified using a variety of hearing and brain scan tests [11, 26, 31], however, such manual methods are extremely arduous, time-consuming, and expensive. Stuttering detection (SD) can also be adapted towards voice assistants such as Cortona, Alexa, etc. where the automatic speech recognition systems fail to recognize stuttered speech [30].

Most of the previous work in SD explored traditional classifiers mainly on hand-engineered features such as Mel-frequency cepstral coefficients (MFCCs), etc [26]. However, the trend has recently shifted towards the deep learning paradigm-based SD. Kourkounakis et al. [14] exploited residual network in conjunction with bi-directional long short-term memory (ResNet+BiLSTM) on a small subset of speakers (25) from the UCLASS dataset. Sheikh et al. [27] approached SD as a multi-class classification problem and introduced *StutterNet* which is based on a time delay neural network (TDNN) on a large set of speakers (100+) on the UCLASS dataset. Another study by Lea et al. [15] proposed a new SEP-28k stuttering dataset and used ConvLSTM on top of phoneme features for SD. A recent study carried out by Sheikh et al. [28] introduces adversarial learning to unlearn the podcast information from a speech utterance to learn robust stutter representations that are stutter discriminative, and at the same time are podcast invariant. Recently, Bayerl et al. [6] demonstrated the usefulness of leveraging features extracted from self-supervised pre-trained models that are trained on enormous datasets by utilizing the Wav2Vec2.0 model as a feature extractor. In a related study by Sheikh et al. [29], embeddings from the emphasized channel attention, propagation, and aggregation-time-delay neural network (ECAPA-TDNN) were also examined in addition to Wav2Vec2.0 embeddings in SD domain.

Our contribution to ComParE 2022 KSF-C stuttering sub-challenge explores end-to-end and self-supervised systems. For end-to-end systems, we use *StutterNet* [27] and ResNet+BiLSTM [14] with MFCC input features computed from KSoF dataset, and for addressing limited data issue, we explored self-supervised pre-trained speech embeddings extracted from various layers of Wav2vec2.0 model [4]. In addition, we demonstrate the impact and concatenation of Wav2Vec2.0 layer embeddings in SD.

2 OVERVIEW OF THE CHALLENGE

The ACM Multimedia 2022 Computational Paralinguistics ChallengE (ComParE) proposes four sub-competitions including *Vocalisations*, *Stuttering*, *Activity*, & *Mosquitoes* [24]. Among these, we only focus on the *Stuttering* (KSF-C) sub-challenge. In this sub-challenge, the task is to classify the speech segments into one of

'MM 22', October 10–14, 2022, Lisbon, Portugal

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/XXXXXXX.XXXXXXX

the eight categories including filler, garbage, prolongation, sound repetition, block, modified, word repetition, and no_disfluency. The details about the stuttering challenge, KSoF dataset, and its partitioning are available in [24].

3 SYSTEM DESCRIPTION

3.1 End-to-End Systems

In end-to-end speech classification systems, the model directly maps the input speech (e.g. raw speech or features such as MFCCs [10]) into its corresponding class. This involves a single-phase model training. In this work, we utilize two end-to-end models: *StutterNet* and ResNet+BiLSTM [14].

3.1.1 StutterNet. The *StutterNet* is based on a time delay neural network which has been proven effective in capturing temporal and contextual aspects of speech signal [20, 27, 33]. It is composed of five time delay layers with the initial layers capturing smaller contexts and deeper ones learning wider contexts in contrast to the standard neural networks which learn wider contexts at initial layers. The output activations from the last layer are fed to the statistical pooling layer to compute the mean and standard deviation across the temporal dimension. This is followed by three fully connected layers with each layer followed by a ReLU activation function. We apply batch normalization after each layer except the statistical pooling layer.

The KSoF dataset provided in this challenge is highly imbalanced [24]. To address the class imbalance, we used cost-level and architecture-level approaches. For cost-level, we penalize the majority class by modifying the standard cross entropy as:

$$\mathbf{L}_{\text{WCE}} = \frac{1}{\mathbf{B}} \sum_{b=1}^{\mathbf{B}} \frac{\sum_{i}^{N} \alpha_i * \log(p_i)}{\sum_{i,\, i \in \mathbf{B}}^{N} \alpha_i} \tag{1}$$

where $\mathbf{B}$ is total batches, N is the number of stuttered speech samples in a batch b_i, $\alpha_i = \frac{\mathcal{N}}{C * \mathcal{N}_i}$ ($\mathcal{N}$ is the number of training samples, C is number of classes, $\mathcal{N}_i$ is the number of training samples for class i), $p_i = \left(\frac{e^{c_i}}{\sum_{j=1}^{C} e^{c_j}} \right)$ is the predicted probability of class c_i of sample i.

For architecture-level, we use multi-branch training scheme similar to the work from Lea et al. [15], Sheikh et al. [28, 29]. This comprises a base encoder $\mathbf{E}$ (θ_e) followed by two parallel branches referred as *DisfluentBranch* $\mathcal{D}$ (θ_d) and *FluentBranch* $\mathcal{F}$ (θ_f). The embeddings generated by $\mathbf{E}$ (θ_e) are simultaneously passed to both the *FluentBranch* and *DisfluentBranch*, where the *FluentBranch* is trained to distinguish between fluent and disfluent samples, and the *DisfluentBranch* is trained to differentiate and classify the disfluent[1] sub categories with an overall objective function to optimize as:

$$\mathcal{L}_{\text{tot.}}(\theta_e, \theta_f, \theta_d) = \mathcal{L}_f(\theta_e, \theta_f) + \mathcal{L}_d(\theta_e, \theta_d) \tag{2}$$

3.1.2 ResNet+BiLSTM. The ResNet+BiLSTM based SD, proposed by Kourkounakis et al. [14] comprises residual unit with 18 convolution blocks to capture stutter-specific features. These features are then provided to two recurrent layers, with each layer having 512 BiLSTM units. Moreover, we also employed a multi-branched version of ResNet+BiLSTM in a similar fashion applied to *StutterNet*.

[1]If the prediction of a sample in $\mathcal{F}$ is not fluent, then the predictions of $\mathcal{D}$ are taken into consideration to reveal the disfluent class category

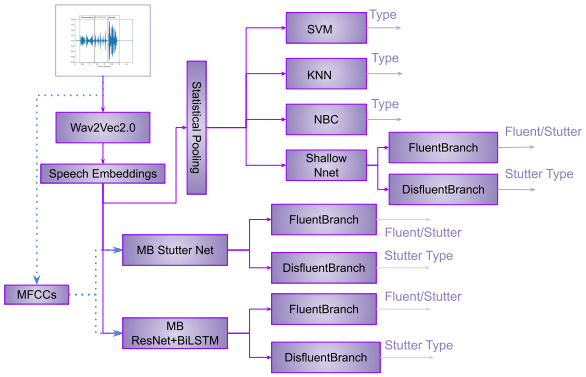

Figure 1: Block Diagram of the Proposed Pipeline for SD.

3.2 Self-Supervised Framework

Due to the availability of small-sized stuttering datasets, the deep learning paradigm hasn't shown much improvement in SD in comparison to other speech domains such as automatic speech recognition [4], speaker verification [5], emotion detection [1, 22], speaker diarizarion [18], etc. The small datasets are limited in capturing accents, linguistic content, age group, different speaking styles, etc. Moreover, the speech pathology datasets are very expensive to collect, as a result, this prevents the adoption of advanced deep learning models for SD. To overcome this bottleneck, we employ self-supervised learning (SSL), where we extract the features from the model pre-trained on a huge dataset for downstream SD tasks. SSL makes use of the data's underlying structure. In SSL classification systems, the model is first pre-trained on some pre-auxiliary task to capture rich embeddings from the innate structure of the data [4, 8, 16, 23]. These embeddings are then used for other downstream classification tasks. In this paper, we use speech representations from various layers of the pre-trained Wav2Vec2.0 model for the downstream SD task.

3.2.1 Wav2Vec2.0. The Wav2Vec2.0 model is a three-module-based self-supervised representation learning framework for raw audio, pre-trained on LibriSpeech dataset followed by fine-tuning towards automatic speech recognition task using connectionist temporal classification loss function. The three modules consist of feature encoder $\mathbb{F}$, contextual block $\mathbb{C}$ and quantization block $\mathbb{Q}$. The $\mathbb{F}$ encodes the raw input signal $\mathcal{X}$ into local features. These encoded features are then passed to $\mathbb{C}$ and $\mathbb{Q}$ modules to learn contextual speech representations. Several pre-trained models of contextual embedding dimensions of 768 (base) and 1024 (large) have been released. The model is trained in a self-supervised manner to learn the speech representations of an utterance by optimizing contrastive loss function using equation (3) as below:

$$\mathcal{L}_c = -\log \frac{\exp(sim(c_t, q_t)/\tau)}{\sum_{\tilde{q} \in Q_t} \exp(sim(c_t, \tilde{q})/\tau)} \tag{3}$$

where $sim(c_t, q_t) = c_t^T q_t / ||c_t|| \; ||q_t||$ is the cosine similarity between the contextualized transformer vector c_t and quantized vector q_t. Equation (3) is then augmented by a diversity loss. For details, please refer to the paper by Baevski et al. [4].

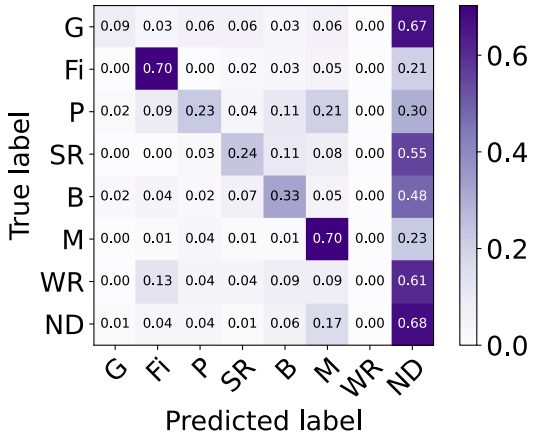

Figure 2: Confusion Matrix of SVM on Val. Set with L5 Embeddings[2]

In this paper, we use only the base pre-trained model of 768 embedding dimensions (trained on 960 hours of data) and we extract embeddings from the $\mathbb{F}$ block and from 12 layers of $\mathbb{C}$ block for SD. Moreover, we use statistical pooling over the temporal domain and concatenated the mean and standard deviation, resulting in a 768×2-dimensional feature vector before passing it to the downstream classifiers except multi-branched (MB) *StutterNet*, where we pass the speech embeddings directly.

4 RESULTS AND DISCUSSION

4.1 Training Details

We train the MB *StutterNet* and a shallow neural network (NNet) with Adam optimizer on a cross entropy loss function ($L = Lf + L_d$, L_f:*FluentBranch loss*, L_d:*DisfluentBranch loss* similar to equation (2)) using a learning rate of 1e-2, with a batch size of 128 over 50 epochs. The shallow NNet is composed of two branches with *FluentBranch* and *DisfluentBranch* with each branch having three fully connected (FC) layers. Each FC layer is followed by a ReLU activation function [2] and a 1D-batch normalization [12]. A dropout [32] of 0.3 is applied to the first two FC layers. A patience of seven is applied on a validation loss to stop the training. We train the MB *StutterNet* with two input features including $20 \times T$ MFCCs and $768 \times T$ speech embeddings extracted from Wav2Vec2.0. For support vector machines (SVM) [17], we experiment with different kernels including linear, polynomial, Radial basis function, and sigmoid, however, in this paper, we report the results only with the linear kernel. For K-nearest neighbor (KNN) [17], we choose the value of K = 5 empirically by grid search using the elbow method with $p = 2$ (Euclidean) distance metric. A statistical pooling (mean and standard deviation) is applied to Wav2Vec2.0 $768 \times T$ speech embeddings before passing them to KNN, naive Bayes classifier (NBC) [17], SVM, and NNet downstream classifiers. We use PyTorch [19] and Scikit-learn [21] for implementation purposes.

4.2 Evaluation

Table 1 summarizes the results of our experiments on KSoF dataset provided in this ComParE challenge [24]. We first train the end-to-end single branched version of *StutterNet* and ResNet+BiLSTM [14]

[2]Please refer to Table. 1 for class names.

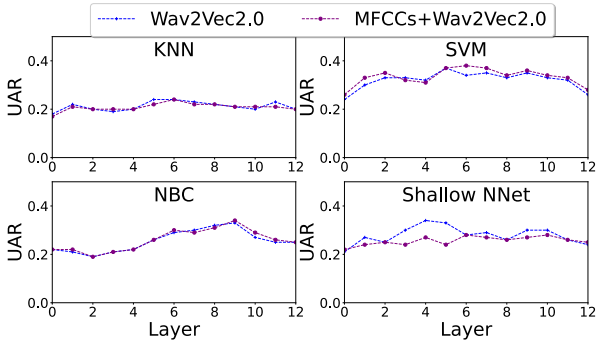

Figure 3: Comparison of Various Wav2Vec2.0 Contextual Layers and its Concatenation with MFCCs in SD with KNN, NBC, SVM, and Shallow NNet Classifiers where Y-axis Represents Unweighted Average Recall, and X-axis Represents Wav2Vec2.0 Layer Embeddings.

models on MFCC input features using a standard cross entropy loss function. Due to the class imbalance nature in KSoF dataset provided in this ComParE challenge, the models are skewed towards majority class recognition including *blocks*, *modified*, and *no_disfluencies* as can be confirmed by the results Table 1. We then applied cost-based approach by modifying the standard cross entropy using equation (1) with an aim to focus more on the minority classes. We found a huge improvement in garbage and word *repetitions category* using ResNet+BiLSTM, and, we found *prolongations*, *fillers*, and *sound repetitions* are the most difficult to recognize for ResNet+BiLSTM based model. For WCE based *StutterNet*, we found improvement in *garbage, fillers, prolongations and word repetitions*. Similar to ResNet+BiLSTM, we found that the *sound repetitions* are still the most challenging to identify. Further, we applied a multi-branched training scheme to our baselines, and, we found that except for *word repetition and no_disfluencies* classes, the MB *StutterNet* shows respectively a huge improvement of 600%, 275%, 280%, 4.63%, and 9.85% in *garbage, fillers, prolongation, sound repetition, blocks, and modified* over the baseline.

In the SSL setup, we experiment with different speech embeddings extracted from various layers of a Wav2Vec2.0 pre-trained model. The extracted speech embeddings improve the detection performance of almost all the classes over the baseline. Similar to Schuller et al. [24], we also found that the *word repetitions* are the most challenging ones to recognize. Among the various layers, layer five (L5) shows the best performance on UAR with all the downstream classifiers except for NBC, where, speech embeddings from layer nine (L9) perform better. Figure 3. shows the impact of various speech layer embeddings in SD with downstream classifiers. As presented by the plots, the first initial and last layers show lower UAR in comparison to the middle layers. We hypothesize that the lower layer speech representations contain information only from smaller contexts, and passing this information after statistical pooling to SVM, KNN, NBC, and NNet further inhibits it in learning stutter-specific patterns. Furthermore, Shah et al. [25] have also shown that the middle layers are good at capturing fluency and pronunciation features which are extremely important for stuttering detection, while initial layers of Wav2Vec2.0 exhibit more deviation in learning fluency (e.g., speech rate, pauses, etc.) and pronunciation (e.g., vowels, consonants, syllables, stress, etc.) features. The trend

Table 1: SD Results in Accuracy and UAR (G: Garbage, Fi: Fillers, P; Prolongation, SR: SoundRepetition, B: Block, M: Modified, WR: Word Repetition, ND: no_disfluencies, TA; Total Accuracy, BL: Baseline, CE: Cross Entropy, WCE: Weighted Cross Entropy, UAR: Unweighted Average Recall, L_i: Embeddings from Layer i, ":" is Concatenation, $\sum_i L_i$: Summing Information Over all Embedding Layers).

Model	Feature	G	Fi	P	SR	B	M	WR	ND	TA	UAR%(Val)	UAR%(Test)
ResNetBiLSTM+CE (BL)	MFCCs	0.00	0.00	0.00	7.89	38.46	60.54	0.00	58.33	42.16	20.7	NA
ResNetBiLSTM+WCE	MFCCs	42.42	0.00	60.38	0.00	01.92	18.38	8.70	01.13	09.06	16.6	NA
MB ResNetBiLSTM	L5	0.00	0.00	35.85	7.89	64.42	65.95	0.00	16.67	29.02	23.8	NA
SB *StutterNet* + CE (BL)	MFCCs	03.03	03.92	09.43	0.00	41.35	71.35	0.00	52.70	42.67	23.0	NA
SB *StutterNet* + WCE	MFCCs	51.52	20.59	26.42	10.53	19.23	43.24	17.39	03.60	17.92	24.0	NA
MB *StutterNet*	MFCCs	21.21	14.71	35.85	18.42	43.27	78.38	0.00	20.27	33.40	29.0	NA
SVM (Linear)	L5	9.09	69.61	22.64	23.68	32.69	70.27	0.00	67.57	56.93	36.9	41.0
SVM (Linear)	MFCCs:L5	21.21	59.80	26.42	26.32	39.42	71.35	0.00	56.08	52.34	37.6	42.6
NBC	L9	45.46	41.18	30.19	26.32	39.42	70.81	0.00	08.56	29.84	32.7	NA[3]
NNet	L5	6.06	59.80	26.42	10.53	0.00	75.14	4.35	**81.08**	**59.17**	32.9	NA
MB *StutterNet*	L5	15.15	71.57	32.08	15.79	39.42	76.22	0.00	57.43	54.79	38.5	40.3
MB *StutterNet*	L5:L9	03.03	81.37	28.30	**28.95**	50.96	74.60	4.35	54.50	55.40	40.8	42.6
MB *StutterNet*	$\sum_i L_i$	12.12	**78.43**	**39.62**	26.32	58.65	**80.00**	0.00	32.88	47.86	**41.0**	**42.7**

also shows that the UAR curve decreases for the last layers. This is most likely due to the reason that the Wav2Vec2.0 model is fine-tuned and adapted towards the ASR task, and it is quite possible that the information such as prosody, stress, and emotion state gets diluted, as also can be confirmed from the study by Shah et al. [25]. We also observe that the improvement gap between MFCC-based systems and Wav2Vec2.0 embedding-based systems is lower than the study carried out by Sheikh et al. [29]. The most obvious reason seems the linguistic information between the two. Sheikh et al. [29] used the SEP-28k corpus for the downstream SD task, which is also an English language stuttering dataset that coincides with the language of the LibriSpeech on which the Wav2Vec2.0 model was trained. Contrary to the SEP-28k dataset, there is a possibility of the loss of linguistic information while extracting embeddings of the KSoF German dataset (provided in this ComParE challenge) from the English-based Wav2Vec2.0 pre-trained model.

In addition, we experiment by concatenating the MFCC features to each speech embeddings layer of Wav2Vec2.0 after applying statistical pooling to both the features and using the downstream classifiers KNN, SVM, NBC, and Shallow NNet for SD, the plots of which are shown in Figure 3. We observe from the curves that the concatenation of MFCCs with layer embeddings only helps for the SVM classifier while for KNN and NBC, it remains almost unchanged. For shallow NNet, the UAR degrades while concatenating MFCC and speech embeddings. Over the CBL baseline, the SVM with speech embeddings and concatenated (MFCCs+speech embeddings) results in a relative improvement of 31.32%, and 33.81% on the validation set, and 1.49% and 5.45% on the test set respectively. We make use of the fact that each layer of the Wav2Vec2.0 model has different information and passing the summation over all layers ($\sum_i L_i$) to MB *StutterNet* results in a relative improvement of 45.91% on the validation set and 5.69% on the test set in UAR over the baseline (CBL) as shown in Table 2.

After analyzing various confusion matrices from the SD systems mentioned in Table 1, we found the trend that the most of the disfluent classes are still being falsely classified as ND (*no_disfluencies* class), and confusion in *garbage* is maximum followed by *word repetitions*, *sound repetitions and blocks*. This makes intuitive sense because the *repetitions* contain certain words or phrasal parts that, when examined individually, reveal themselves to be fluent utterances. Consider the phrase, "*for for* the movement". The word *for* is repeated twice, however, it is a fluent component if two *fors* are examined separately. For *garbage*, it is hard to understand the reason since it is either unintelligible or contains no speech. Furthermore, we observe that switching from inputting speech embeddings from a single layer to summing information across all layers ($\sum_i L_i$) reduces the confusion rate of various classes for MB *StutterNet*.

[4]We are only allowed to submit five UAR results on the test set due to the restriction in this ComParE challenge.

Table 2: UARs of the challenge baselines and our systems. (Val: Validation Set, Test: Test Set, CBL: Challenge Baseline)

Approach	Val(%)	Test(%)
ComParE [24]	30.2	37.6
auDeep [24]	17.7	25.9
BoAWs [24]	26.7	32.1
DeepSpectrum (CBL) [24]	28.1	40.4
Fusion [24]	28.7	38.3
(Ours) (SVM+L5)	36.9	41.0
(**Ours**) SVM+MFCCs:L5	37.6	42.6
(**Ours**) *MB StutterNet* + $\sum_i L_i$	**41.0**	**42.7**

5 CONCLUSION

This work presents our contribution to the ComParE competition by addressing the stuttering sub-challenge using end-to-end and pre-trained self-supervised models. In comparison to the best baseline (CBL), the Wav2Vec2.0 speech embeddings based SD system outperforms by a relative margin of 5.69% in UAR on the test set. The KSoF stuttering dataset provided in this challenge is highly imbalanced similar to the SEP-28k dataset, and in addition, the meta-data information such as age, transcription, and speaker information is also absent that could have been exploited to make the SD systems more robust.

ACKNOWLEDGMENTS

This work was made with the support of the French National Research Agency, in the framework of the project ANR BENEPHIDIRE (18-CE36- 0008-03)

REFERENCES

[1] Mehmet Berkehan Akçay and Kaya Oğuz. 2020. Speech emotion recognition: Emotional models, databases, features, preprocessing methods, supporting modalities, and classifiers. *Speech Communication* 116 (2020), 56–76.

[2] Andrea Apicella, Francesco Donnarumma, Francesco Isgrò, and Roberto Prevete. 2021. A survey on modern trainable activation functions. *Neural Networks* 138 (2021), 14–32.

[3] National Stuttering Association. 2008. The Experience of People Who Stutter: A Survey by the National Stuttering Association. Retrieved June 15, 2022 from https://westutter.org/wp-content/uploads/2016/12/NSAsurveyMay09.pdf

[4] Alexei Baevski et al. 2020. wav2vec 2.0: A framework for self-supervised learning of speech representations. In *Advances in NIPS*, Vol. 33. Curran Associates, Inc., 12449–12460.

[5] Zhongxin Bai and Xiao-Lei Zhang. 2021. Speaker recognition based on deep learning: An overview. *Neural Networks* 140 (2021), 65–99. https://doi.org/10.1016/j.neunet.2021.03.004

[6] Sebastian P Bayerl, Dominik Wagner, Elmar Nöth, and Korbinian Riedhammer. 2022. Detecting Dysfluencies in Stuttering Therapy Using wav2vec 2.0. *arXiv preprint arXiv:2204.03417* (2022).

[7] Joseph R Duffy. 2019. *Motor Speech Disorders e-Book: Substrates, Differential Diagnosis, and Management.* Elsevier Health Sciences.

[8] Linus Ericsson, Henry Gouk, Chen Change Loy, and Timothy M Hospedales. 2022. Self-supervised representation learning: Introduction, advances, and challenges. *IEEE Signal Processing Magazine* 39, 3 (2022), 42–62.

[9] Barry Guitar. 2013. *Stuttering: An Integrated Approach to its Nature and Treatment.* Lippincott Williams & Wilkins.

[10] Xuedong Huang, Alex Acero, Hsiao-Wuen Hon, and Raj Reddy. 2001. *Spoken language processing: A Guide to Theory, Algorithm, and System Development.* Prentice hall PTR.

[11] R. J. Ingham et al. 1996. Functional-lesion investigation of developmental stuttering with positron emission tomography. *Journal of Speech and Hearing Research* 39 (1996), 208–27.

[12] Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 37)*, Francis Bach and David Blei (Eds.). PMLR, Lille, France, 448–456. https://proceedings.mlr.press/v37/ioffe15.html

[13] Thomas D Kehoe and Wikibooks Contributors. 2006. *Speech Language Pathology-Stuttering.* Kiambo Ridge.

[14] Tedd Kourkounakis et al. 2020. Detecting multiple speech disfluencies using a deep residual network with bidirectional long short-term memory. In *Proc. ICASSP 2020.* 6089–6093.

[15] Colin Lea et al. 2021. SEP-28k: A dataset for stuttering event detection from podcasts with people who stutter. In *Proc. ICASSP.* 6798–6802.

[16] Abdelrahman Mohamed, Hung-yi Lee, Lasse Borgholt, Jakob D Havtorn, Joakim Edin, Christian Igel, Katrin Kirchhoff, Shang-Wen Li, Karen Livescu, Lars Maaløe, et al. 2022. Self-supervised speech representation learning: A review. *arXiv preprint arXiv:2205.10643* (2022).

[17] Kevin P Murphy. 2012. *Machine Learning: A Probabilistic Perspective.* MIT press.

[18] Tae Jin Park, Naoyuki Kanda, Dimitrios Dimitriadis, Kyu J. Han, Shinji Watanabe, and Shrikanth Narayanan. 2022. A review of speaker diarization: Recent advances with deep learning. *Comput. Speech Lang.* 72, C (mar 2022), 34 pages. https://doi.org/10.1016/j.csl.2021.101317

[19] Adam Paszke et al. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in NIPS.* Curran Associates, Inc., 8024–8035.

[20] Vijayaditya Peddinti, Daniel Povey, and Sanjeev Khudanpur. 2015. A time delay neural network architecture for efficient modeling of long temporal contexts. In *Proc. Interspeech 2015.* 3214–3218. https://doi.org/10.21437/Interspeech.2015-647

[21] F. Pedregosa, G. Varoquaux, et al. 2011. Scikit-learn: Machine Learning in Python. *JMLR* 12 (2011), 2825–2830.

[22] Leonardo Pepino et al. 2021. Emotion recognition from speech using wav2vec 2.0 embeddings. In *Proc. Interspeech 2021.* 3400–3404. https://doi.org/10.21437/Interspeech.2021-703

[23] Steffen Schneider, Alexei Baevski, Ronan Collobert, and Michael Auli. 2019. wav2vec: Unsupervised pre-training for speech recognition . In *Proc. Interspeech 2019.* 3465–3469. https://doi.org/10.21437/Interspeech.2019-1873

[24] Björn W. Schuller, Anton Batliner, Shahin Amiriparian, Christian Bergler, Maurice Gerczuk, Natalie Holz, Pauline Larrouy-Maestri, Sebastian P. Bayerl, Korbinian Riedhammer, Adria Mallol-Ragolta, Maria Pateraki, Harry Coppock, Ivan Kiskin, Marianne Sinka, and Stephen Roberts. 2022. The ACM Multimedia 2022 Computational Paralinguistics Challenge: Vocalisations, Stuttering, Activity, & Mosquitos. In *Proceedings ACM Multimedia 2022.* ISCA, Lisbon, Portugal. to appear.

[25] Jui Shah, Yaman Kumar Singla, Changyou Chen, and Rajiv Ratn Shah. 2021. What all do audio transformer models hear? Probing acoustic representations for language delivery and its structure. *arXiv preprint arXiv:2101.00387* (2021).

[26] Shakeel Ahmad Sheikh et al. 2021. Machine learning for stuttering identification: Review, challenges & future directions. *arXiv preprint arXiv:2107.04057* (2021).

[27] Shakeel A. Sheikh et al. 2021. StutterNet: Stuttering detection using time delay neural network. In *Proc. 29th EUSIPCO.* 426–430.

[28] Shakeel A. Sheikh et al. 2022. Robust stuttering detection via multi-task and adversarial learning. In *Proc. 30th EUSIPCO.*

[29] Shakeel Ahmad Sheikh, Md Sahidullah, Fabrice Hirsch, and Slim Ouni. 2022. Introducing ECAPA-TDNN and Wav2Vec2.0 embeddings to stuttering detection. *arXiv preprint arXiv:2204.01564* (2022).

[30] Olabanji Shonibare, Xiaosu Tong, and Venkatesh Ravichandran. 2022. Enhancing ASR for stuttered speech with limited data using detect and pass. *arXiv preprint arXiv:2202.05396* (2022).

[31] Anne Smith and Christine Weber. 2017. How stuttering develops: The multifactorial dynamic pathways theory. *JSLHR* 60, 9 (2017), 2483–2505.

[32] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research* 15, 56 (2014), 1929–1958. http://jmlr.org/papers/v15/srivastava14a.html

[33] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K.J. Lang. 1989. Phoneme recognition using time-delay neural networks. *IEEE Transactions on Acoustics, Speech, and Signal Processing* 37, 3 (1989), 328–339. https://doi.org/10.1109/29.21701

[34] David Ward. 2017. *Stuttering and Cluttering: Frameworks for Understanding and Treatment.* Psychology Press.