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

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

To cite this version:

Shakeel A Sheikh, Md Sahidullah, Fabrice Hirsch, Slim Ouni. End-to-End and Self-Supervised Learning for ComParE 2022 Stuttering Sub-Challenge. ACM Multimedia 2022 Computational Paralinguistics Challenge (ComParE), Oct 2022, Lisbon, Portugal. hal-03728331

HAL Id: hal-03728331
https://inria.hal.science/hal-03728331
Submitted on 20 Jul 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
End-to-End and Self-Supervised Learning for ComParE 2022 Stuttering Sub-Challenge

Shakeel A. Sheikh¹, Md Sahidullah¹, Fabrice Hirsch², Slim Ouni¹
Université de Lorraine, CNRS, Inria, LORIA, F-54000, Nancy, France
Univ. Paul-Valery Montpellier, CNRS, Praxiling, Montpellier, France²

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 summarization 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:

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/stopped 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] explored 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 domain.

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
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:

$$L_{WCE} = \frac{1}{B} \sum_{b=1}^{B} \sum_{i=1}^{N_b} \alpha_i \log(p_i)$$

where $B$ is total batches, $N_b$ is the number of stuttered speech samples in a batch $b$, $\alpha_i = \frac{N_i}{N_b}$ ($N_i$ is the number of training samples, $C$ is the number of classes, $N_b$ is the number of training samples for class $i$), $p_i = \frac{\exp(c_{i,\hat{y}})}{\sum_{c_i} \exp(c_{i,\hat{y}})}$ 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 $E$ ($\theta_e$) followed by two parallel branches referred as DisfluentBranch $D$ ($\theta_d$) and FluentBranch $F$ ($\theta_f$). The embeddings generated by $E$ ($\theta_e$) 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\footnote{If the prediction of a sample in $F$ is not fluent, then the predictions of $D$ are taken into consideration to reveal the disfluent class category} sub categories with an overall objective function to optimize as:

$$L_{tot} = L_f(\theta_e, \theta_f) + L_d(\theta_e, \theta_d)$$

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

$$L_c = -\log \frac{\exp(c_{i, q_i} / \tau)}{\sum_{q_j \in Q} \exp(c_{i, q_j} / \tau)}$$

where $\exp(c_{i, q_i}) = c_i^T \frac{q_i}{||q_i||} \frac{||q_i||}{||q_i||}$ is the cosine similarity between the contextualized transformer vector $c_i$ and quantized vector $q_i$. Equation (3) is then augmented by a diversity loss. For details, please refer to the paper by Baevski et al. [4].

![Figure 1: Block Diagram of the Proposed Pipeline for SD.](image-url)
We train the MB StutterNet with two input features including Adam optimizer on a cross entropy loss function (Table 1 summarizes the results of our experiments on KSoF dataset). Each FC layer is followed by a ReLU activation function.

The shallow NNet is composed of two branches with $L_f$ layers. Each FC layer is followed by a ReLU activation function 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 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 ($L = L_f + L_d$; $L_f$: 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] 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%, 463%, 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 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...
We make use of the fact that each layer of the Wav2Vec2.0 model has different information and passing the summation over all layers (\(\sum_i L_i\)) 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. 

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 statistical pooling to both the features and using the downstream classification. 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.

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)

---

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, \(\sum_i\) is Concatenation, \(\sum_i L_i\): Summing Information Over all Embedding Layers).

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

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

<table>
<thead>
<tr>
<th>Approach</th>
<th>Val(%)</th>
<th>Test(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ComParE  [24]</td>
<td>30.2</td>
<td>37.6</td>
</tr>
<tr>
<td>auDeep  [24]</td>
<td>17.7</td>
<td>25.9</td>
</tr>
<tr>
<td>BoAWS  [24]</td>
<td>26.7</td>
<td>32.1</td>
</tr>
<tr>
<td>DeepSpectrum (CBL)  [24]</td>
<td>28.1</td>
<td>40.4</td>
</tr>
<tr>
<td>Fusion  [24]</td>
<td>28.7</td>
<td>38.3</td>
</tr>
<tr>
<td>(Ours) (SVM+L5)</td>
<td>36.9</td>
<td>41.0</td>
</tr>
<tr>
<td>(Ours) SVM+MFCCs L5</td>
<td>37.6</td>
<td>42.6</td>
</tr>
<tr>
<td>(Ours) MB StutterNet (\sum_i L_i)</td>
<td>41.0</td>
<td>42.7</td>
</tr>
</tbody>
</table>