CONFORMER-BASED SOUND EVENT DETECTION WITH SEMI-SUPERVISED LEARNING AND DATA AUGMENTATION

Koichi Miyazaki[1], *Tatsuya Komatsu*[2], *Tomoki Hayashi*[1,3],
Shinji Watanabe[4], *Tomoki Toda*[1], *Kazuya Takeda*[1]

[1]Nagoya University, Japan, [2]LINE Corporation, Japan,
[3]Human Dataware Lab. Co., Ltd., Japan, [4]Johns Hopkins University, USA
miyazaki.koichi@g.sp.m.is.nagoya-u.ac.jp

ABSTRACT

This paper presents a Conformer-based sound event detection (SED) method, which uses semi-supervised learning and data augmentation. The proposed method employs Conformer, a convolution-augmented Transformer that is able to exploit local features of audio data more effectively using CNNs, while global features are captured with Transformer. For SED, both global information on background sound and local information on foreground sound events are essential for modeling and identifying various types of sounds. Since Conformer can capture both global and local features using a single architecture, our proposed method is able to model various characteristics of sound events effectively. In addition to this novel architecture, we further improve performance by utilizing a semi-supervised learning technique, data augmentation, and post-processing optimized for each sound event class. We demonstrate the performance of our proposed method through experimental evaluation using the DCASE2020 Task4 dataset. Our experimental results show that the proposed method can achieve an event-based macro F1 score of 50.6% when using the validation set, significantly outperforming the baseline method score (34.8%). Our system achieved a score of 51.1% when using the DCASE2020 challenge's evaluation set, the best results among the 72 submissions.

Index Terms— Sound event detection, Conformer, Transformer, semi-supervised learning, data augmentation

1. INTRODUCTION

Sound event detection (SED) is a useful technique for helping us what is happening in an environment by identifying sounds. The recent development of neural network-based approaches such as convolutional neural network (CNN) [1], recurrent neural network (RNN) [2], [3], and convolutional RNN (CRNN) [4], [5] has yielded notable improvements in the performance of SED. However, a large amount of annotated data is required when training these neural network-based approaches. Among the annotated data, the data with a timestamp is known as *strongly labeled data*, while data with only tag information is known as *weakly labeled data*. Most SED systems assume that sufficient amounts of strongly labeled data available, but in reality, the annotation cost of applying strong labels to audio data is huge, and data collection is difficult. Therefore, the development of model training methods which are effective even when using a limited amount of strongly labeled data is important.

Researchers have proposed various SED models using a limited amount of strongly labeled data, employing weakly-supervised and semi-supervised learning techniques [6]–[8], which use weakly labeled and unlabeled data, respectively. Many approaches have been developed using these frameworks, and they have been reported to improve detection performance, even when using limited amounts of strongly labeled data. Data augmentation is another powerful technique, which is used to artificially generates new data through data manipulation, resulting in improved generalization performance [1], [9].

In our previous work for weakly-spuervised SED [7], in order to more efficiently capture global and local context information within audio feature sequences, we proposed using a self-attention mechanism, a technique originally proposed in the field of machine translation [10], which has been successfully applied to various audio and speech applications [11], [12] for SED. We also introduced a special token which allowed the use of weakly labeled data. *Gulati et al.* have recently proposed Conformer [13], a method of modeling the global and local dependencies of audio sequences, which has been reported to achieve state-of-the-art performance in automatic speech recognition (ASR). Conformer achieves high performance and efficient parameter reduction by combining self-attention with the use of a CNN. This is because self-attention is better at modeling long-range, global context information, while CNNs are better at extracting local features.

In this paper, we propose a Conformer-based SED model, which enables us to efficiently capture both local and global context information in audio feature sequences through Conformer blocks. To further improve performance, we also introduce semi-supervised learning based on mean teacher [14], data augmentation techniques such as time-shifting [8] and mixup [15], event-dependent post-processing refinement, and posterior-level score fusion. We conducted experimental evaluations using the DCASE2020 Task4 validation set and the DESED public evaluation set to investigate the effectiveness of our proposed network architecture and each of the proposed performance enhancement techniques.

2. PROPOSED METHOD

2.1. Network architecture

An overview of our proposed method is shown in Figure 1. It consists of three modules; a CNN-based feature extractor, Conformer blocks, and a position-wise classifier [7]. The architecture of the CNN-based feature extractor follows that of the baseline CRNN system used in the DCASE2020 Task4 [16], which consists of seven convolution layers. To match feature shape and network input size, we slightly modified the kernel size of the third and seventh average pooling layers from (2,2) to (1,2) and from (1,2) to (1,1), respectively. Our Conformer-based model uses the same architecture as our conventional Transformer-based model, except that the Transformer encoder is replaced with the Conformer block. The final

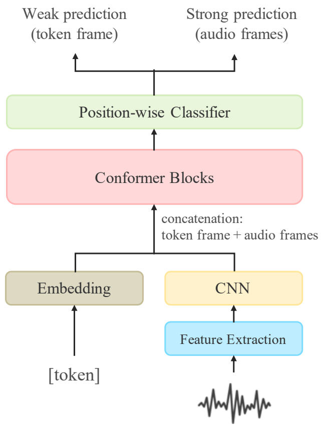

Figure 1: *Overview of our proposed method. For weak label prediction, an embedded token is concatenated at the beginning of the feature sequence obtained using a CNN. The first frame of the output corresponds to weak label prediction, and the other frames correspond to strong label prediction.*

position-wise classifier is a simple linear layer to calculate the posteriors corresponding to the sound event classes. Additionally, we introduce a special tagging token dedicated to weak label prediction [7], making it possible to explicitly summarize sequence-level information through the self-attention layers, similar to the special classification token used in BERT [17]. We attach the tagging token to the first frame of the feature sequence obtained with the CNN-based feature extractor. Then, network output corresponding to the first frame is used for weak label prediction, while the output of the other frames is used for strong label prediction. The tagging token is initialized to a 1-D zero vector, and the embedding layer expands the token's dimension.

2.2. Conformer block

The Conformer block is illustrated in Figure 2. It consists of three modules; a feed-forward module, a multi-head self-attention module, and a convolution module. The feed-forward module consists of a layer-normalization layer and a linear layer with a Swish activation function [18], followed by another linear layer. The first linear layer expands the dimensions of the input four times, while the second linear layer projects it back to the original input dimensions. After passing through both linear layers, we multiply the output by 0.5, using the original Conformer setting [13]. The multi-head self-attention module consists of a layer-normalization and multi-head self-attention with relative positional embedding used in Transformer-XL [19]. The convolution module consists of a layer-normalization layer, a point-wise convolution layer with a gated linear unit (GLU) activation function [20], and a 1-D depth-wise convolution layer. The depth-wise convolution layer is followed by a batch normalization layer, a Swish activation, and a point-wise convolution layer. Therefore, the correspondence between the input $\boldsymbol{X}$ and output $\boldsymbol{Y}$ of the Conformer block can be formulated as follows:

$$\tilde{\boldsymbol{X}} = \boldsymbol{X} + \frac{1}{2}\mathrm{FFN}(\boldsymbol{X}), \tag{1}$$

$$\boldsymbol{X}' = \tilde{\boldsymbol{X}} + \mathrm{MHSA}(\tilde{\boldsymbol{X}}), \tag{2}$$

$$\boldsymbol{X}'' = \boldsymbol{X}' + \mathrm{Conv}(\boldsymbol{X}'), \tag{3}$$

$$\boldsymbol{Y} = \mathrm{LayerNorm}(\boldsymbol{X}'' + \frac{1}{2}\mathrm{FFN}(\boldsymbol{X}'')), \tag{4}$$

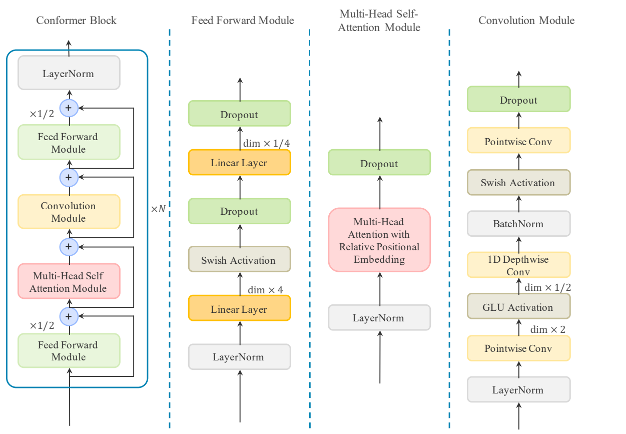

Figure 2: *Overview of Conformer block (on the left), with details of its components.*

where FFN$(\cdot)$, MHSA$(\cdot)$, Conv$(\cdot)$, and LayerNorm$(\cdot)$ refer to the feed-forward module, multi-head self-attention module, convolution module, and layer-normalization layer, respectively.

2.3. Semi-supervised learning

To further improve performance, we employed the mean teacher technique [14] as a semi-supervised learning method capable of using unlabeled data for training. We used a mean square error function as the consistency criterion and set the exponential ramp-up steps [21] and consistency cost to 10,000 and 2.0, respectively.

2.4. Data augmentation

For data augmentation, we employed Gaussian noise, frequency masking from SpecAugment [22], time-shifting [8], and mixup [15]. We added Gaussian noise to the input sequence at a 30 dB signal-to-noise ratio. Frequency masking replaces values in the frequency domain with zero. Time-shifting shifts a feature sequence along the time axis, and the overrun frames are then concatenated with the opposite end of the sequence. We randomly chose the frame-shift size by sampling from a Gaussian distribution, with a zero mean and a standard deviation of 90. The use of time-shifting helps prevent the model from learning bias in time event localization. Mixup smooths out the decision boundary by adding pseudo data generated by mixing different data points $(\boldsymbol{x}_1, \boldsymbol{x}_2)$ and the corresponding labels $(\boldsymbol{y}_1, \boldsymbol{y}_2)$. The mixup is formulated as follows:

$$\bar{\boldsymbol{x}} = \lambda \boldsymbol{x}_1 + (1-\lambda)\boldsymbol{x}_2, \tag{5}$$

$$\bar{\boldsymbol{y}} = \lambda \boldsymbol{y}_1 + (1-\lambda)\boldsymbol{y}_2, \tag{6}$$

where $\lambda \in [0, 1]$ is the mixing ratio. In this study, we randomly chose this value by sampling from a beta distribution with $\alpha = 0.2$.

2.5. Posterior-level score fusion

To improve generalization performance, we performed score fusion as a model ensemble technique. We first performed averaging over the raw posterior outputs $p(\boldsymbol{X})$ for inputs $\boldsymbol{X}$ of the multiple models, with different training settings, as follows:

$$p_{\text{fusion}}(\boldsymbol{X}) = \frac{1}{N}\sum_{n=1}^{N} p_n(\boldsymbol{X}), \tag{7}$$

where N represents the total number of models for our fusion. We then performed thresholding and applied post-processing, which will be explained in more detail in Section 2.6.

2.6. Event-dependent post-processing

To transform posterior to binary-valued sound event activation, we perform thresholding for the network output posterior. We then performed median filtering as post-processing to smooth the binary activation sequence. Since each sound event has different characteristics, such as its temporal structures, the optimal post-processing parameters depend on the individual sound event classes. Hence, we determine the optimal post-processing parameters for each sound event using the validation set by searching for the optimal threshold (from 0.1 to 0.9 in increments of 0.1) and optimal median filter size (from 1 to 31 in increments of 2).

3. EXPERIMENTAL EVALUATION

3.1. Experimental conditions

We conducted experimental evaluations using the DCASE2020 Task4 dataset [23] and DESED public evaluation set. The training set of the DCASE2020 Task4 dataset included 2,584 synthesized audio clips with strong labels, 1,578 real audio clips with weak labels, and 14,412 real audio clips without labels. Since the audio clips were collected from YouTube, the dataset included various types of audio file formats with different recording settings (e.g., 16 kHz sampling rate vs. 44.1 kHz sampling rate). To address this issue, we first converted all of the audio clips into a 1-channel, 16-bit format, at a 16 kHz sampling rate using SoX [24]. We then normalized each audio clip to -3 dBFS and removed the direct current component by applying a 10 Hz high-pass filter.

As an input to the proposed system, we extracted 64-dimensional log-Mel filterbanks from the audio clips. The window and hop sizes were 1,024 and 323 points, respectively, at 16 kHz sampling, to extract 496 frames (a multiple of 8) from 10 second audio signals. We performed zero-padding for shorter sequences and truncation for longer sequences from their last frames to equalize the length of the feature sequences. We then performed normalization for each bin so that the feature sequences had zero mean and unit variances over the training data. The normalization procedure can be expressed as follows:

$$\bar{\boldsymbol{X}}[t] = (\boldsymbol{X}[t] - \boldsymbol{\mu}) / \boldsymbol{\sigma}, \tag{8}$$

where $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ represent the mean and variance calculated from the training dataset, respectively.

To evaluate the performance of our proposed method, we compared the following models:

Baseline: The DCASE2020 Task4 official baseline SED system [16]. Its architecture consisted of a convolutional recurrent neural network (CRNN), trained using the mean teacher semi-supervised learning technique [14].

Transformer-based (Ours): Our proposed Transformer-based model. The number of attention units and attention heads were 512 and 16, respectively. The Transformer encoder was stacked 3 times and the dropout rate was set to 0.1.

Conformer-based (Ours): Our proposed Conformer-based model. The number of attention units and attention heads were 144 and 4, respectively. The Conformer block was stacked 4 times, the kernel size of the depth-wise convolution was 7, and the dropout rate was set to 0.1.

Table 1: *Event-wise performance using DCASE2020 Task4 validation set. using event-based macro F1 scores [%].*

Event class	Baseline	Transformer	Conformer
Alarm bell ringing	36.9	44.8	**45.5**
Blender	31.4	36.0	**38.7**
Cat	**43.6**	36.4	38.0
Dishes	25.5	28.1	**28.2**
Dog	20.3	**24.0**	23.1
Electric shaver	37.8	50.8	**52.1**
Frying	24.2	**44.8**	36.4
Running water	31.9	30.1	**33.3**
Speech	48.0	52.9	**57.3**
Vacuum cleaner	47.7	62.2	**64.6**
Average	34.8	41.0	**41.7**

We used RAdam [25] optimizer with a batch size of 128 and a learning rate of 0.001. Each batch contained strong, weak, and unlabeled data at a ratio of 1:1:2. We multiplied the learning rate by 0.1 every 10,000 mini-batch iterations and trained for 30,000 mini-batch iterations. Note that we chose different hyper-parameters for the attention units and heads for Transformer and Conformer-based models. Since we found that the optimal hyper-parameters are different between them.

The evaluation metrics were the event-based macro F1 score (EB-F1), the segment-based macro F1 score (SB-F1), and the polyphonic sound event detection score (PSDS) [26]. These metrics were calculated using sed_eval toolkit [27]. The segment length for segment-based evaluation was set to 1 second. The event-based metrics were calculated using both the onset and offset of detection. The allowable length of detection errors was set to 200 ms for the onsets and 200 ms / 20 % of the offsets' event length. We computed PSDS using 50 thresholds from 0.01 to 0.99.

3.2. Experimental Results

3.2.1. Effects of model architecture

First, we investigated the effects of the model architecture. When performing this comparison, we used post-processing and semi-supervised learning but did not use data augmentation. From the results shown in Table 1, we can observe that both of the proposed models outperformed the baseline model even without data augmentation. When we compare the event-wise performance of the Conformer-based and the Transformer-based models, as shown in Table 1, these results show that the Conformer-based model outperformed the Transformer-based model for most of the sound events. This result suggests that local features are essential for SED and that Conformer's ability to handle local features effectively contributed to its improved performance.

Next, we investigated the effects of the kernel size of depth-wise convolution in the Conformer blocks. The results in Table 2 show that the kernel size used for depth-wise convolution affects performance and that 7 was optimum kernel size. In this study, the resolution of one frame of feature sequence input to the Conformer blocks corresponds to 0.16 seconds. Therefore, a kernel size of 7 corresponds to a receptive field of 1.12 seconds.

3.2.2. Effects of post-processing

Next, we investigated the effects of post-processing. We fixed the threshold to 0.5 for all sound event classes for the model without post-processing and did not apply the median filter. Table 3 shows the results with and without post-processing. From these results, we can confirm that post-processing improves detection performance,

Table 2: *Effects of kernel size used for depth-wise convolution on event-based and segment-based macro F1 scores using Conformer-based method.*

Kernel size	EB-F1[%]	SB-F1 [%]
3	40.8	66.4
7	**41.7**	**67.2**
15	41.6	64.6
31	39.8	64.0

Table 3: *Effects of post-processing (p.p.) on event-based and segment-based macro F1 scores.*

Method	EB-F1[%]	SB-F1 [%]
Transformer w/o p.p.	28.6	64.4
Transformer w/ p.p.	41.0	**69.3**
Conformer w/o p.p.	34.4	64.5
Conformer w/ p.p.	**41.7**	67.2

Table 4: *Effects of data augmentation on event-based and segment-based macro F1 scores using Conformer-based method. 95% confidence interval was calculated by 5 experimental results*

Data augmentation	EB-F1 [%]	PSDS
Nothing	41.8 ± 1.03	0.592 ± 0.017
Gaussian noise	41.4 ± 1.12	0.574 ± 0.011
Frequency masking	41.4 ± 1.15	0.586 ± 0.017
Time-shifting	42.9 ± 1.36	0.616 ± 0.022
Mixup	42.4 ± 1.93	0.620 ± 0.016
Time-shifting & mixup	$\mathbf{46.0 \pm 0.98}$	$\mathbf{0.641 \pm 0.010}$
All	45.4 ± 1.25	0.633 ± 0.014

especially in relation to the event-based macro F1 score. This is because the event-based metric is more sensitive to deletion or insertion errors than the segment-based metric.

3.2.3. *Effects of data augmentation*

Next, we investigated the effects of data augmentation. Table 4 shows the results for the Conformer-based model without data augmentation, and various methods of data augmentation. These results show that Gaussian noise and frequency masking did not improve the performance, while time-shifting and mixup could improve the performance, although they are not statistically significant. However, a combination of time-shifting and mixup yielded a statistically significant improvement from the original method and individual data augmentation techniques.

3.2.4. *Effects of score fusion*

Finally, we investigated the effects of posterior-level score fusion. To compare the effectiveness of various model selection strategies, we selected the following three sets of models:

Conformer fusion: Score fusion using the top 8 Conformer-based models, when each model was trained with different hyper-parameter settings.

Transformer fusion: Score fusion using the top 7 Transformer-based models, when each model was trained with different hyper-parameter settings.

Conformer & Transformer fusion: Score fusion using the top 8 Conformer-based models and top 7 Transformer-based models.

Table 5: *Effects of score fusion. Note that the "public eval" results represent results using the DESED public evaluation dataset. This is not the official challenge evaluation set.*

	Validation		Public eval	
Method	EB-F1 [%]	PSDS	EB-F1 [%]	PSDS
Baseline	34.8	0.610	-	-
Transformer fusion	47.3	0.643	51.1	0.691
Conformer fusion	**50.6**	**0.700**	**55.8**	**0.747**
Conf. & Trans. fusion	49.8	0.626	55.0	0.737

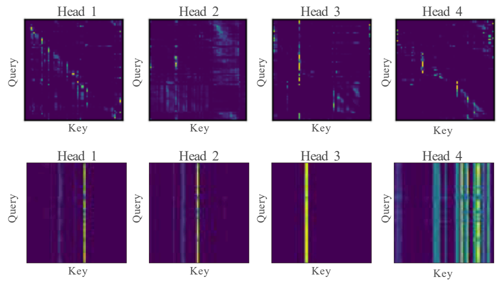

Figure 3: *Visualization of various attention weights. The top row shows attention weights examples of the Conformer-based model, while the bottom row shows attention weights examples of the Transformer-based model. Only vertical patterns appear in the Transformer-based model, but vertical and diagonal patterns appear in the Conformer-based model.*

Score fusion results are shown in Table 5. The post-processing hyper-parameter for the DESED public eval set was tuned by using the validation set. From these results, we can see that score fusion yields further performance improvement, significantly outperforming the baseline. The best model achieved the event-based F1 score of 50.6% a the PSDS of 0.700. This particular model is the one evaluated using the DCASE2020 challenge's evaluation set. It achieved that of 51.1%, winning first place among 72 DCASE2020 Task4 submissions.

3.3. Visualization of attention weights

In order to further investigate the use of self-attention, we created a visualization of attention weights, which are shown in Figure 3. We observed that there are diagonal and vertical patterns that appear in the attention weight visualizations for the Conformer-based method, which implies that the proposed method is capable of capturing both global and local context information from the feature sequence.

4. CONCLUSION

In this paper, we have presented a Conformer-based SED method, which was developed using a self-attention architecture embedded inside Conformer blocks. We also employed the data augmentation techniques, the event-dependent post-processing, and the score fusion. Our experimental results, when using the DCASE2020 Task4 validation set, demonstrated that these techniques improved SED performance and that our proposed system significantly outperform the baseline system, achieving an event-based macro F1 score of 50.6%. In future work, we will investigate our system's event-wise performance more carefully in order to develop a more effective model ensemble technique. We would also like to explore the further integration of source separation techniques into SED and publish our proposed method as reproducible, open-source code.

5. REFERENCES

[1] N. Takahashi, M. Gygli, B. Pfister, and L. Van Gool, “Deep convolutional neural networks and data augmentation for acoustic event detection,” *arXiv preprint arXiv:1604.07160*, 2016.

[2] T. H. Vu and J.-C. Wang, “Acoustic scene and event recognition using recurrent neural networks,” *Detection and Classification of Acoustic Scenes and Events*, 2016.

[3] T. Hayashi, S. Watanabe, T. Toda, *et al.*, “Duration-controlled LSTM for polyphonic sound event detection,” *IEEE/ACM TASLP*, vol. 25, no. 11, pp. 2059–2070, 2017.

[4] E. Cakır, G. Parascandolo, T. Heittola, H. Huttunen, and T. Virtanen, “Convolutional recurrent neural networks for polyphonic sound event detection,” *IEEE/ACM TASLP*, vol. 25, no. 6, pp. 1291–1303, 2017.

[5] S. Adavanne, P. Pertilä, and T. Virtanen, “Sound event detection using spatial features and convolutional recurrent neural network,” in *Proc. ICASSP*, 2017, pp. 771–775.

[6] T.-W. Su, J.-Y. Liu, and Y.-H. Yang, “Weakly-supervised audio event detection using event-specific gaussian filters and fully convolutional networks,” in *Proc. ICASSP*, 2017, pp. 791–795.

[7] K. Miyazaki, T. Komatsu, T. Hayashi, *et al.*, “Weakly-supervised sound event detection with self-attention,” in *Proc. ICASSP*, 2020, pp. 66–70.

[8] L. Delphin-Poulat and C. Plapous, “Mean teacher with data augmentation for DCASE 2019 task 4,” Orange Labs Lannion, France, Tech. Rep., 2019.

[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] A. Vaswani, N. Shazeer, N. Parmar, *et al.*, “Attention is all you need,” in *Proc. NIPS*, 2017, pp. 5998–6008.

[11] S. Karita, N. Chen, T. Hayashi, *et al.*, “A comparative study on transformer vs RNN in speech applications,” in *Proc. ASRU*, 2019, pp. 449–456.

[12] Y. Zhu, T. Ko, D. Snyder, B. Mak, and D. Povey, “Self-attentive speaker embeddings for text-independent speaker verification.,” in *Proc. INTERSPEECH*, 2018, pp. 3573–3577.

[13] A. Gulati, J. Qin, C.-C. Chiu, *et al.*, “Conformer: Convolution-augmented transformer for speech recognition,” *arXiv preprint arXiv:2005.08100*, 2020.

[14] A. Tarvainen and H. Valpola, “Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results,” in *Proc. NIPS*, 2017, pp. 1195–1204.

[15] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, “Mixup: Beyond empirical risk minimization,” *arXiv preprint arXiv:1710.09412*, 2017.

[16] N. Turpault and R. Serizel, “Training sound event detection on a heterogeneous dataset,” *arXiv preprint arXiv:2007.03931*, 2020.

[17] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of deep bidirectional transformers for language understanding,” in *Proc. NAACL*, 2019, pp. 4171–4186.

[18] P. Ramachandran, B. Zoph, and Q. V. Le, “Searching for activation functions,” *arXiv preprint arXiv:1710.05941*, 2017.

[19] Z. Dai, Z. Yang, Y. Yang, *et al.*, “Transformer-XL: Attentive language models beyond a fixed-length context,” in *Proc. ACL*, 2019, pp. 2978–2988.

[20] Y. N. Dauphin, A. Fan, M. Auli, and D. Grangier, “Language modeling with gated convolutional networks,” in *Proc. ICML*, 2017, pp. 933–941.

[21] S. Laine and T. Aila, “Temporal ensembling for semi-supervised learning,” *arXiv preprint arXiv:1610.02242*, 2016.

[22] D. S. Park, W. Chan, Y. Zhang, *et al.*, “SpecAugment: A simple data augmentation method for automatic speech recognition,” in *Proc. INTERSPEECH*, 2019, pp. 2613–2617.

[23] `http://dcase.community/challenge2020/`.

[24] `http://sox.sourceforge.net/`.

[25] L. Liu, H. Jiang, P. He, *et al.*, “On the variance of the adaptive learning rate and beyond,” in *Proc. ICLR*, 2020.

[26] 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.

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