CONVOLUTION-AUGMENTED TRANSFORMER FOR SEMI-SUPERVISED SOUND EVENT DETECTION

Technical Report

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,
komatsu.tatsuya@linecorp.com, hayashi@hdwlab.co.jp
shinjiw@ieee.org, tomoki@icts.nagoya-u.ac.jp, kazuya.takeda@nagoya-u.jp

ABSTRACT

In this technical report, we describe our submission system for DCASE2020 Task4: sound event detection and separation in domestic environments. Our model employs conformer blocks, which combine the self-attention and depth-wise convolution networks, to efficiently capture the global and local context information of an audio feature sequence. In addition to this novel architecture, we further improve the performance by utilizing a mean teacher semi-supervised learning technique, data augmentation, and post-processing optimized for each sound event class. We demonstrate that the proposed method achieves the event-based macro F1 score of 50.7% on the validation set, significantly outperforming that of the baseline score (34.8%).

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

1. INTRODUCTION

This technical report describes our submission system for DCASE2020 Challenge Task4: sound event detection (SED) and separation in domestic environments [1]. The goal of this task is to build a system for the detection of sound events using real data either weakly labeled or unlabeled and simulated data that is strongly labeled (with timestamps). To address this task, we propose two neural network models that utilize the *self-attention* mechanism;

- Transformer-based model [2], [3],
- Conformer-based model [4].

These models can efficiently capture both local and global context information of an audio feature sequence through the stack of CNN and self-attention layers. Besides, to further improve the performance, we implement

- semi-supervised learning based on mean teacher [5],
- data augmentation techniques, such as time-shifting [6] and mixup [7],
- post-processing refinement,
- posterior-level score fusion.

We conduct experimental evaluations on the DCASE2020 Task4 validation set to investigate the effectiveness of the proposed network architecture and each of the implemented techniques. The experimental results show that the proposed models significantly outperform the baseline system, achieving the event-based macro F1 score of 46.0% with the best single system and that of 50.7% with the score fusion.

2. PROPOSED METHOD

2.1. Feature extraction

We extract 64-dimensional log-Mel filterbanks from the input audio. The window size and the hop size are 1,024 points and 323 points, respectively, in 16 kHz sampling. We fix the length of the feature sequence to 496 frames (corresponding to around 10 seconds). To make the length of feature sequences the same, we perform zero-padding for shorter sequences and truncation for longer sequences from their last frames. Then, we perform the normalization to make the feature sequences have zero means and unit variances over the training data.

2.2. Network architecture

Inspired by the great success of the self-attention architectures in various fields [2]–[4], [8], [9], we propose two neural network models for SED; Transformer-based model and Conformer-based model. The Transformer-based model consists of three modules; a CNN-based feature extractor, Transformer blocks, and a position-wise classifier [3]. The architecture of the CNN-based feature extractor follows the baseline system of DCASE2020 Task4 [1], which consists of three or seven convolution layers. To match with the input size, we slightly modify the third and seventh average pooling layers' kernel size from (2, 2) to (1, 2) and from (1, 2) to (1, 1), respectively. The Transformer block follows the architecture in [2], which consists of a multi-head self-attention layer, a layer-normalization layer, and a linear layer with a rectified linear unit (ReLU) activation function followed by another layer normalization. The final position-wise classifier is a simple linear layer to calculate the final outputs that correspond to the sound event types.

The Conformer-based model is the same architecture as the Transformer model, except that the above Transformer block is re-

placed with the Conformer block. The Conformer block consists of two feed-forward modules that sandwich 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 [10] followed by another linear layer. The first linear layer expands the dimension four times, and the second one projects back to the original input dimension. After both the linear layers, we multiply 0.5 with the outputs by following the original Conformer's setting [4]. The convolution module consists of a layer-normalization layer, a point-wise convolution layer with a gated linear unit (GLU) activation function [11], and a 1D 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. We set a kernel size of the depth-wise convolution to seven instead of following the original setting [4] due to the shorter input feature sequence.

Additionally, we introduce a special tag token for the weakly labeled training [3], making it possible to explicitly summarize the sequence-level information through the self-attention layers, similar to the special classification token used in BERT [12]. We attach the tag token as the first frame of the latent feature sequence transformed through the CNN-based feature extractor. Therefore, the network output corresponding to the first frame is used for the weak label prediction, while those of the other frames are used for the strong label prediction. The tag token is initialized with a uniform distribution and it is updated through the training.

2.3. Semi-supervised learning

To further improve the performance, we employ the mean teacher technique [5] as one of the typical semi-supervised training methods capable of using unlabeled data in training. We use a mean square error function as the consistency criterion, and set the exponential ramp-up steps [13] and the consistency cost to 10,000 and 2.0, respectively.

2.4. Data augmentation

For data augmentation, we employ time-shifting [6] and mixup [7]. The time-shifting shifts a feature sequence on the time axis, and overrun frames are concatenated with the opposite side of the sequence. We randomly choose the shift size by sampling from a normal distribution with a zero mean and a standard deviation of 90. The use of the time-shifting is helpful for preventing the network from inappropriately learning the location information over the sequence.

The mixup smoothes 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;

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

$$\bar{\boldsymbol{y}} = \lambda \boldsymbol{y}_1 + (1-\lambda)\boldsymbol{y}_2, \tag{2}$$

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

2.5. Post-processing

To determine the sound event activation, we perform thresholding for the network output posterior. Then, we perform median filtering as post-processing to smooth the detected activation sequence. Since each sound event has different characteristics, such as temporal structures, the optimal post-processing parameters depend on the individual sound events. Hence, we determine the optimal post-processing parameters for each sound event using the validation set. We search the optimal threshold and median filter size from 0.1 to 0.9 in increments of 0.1, and from 1 to 31 in increments of 2, respectively.

2.6. Score fusion

To improve generalization performance, we perform score fusion as a model ensemble technique. We first average the raw posterior outputs of the multiple models with different training settings. Then, we perform thresholding and apply post-processing, as mentioned in Section 2.5.

3. EXPERIMENTAL EVALUATION

3.1. Experimental conditions

We conducted experimental evaluations using DCASE2020 Task4 dataset [1]. The dataset included 2,584 audio clips with a strong label, 1578 audio clips with a weak label, and 14,412 unlabeled audio clips. Since the audio clips were collected from YouTube, the dataset included various audio clips 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 to be 1 ch, 16 bit, and 16 kHz sampling rate using sox [14]. We then normalized each audio clip to be -3 dBFS and removed a direct current component by applying a 10 Hz high-pass filter.

To verify the performance, we compared the following models:

Baseline: The DCASE2020 Task4 official baseline system [15]. The architecture was a convolutional recurrent neural network (CRNN), and it was trained with the mean-teacher semi-supervised learning technique [5]. We used the numbers provided in the official HP.

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

Conformer (Ours): The proposed Conformer-based model. The number of attention units and that of the attention heads were 144 and 4, respectively. The kernel size of the depth-wise convolution was 7. The dropout rate was set to 0.1. The number of Conformer blocks was varied from 2 to 10.

We used RAdam [16] optimizer with a batch size of 128 and a learning rate of 0.001. We multiplied the learning rate by 0.1 every 10000 iteration and continued to train until 30,000 iterations. We used a GPU (NVIDIA Titan pascal) to train the models. It took around 12 hours to finish the training. The detailed training condition is shown in Table 1.

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) [17]. These metrics were calculated using sed_eval toolkit [18]. The segment length in the segment-based evaluation was set to 1 second. The event-based metrics were calculated using both the onset and offset of the detection. The allowable length of detection errors was set to 200 ms for the onsets and 200 ms / 20 % of the event length for the offsets. We computed PSDS using 50 thresholds from 0.01 to 0.99.

Table 1: *Network training configuration.*

# of training samples	# of strong = 2,584 # of weak = 1,578 # of unlabeled = 14,412
Batch size	128
# of iterations	30,000
Optimizer	RAdam [16]
Learning rate	0.001
Consistency cost	2.0
Rampup steps	10,000

Table 2: *Effects of model architectures.*

Method	EB-F1[%]	SB-F1 [%]
Baseline	35.6	-
Transformer	41.0	**69.3**
Conformer	**41.7**	67.2

3.2. Experimental Results

3.2.1. Effects of model architecture

First, we investigated the effects of the model architecture. In a comparison of the model architectures, we used the post-processing and the mean teacher learning but we did not use the data augmentation for the proposed method. From the result shown in Table 2, we can observe that both the proposed models outperform the baseline even if we do not use data augmentation, revealing the effectiveness of the self-attention architecture for SED.

Next, we investigated the effects of the number of Conformer blocks. The result in Table 3 shows that the number of blocks affects the performance and 4 was the best. We used this configuration in the following experiments.

3.2.2. Effects of post-processing

Next, we investigated the effects of the post-processing. For the model without post-processing, we fixed the threshold to 0.5 for all sound event classes and did not apply the median filter. Table 4 shows the result with or without post-processing. From the result, we can confirm that the post-processing improves the performance especially on the event-based macro F1 score. This is because the event-based metric is more sensitive to the deletion or insertion errors than the segment-based metric.

3.2.3. Effects of data augmentation

Next, we investigated the effects of the data augmentation. Table 5 shows the result of the Conformer-based model with or without each data augmentation method. The result shows that both time-shifting and mixup improve the performance and that a combination of these two methods yields further performance improvement.

Table 3: *Effects of the number of Conformer blocks.*

# of blocks	EB-F1[%]	SB-F1 [%]
2	39.39	65.91
3	40.29	67.03
4	**41.30**	**67.17**
5	38.47	66.18
6	39.44	67.14
7	39.44	66.29
8	38.18	64.96
9	37.82	65.90
10	36.49	65.10

Table 4: *Effects of post-processing (p.p.).*

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 5: *Effects of data augmentation.*

Method	EB-F1 [%]	PSDS
w/o data augmentation	41.7	0.593
w/ time-shifting	41.4	0.616
w/ mixup	42.5	0.620
w/ time-shifting & mixup	**46.0**	**0.643**

3.2.4. Effects of score fusion

Finally, we investigated the effects of the score fusion. We compared the following three models;

Conformer fusion: Score fusion with the top 8 Conformer models, where each model was trained with different hyperparameter settings.

Transformer fusion: Score fusion with the top 7 Transformer models, where each model was trained with of different hyperparameter settings.

Conformer & Transformer fusion: Score fusion with the top 8 Conformer models and the top 7 Transformer models.

The score-fusion result is shown in Table 6 From the result, the score fusion yields further performance improvements and significantly outperforms the baseline. The best model achieves the event-based F1 score of 50.6 and the PSPD of 0.700.

4. CONCLUSION

In this technical report, we have described our submission system for DCASE2020 Task4. Our system has been developed by using the self-attention architecture including the Transformer and the Conformer blocks, the data augmentation techniques, the class-dependent post-processing, and the score fusion. The experimental

Table 6: *Effects of score fusion.*

Method	EB-F1 [%]	PSDS
Baseline	35.6	0.626
Conformer fusion	**50.6**	**0.700**
Transformer fusion	47.3	0.643
Conformer & Transformer fusion	49.8	0.626

results using the validation set have demonstrated that these techniques are helpful for improving the sound event detection performance, and our system significantly outperforms the baseline. In future work, we will investigate the class-wise performance more carefully to develop more effective model ensemble technique, and further integrate source separation techniques to sound event detection.

5. REFERENCES

[1] http://dcase.community/challenge2020/.

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

[3] M. Koichi, K. Tatsuya, H. Tomoki, *et al.*, "Weakly-supervised sound event detection with self-attention," in *ICASSP*, 2020, pp. 66–70.

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

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

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

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

[8] 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.

[9] Y. Fujita, N. Kanda, S. Horiguchi, *et al.*, "End-to-end neural speaker diarization with self-attention," in *Proc. ASRU*, 2019, pp. 296–303.

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

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

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

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

[14] http://sox.sourceforge.net/.

[15] https://github.com/turpaultn/dcase20_task4/tree/public_branch/baseline.

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

[17] 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.

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