SEMI-SUPERVISED SOUND EVENT DETECTION SYSTEM FOR DCASE 2023 TASK4A

Technical Report

Wenxin Duo[1], *Xiang Fang*[2], *Jie Li*[2],

[1] Tianjin University, School of Electrical and Information Engineering, Tianjin, China, hiedean@tju.edu.cn
[2] China Telecom Corporation Ltd., Data&AI Technology Company, Beijing, China, {fangx1, lij86}@chinatelecom.cn

ABSTRACT

In this technical report, we describe our systems for DCASE 2023 Challenge Task4a. Our systems are mainly based on Frequency Dynamic Convolutional Recurrent Neural Network (FDYCRNN) and Mutual Mean Teaching (MMT) semi-supervised strategy. In order to prevent overfitting, we adopt data augmentation using mixup, frame shift, SpecAugment, FilterAugment, Interpolation Consistency Training (ICT) and Shift Consistency Training (SCT). Besides, we utilize strongly labeled AudioSet data as external data and several pretrained models to further improve performance, and try an ensemble of multiple systems with different pretrained models to enhance the generalization capability of our system.

Index Terms— Sound event detection, semi-supervised learning, pretrained model, mutual mean teaching

1. INTRODUCTION

Sound Event Detection (SED) is designed to detect sound event categories and their corresponding onset and offset times (timestamps) within sound clips. In supervised learning, to accomplish this task better, a large amount of data with strong labels is required. However, labeling the collected data manually is extremely expensive, thus, high-quality datasets are difficult to obtain. To solve this problem, many semi-supervised learning methods that use weakly labeled data and unlabeled data with limited strongly labeled data are proposed.

Labels for weakly labeled data include only sound event categories without event timestamps and unlabeled datasets where labels are completely uninformative. In semi-supervised learning, pseudo-label are often used to deal with non-strongly labeled datasets. [1, 2, 3] first train models with certain performance, and use the model to predict the strong label of the non-strong label data to obtain the pseudo-strong label. In order to take full advantage of weakly labeled datasets and unlabeled datasets, [4] proposed to use the mean teacher (MT) framework for SED. More specifically, the MT framework performs supervised training on labeled data, and self-supervised training on both labeled and unlabeled data. The self-supervised part of the MT uses the teacher model to predict. While the loss is calculated between the teacher prediction and the student prediction. To some extent, the teacher model is used for pseudo-label prediction. In order to further improve the robustness and stability of the self-supervised part of the MT, [5] proposed MMT. MMT leverages the output of peer networks to mitigate noise in pseudo-labels, and leverages the complementarity of this output to optimize each other.

In this report, we use FDYCRNN to build MMT. To further improve the model performance, we also incooperate several pretrained models and data augmentations in .

2. METHOD

2.1. Frequency Dynamic Convolution for SED

Nam et al. demonstrated that the frequency-domain energy distribution of different events in the SED is different[6]. Convolutional neural networks (CNNs), commonly used in deep learning, are translation invariant But for SED, this property may misjudge two events with similar energies but different distributions, thus degrading the overall performance of the SED task.

Dynamic convolution enhances representation capability of CNNs by applying input-adaptive kernel on convolution layer. By extracting attention weights for the weighted sum of basis kernels, dynamic convolution generates appropriate kernel for given input. This means dynamic convolution can overcome the translation invariance of convolution. Here, we use the frequency dynamic convolution proposed by [6][1].

2.2. Mutual mean teaching

MMT framework generates soft pseudo labels by collaboratively training two same networks with different initializations. In addition to the hard labels, two collaborative networks also generate on-line soft pseudo labels by network predictions for training each other. However, such soft labels are generally not perfect because of the training errors. To avoid two networks collaboratively bias each other, the past temporally average model of each network instead of the current model is used to generate the soft pseudo labels for the other network. Both off-line hard labels and on-line soft pseudo labels are utilized jointly to train the two collaborative networks.

We denote the two collaborative networks as $F(\cdot|\theta_1)$ and $F(\cdot|\theta_2)$ respectively. To simultaneously train the coupled networks, we feed the same audio batch to the two networks but with different data augmentations. Each target-domain audio can be denoted by x and x for the two networks, and their pseudo label confidences can be predicted as $F(x|\theta_1)$ and $F(x'|\theta_2)$. One naive way to train the collaborative networks is to directly utilize the above pseudo label confidence vectors as the soft pseudo labels for training the other network. However, in such a way, the two networks' predictions might converge to equal each other and the two networks

[1] https://github.com/frednam93/FDY-SED

lose their output independences. The classification errors as well as pseudo label errors might be amplified during training. In order to avoid error amplification, we propose to use the temporally average model of each network to generate reliable soft pseudo labels for supervising the other network. Specifically, the parameters of the temporally average models of the two networks at current iteration T are denoted as $E^T[\theta_1]$ and $E^T[\theta_2]$ respectively, which can be calculated as

$$\begin{aligned} E^T[\theta_1] &= \alpha E^{T-1}[\theta_1] + (1-\alpha)\theta_1, \\ E^T[\theta_2] &= \alpha E^{T-1}[\theta_2] + (1-\alpha)\theta_2, \end{aligned} \tag{1}$$

where $E^{T-1}[\theta_1]$, $E^{T-1}[\theta_1]$ indicate the temporal average parameters of the two networks in the previous iteration $T-1$, the initial temporal average parameters are $E^{(0)}[\theta_1] = \theta_1$, $E^{(0)}[\theta_2] = \theta_2$, and α is the ensembling momentum to be within the range $[0, 1)$. The robust soft pseudo label supervisions are then generated by the two temporal average models as $F(x'|E^T[\theta_1])$ and $F(x|E^T[\theta_2])$ respectively. The soft classification loss for optimizing θ_1 and θ_2 with the soft pseudo labels generated from the other network can therefore be formulated as

$$\begin{aligned} L_{\theta_1} &= L_{bce,strong}(F(x|\theta_1)_{strong}, y_{strong}) \\ &\quad + L_{bce,weak}(F(x|\theta_1)_{weak}, y_{weak}) \\ &\quad + \lambda L_{mse}(F(x|\theta_1), F(x|E^T[\theta_2])), \\ L_{\theta_2} &= L_{bce,strong}(F(x'|\theta_2)_{strong}, y_{strong}) \\ &\quad + L_{bce,weak}(F(x'|\theta_2)_{weak}, y_{weak}) \\ &\quad + \lambda L_{mse}(F(x'|\theta_2), F(x'|E^T[\theta_1])), \\ L_{total} &= L_{\theta_1} + L_{\theta_2}, \end{aligned} \tag{2}$$

2.3. Pretrained model

The pretrained models we use in our system are AST[7], PANNS[8], PASST[9], HTSAT[10] and BEATs[11]. AST is the first convolution-free, purely attention-based model for audio classification. While PANNS achieves the state-of-the-art performance (0.439 mAP on AudioSet) in CNN based architecture. PASST is also a transformer-based audio classifier, while it requires less computing consumption and is faster to train. HTSAT is a lightweight audio transformer with a hierarchical structure. BEATs is an iterative audio pretraining framework to learn bidirectional encoder representation from audio transformers, where an acoustic tokenizer and an audio self-supervised learning model are optimized by iterations. Because these pretrained models have different temporal resolutions, we use interpolation or adaptive pooling to align the features extracted by these pretrained models with CNN features. These aligned features are regard as frame-level features and then concatenated with CNN features from SED model. Since we found in the previous experiments that the effect of the features of BEATs is much better than that of other pretrained models, while using the BEATs features fixedly, we selected one of the other pretrained models, and use the features of it with BEATs features and CNN features to fuse. There are two options for feature fusion. One is to directly concatenate the three kinds of features and fuse them by 1×1 convolution to a fixed dimension. The other is to fuse two pretrained features first, and then fuse it with the CNN features.

2.4. Weak prediction

As PSDS2 focuses on avoiding confusion between classes rather than the localization of sound events, we only predict weak labels of clips and set timestamp to start and end of the entire duration of the audio. This method can greatly improve the PSDS2 scores. During the training stage, we don't use the strongly labeled data. Instead, all the strongly labeled datasets are relabeled weakly. The loss is calculated as the sum of supervised weak loss and self-supervised weak loss.

3. DATASET AND DATA PROCESS

All of the dataset we use in our training are described as follows

- Unlabeled in domain training set: 14412 clips.
- Synthetic strongly labeled training set: 10000 clips.
- Weakly labeled training set: 1578 clips.
- Strongly labeled validation set: 1168 clips.
- Strongly labeled AudioSet data as external data: 3470 clips.

All audio are resampled to 16kHz and down sampled to mono. We use log-mel energies as acoustic feature and extract 128 dimensional log-mel spectrogram using 2048 STFT window with a hop length of 256. In order to deal with the variable lengths of audio, we set a maximum padding length. All shorter feature will be zero padding to the padding length. When it is longer, it will be truncated. In this work, maximum padding length is set to 626.

During the training stage, we apply mixup, frame shift, SpecAugment[12] and FilterAugment[13] these four data augmentations in our system to increase the robustness. Besides, we also apply ICT and SCT. More implement details about ICT and SCT are available in [14].

4. EXPERIMENT

4.1. Experiment setup

The systems are trained using the Adam optimizer, with a maximum learning rate of 0.001, and a learning rate ramp up during the first 50 epochs. Each system is trained for a total of 200 epochs. The batch size is set to 4, 4, 8 for strongly labeled data, weakly labeled data and unlabeled data respectively in a batch. In our experiments, we save the top5 models for PSDS1 and PSDS2 separately, which can be further used for model ensembling. Because each event class differs in duration length, we use the class-wise median filter. For pretrained models, we adopt the officially released pretrained model checkpoints that works best on AudioSet.

4.2. System without pretrained model

Table 1: Results of system without pretrained model

System	PSDS1	PSDS2
Baseline-MT	0.359	0.562
FDYCRNN-MT	0.412	0.637
FDYCRNN-MT + data augmentation	0.449	0.702
FDYCRNN-MMT	0.457	0.707

Table 1 shows the results of system without pretrained model on validation set. *Baseline-MT* and *FDYCRNN-MT* have the same training process, data augmentations and semi-supervised strategy. The only difference is that the models adopted are different, where Baseline uses CRNN. *FDYCRNN-MT + data augmentation* applies the data augmentations mentioned in 3. And *FDYCRNN-MMT* is trained with MMT framework.

4.3. System with pretrained model

Table 2: Results of system with pretrained model

System	Pretrained model 1	Pretrained model 2	PSDS1	PSDS2
Baseline-MT	-	BEATs	0.500	0.762
FDYCRNN-MMT	AST	BEATs	0.513	0.783
	PANNS	BEATs	0.518	0.790
	PASST	BEATs	0.502	0.774
	HTSAT	BEATs	0.506	0.795

Table 2 shows the results of system with pretrained model on validation set. When training the system with pretrained models, the external strongly labeled AudioSet data are also utilized. As can be seen from the Table 2, the system based on *FDYCRNN-MMT* are not much better than the baseline. We speculate that this is because the features extracted by BEATs is too powerful, so that during the training process, the model is easily overfitted to pretrained features, resulting in the CNN part of FDYCRNN didn't work.

4.4. System ensemble

Table 3: Results of ensembled system

System	Pretrained model	Weak prediction	PSDS1	PSDS2
FDYCRNN-MMT			0.475	0.721
		✓	0.065	0.815
	✓		0.535	0.806
	✓	✓	0.087	0.875

Table 3 shows the results of system ensembled with several models on validation set. *Pretrained model* means the features extracted by pretrained models are utilized, while the systems with different pretrained models, alignments and fusion strategies are ensembled.

5. CONCLUSION

In this report, we present our systems used in the task 4 of DCASE 2023 Challenge. We adopt mixup, frame shift, SpecAugment, FilterAugment, ICT and SCT for data augmentation. We apply MMT strategy on FDYCRNN model. Besides, we add external data and pretrained model to further improve performance.

6. REFERENCES

[1] J. Ebbers and R. Haeb-Umbach, "Pre-training and self-training for sound event detection in domestic environments," DCASE2022 Challenge, Tech. Rep., June 2022.

[2] J. W. Kim, G. W. Lee, H. K. Kim, Y. S. Seo, and I. H. Song, "Semi-supervised learning-based sound event detection using frequency-channel-wise selective kernel for dcase challenge 2022 task 4," DCASE2022 Challenge, Tech. Rep., June 2022.

[3] K. Li, X. Zheng, and Y. Song, "A two-stage training method for dcase 2022 challenge task4," DCASE2022 Challenge, Tech. Rep., June 2022.

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

[5] Y. Ge, D. Chen, and H. Li, "Mutual mean-teaching: Pseudo label refinery for unsupervised domain adaptation on person re-identification," in *International Conference on Learning Representations*, 2020. [Online]. Available: https://openreview.net/forum?id=rJlnOhVYPS

[6] H. Nam, S.-H. Kim, B.-Y. Ko, and Y.-H. Park, "Frequency Dynamic Convolution: Frequency-Adaptive Pattern Recognition for Sound Event Detection," in *Proc. Interspeech 2022*, 2022, pp. 2763–2767.

[7] Y. Gong, Y.-A. Chung, and J. Glass, "AST: Audio Spectrogram Transformer," in *Proc. Interspeech 2021*, 2021, pp. 571–575.

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

[9] K. Koutini, J. Schlüter, H. Eghbal-zadeh, and G. Widmer, "Efficient Training of Audio Transformers with Patchout," in *Proc. Interspeech 2022*, 2022, pp. 2753–2757.

[10] K. Chen, X. Du, B. Zhu, Z. Ma, T. Berg-Kirkpatrick, and S. Dubnov, "Hts-at: A hierarchical token-semantic audio transformer for sound classification and detection," in *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022, pp. 646–650.

[11] S. Chen, Y. Wu, C. Wang, S. Liu, D. Tompkins, Z. Chen, and F. Wei, "Beats: Audio pre-training with acoustic tokenizers," 2022.

[12] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition," in *Proc. Interspeech 2019*, 2019, pp. 2613–2617.

[13] H. Nam, S.-H. Kim, and Y.-H. Park, "Filteraugment: An acoustic environmental data augmentation method," in *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022, pp. 4308–4312.

[14] C.-Y. Koh, Y.-S. Chen, Y.-W. Liu, and M. R. Bai, "Sound event detection by consistency training and pseudo-labeling with feature-pyramid convolutional recurrent neural networks," in *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 376–380.