MULTI-TASK FRAME LEVEL SYSTEM FOR FEW-SHOT BIOACOUSTIC EVENT DETECTION

Technical Report

Genwei Yan[1], *Ruoyu Wang*[2], *Liang Zou*[1], *Jun Du*[2,*], *Qing Wang*[2], *Tian Gao*[3], *Xin Fang*[3]

[1] China University of Mining and Technology, XuZhou, China, {gwyan, liangzou}@cumt.edu.cn
[2] University of Science and Technology of China, HeFei, China,
wangruoyu@mail.ustc.edu.cn, {jundu, qingwang2}@ustc.edu.cn
[3] iFLYTEK Research Institute, HeFei, China, {tiangao5, xinfang}@iflytek.com

ABSTRACT

This technical report describes our new frame-level embedding learning system for DCASE2023 Task5: few-shot bioacoustic event detection. In the previous work, we proposed the frame-level embedding learning system and achieved the best performance of the DCASE 2022 Task5. In this work, we utilize several techniques to improve upon our previous work. Additionally, we introduce multi-task learning and Target Speaker Voice Activity Detection (TS-VAD) strategies to transform our previous system into a new multi-task frame-level embedding learning system. Compare to our previous work, our new system can achieve a better result (F-measure 75.74%, No ML) on the official validation set.

Index Terms— DCASE, few-shot bioacoustic event detection, embedding learning, model adaptation, event filtering

1. INTRODUCTION

Few-shot learning (FSL) [1] describes tasks in which an algorithm must make predictions given only a few instances of each class. Specifically, FSL is usually studied using *N-way-k-shot* classification, where N denotes the number of classes and k denotes the number of known examples for each class. Few-shot bioacoustic event detection (FSBED) is a task that focuses on sound event detection (SED) in few-shot learning (FSL). For the FSBED task, we need to extract information from 5 exemplars (i.e., $k = 5$) vocalizations (shots) of mammals or birds and detect and classify sounds in field recordings [2, 3]. Compared with general SED [4] tasks which need a large amount of labeled data in bioacoustic event detection (BED) tasks, FSBED only needs a few supervised samples to finish the BED tasks, which can greatly save labor costs.

In our previous work, we proposed a frame-level system and achieved the best performance in DCASE 2022 task 5, but there are still some shortcomings in our system. Firstly, due to the independence of frame-level data and the time continuity of events, the sample balance can not be performed as done for the segment-level system. Secondly, because of the scarcity of supervised samples, the model is prone to overfitting during fine-turning. Furthermore, through the experiments, we found that different distributions of the training set have a significant impact on the pre-training of the model. This time to solve or alleviate these problems, we propose several tricks. Firstly, we suggest using an event concatenation to build the sample balance. Specifically, 2 shots are randomly selected from the 5 shots, then fill the 67% of the sample window with the selected shots. Secondly, we adopt the FilterAugment [5] method to perform a linear Time/Frequency (TF) mask on the support samples to build a data augment in the fine-tuning stage. In addition, in our previous work, to achieve the best pre-trained model, we used the early stop method, which means, there will be some information that would be useful for pre-training that we can not put into training. To address this problem, we use the kfold to split the training set and select the best-performing model on the eval set.

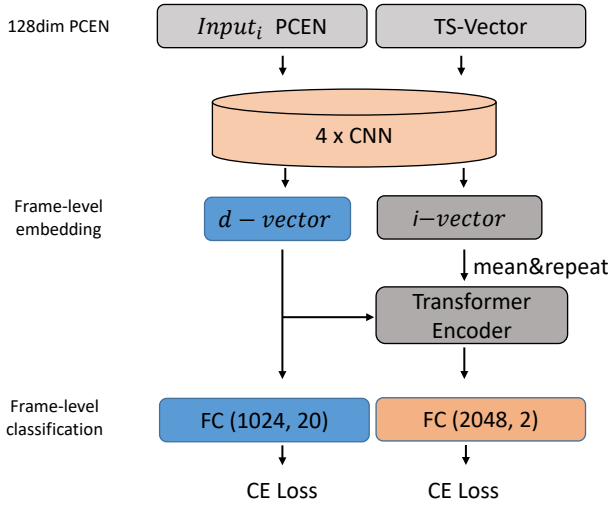
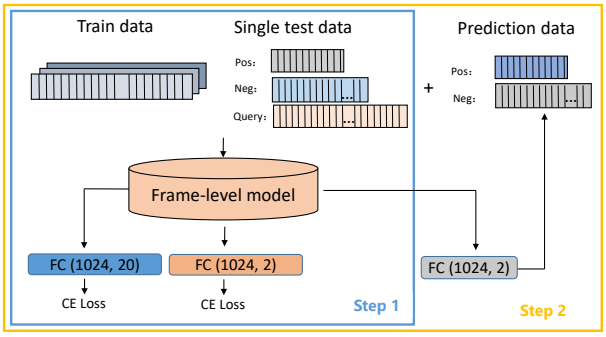


Figure 1: The training framework of the multi-task frame level model.

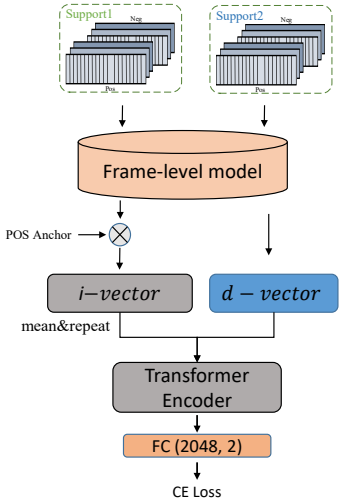
In our report [6] for the DCASE 2022 task 5, we proposed 3 different systems. Later, we attempted to merge the separation model with the frame-level model, but we found that the separation model's massive number of parameters made it difficult to train. Moreover, the noisy source signal prevented the separation model from obtaining the Ideal Binary Mask [7] (IBM). Inspired by Multi-task Learning [8] and Target Speaker Voice Activity Detection [9] (TS-VAD), we extract the separation module from the separation model and converted the IBM outputs into the TS-VAD outputs. Finally, we merge the TS-VAD branch with the original frame-level system to form a multi-task frame-level system.

2. METHODOLOGY

The training/fine-tuning framework is roughly shown in Figure 1 and Figure 2. In the training stage, we improved our previous frame-level classification model by incorporating an additional TS-VAD branch, which allows the network to learn from various perspectives and enhance its generalization performance. In the fine-tuning stage, we similarly add additional branches while keeping the frame-level bedding branch consistent with previous work. In the TS-VAD branch, to enable the TS-VAD branch to learn the distribution characteristics of the POS in support, we develop a strategy that involves applying two different arrangements for the POS segments in support. We use the arrangement that matches the POS event distribution in the audio as the TS-Vector to classify the other arrangement. This approach effectively leverages the support data and improves the performance of the TS-VAD branch. Compared with the original frame-level system (No ML, F1 70.02%), the multi-task frame-level system can achieve better results on the validation set(F1 75.74%). Detailed results are shown in Table 2.



(a) The frame-level embedding branch



(b) The TS-VAD branch

Figure 2: The fine-tuning framework of the multi-task frame level model.

Table 1: The network architecture of multi-task frame-level embedding model.

Block	*kernel_stride*	*BN_Activate*
CNN_Block1	conv,3 × 3, (1, 128)	BN+ReLU
CNN_Block2	conv,3 × 3, (128, 128)	BN+ReLU
CNN_Block3	conv,3 × 3, (128, 128)	BN+ReLU
CNN_Block4	conv,3 × 3, (128, 128)	BN+ReLU
TF_Encoder	2048 enc_dim, 1024 hidden, 8 heads	LN+ReLU
decoder1	fc(1024, 2)	Softmax
decoder2	fc(2048, 2)	Softmax

2.1. Multi-task Training Strategy

As shown in Figure1, our new frame-level network mainly adds an additional branch on the basis of the previous frame-level network. Therefore, in this part, we mainly introduce the training details of the TS-VAD. The configuration of the new frame-level network is shown in Table 1.

TS-Vector Building. For the TS-VAD branch in multi-task training, we provide a Ts-vector for each frame of the primary event class in the window sample. Specifically, when using sliding window to segment and sample the training set, we select the event that accounts for the largest number of the sample window as the primary class c. then we only keep the frames belonging to the c and erase the rest. It can be simulated as eq 3. The x_i denotes the 128-dimensional PCEN of a single frame.$y_i \in \{1, .., 19\}$ denotes the label of a single frame.

$$x_i = \begin{cases} x_i & if\ y_i = c \\ 0 & else \end{cases} \tag{1}$$

To avoid labeling the same event as the primary class c in adjacent sliding windows and when there are multiple event classes in the sliding window, we will use the principles of sliding window time shift and non-repeated sampling of the same event to choose the other event with the highest proportion in the sample window as the primary class c. Additionally, to prevent the model from gaining knowledge of the target position in advance during the training process, we choose the TS-Vector last recorded as the input of the TS-VAD branch to build a temporal mismatching between the TS-Vetor and the TS-VAD target. It is shown in Figure 3.

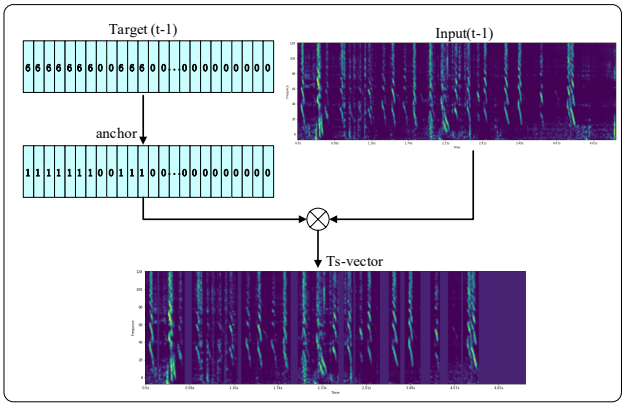


Figure 3: TS-Vector building.

Table 2: Detailed validation results of three systems

$System$	$Precision(\%)$	$Recall(\%)$	$F1(\%)$	$F1-HB(\%)$	$F1-ME(\%)$	$F1-PB(\%)$
Baseline	36.3	24.9	29.5	/	/	/
Frame-Level	73.0	67.6	70.2	77.0	90.0	53.7
Multi-Task Frame-Level	**76.2**	**75.3**	**75.7**	**86.7**	**90.2**	**58.9**

Transformer Encoder Considering that there is temporal information between frames in the sound event, and when the current event frame energy is weak, introducing additional temporal information can help supplement the information of the feature of this frame, we introduce one transformer encoder layer as the temporal module which is shown in Figure 4.

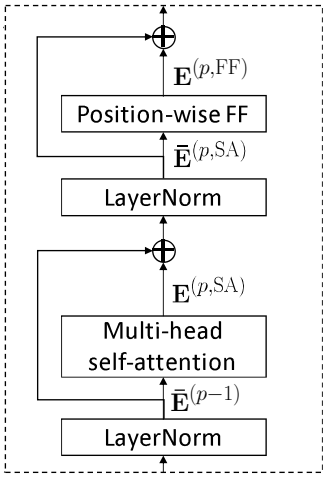


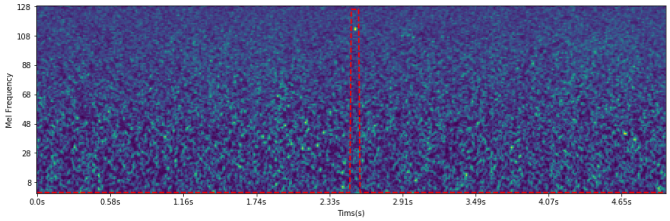
Figure 4: The framework of transformer encoder

2.2. Multi-task Fine-tuning Strategy

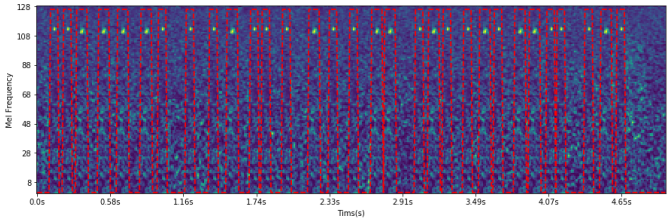
Similar to the training strategy our fine-tuning strategy also has two branches. In the fine-tuning stage, these two branches are fine-tuned together. The total loss of the sum of the loss of each branch. Due to the large framework of the system as a whole, we will describe the details of the two branches separately in this part.

Support set rebuilding Since we believe that the prediction of the event in the frame-level system is not highly dependent on the non-event class (neg), randomly combining the positions of neg and pos can have the effect of data augmentation. Therefore different from the baseline method that segments the whole support set by sliding window and labels 0-1 according to the position of the sample window, we use a random permutation combination to reconstruct the support set (Support 1). Specifically, we first extract the segment to which the 5 shots belong and the neg segment sandwiched in it according to the annotation information. After that, we randomly crop half length of the sample window in negs to fill the sample window and then randomly select 1 shot from the 5 shots into it, and finally fill the sample window with neg. This ensures that there is an event in each sample, and the event will be in the middle of the sample window, which is more realistic. As is shown in Figure 5(a)

However, this reconstruction method has a drawback, that is when the POS is very shot, such as PB, there will be a serious imbalance problem in each sample window. In order to perform sample balance, we reconstruct the second batch of the support set (Support 2). Specifically, we randomly select 2 shots from the 5 shots and carry out the tile operation respectively to make the total length of POS account for 67% of the sample window, then fill the POS uniformly in the sample window with the bottom of neg. As is shown in Figure 5(b)



(a) support 1



(b) support 2

Figure 5: The reconstruct support of BUK4_20171022_004304a.

Frame-level embedding fine-tuning branch As shown in Figure 2(a), our frame-level embedding network is consistent with previous work which adopts the two-step fine-tuning strategy. In the first step, we define the 2-classification task on the Support (Support 1 + Support 2). In addition, to regularize the model, we introduce the training set mixed with the POS to define the 20-classification task. In the second step, we leverage the pseudo-label output in the first stage to further fine-tune the model. Then the two-step is iterated until the set number of iterations.

TS-VAD embedding fine-tuning branch As shown in Figure 2(b). In the TS-VAD branch, support 1, whose event distribution is more in line with the actual situation, is fed into the model as TS-Vector. Then the corresponding is obtained as the i-vector according to the position of POS in the sample window. After getting the d-vector, we concatenate the i-vector with the d-vector in the channel dimension and feed it into the Transformer Encoder for self-attention encoding. After that, we use the decoder2 which has been trained in the training stage to decode the output of the Transformer Encoder and use the CE loss function to compute the TS-VAD loss. At the end of the fine-tuning iteration, we take the

whole support 1 as TS-Vector, and then feed the TS-Vector together with the query into the frame-level model to get the final prediction.

3. EXPERIMENTS

3.1. Data Preparation

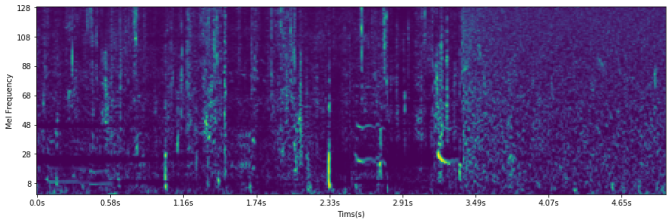
Similarly to the previous work, we use a sampling rate of 22050 HZ for audio sampling. For the multi-channel audios, we perform the mean function in the channel dimension to convert it into a single-channel signal. we perform Per-Channel Energy Normalisation (PCEN) on the Mel spectrum with 1024 n_fft and 256 hop_len to extract PCEN features. 431 window-length and 86 window-shift (11.6ms for each frame) are adopted for sliding window segmentation of PCEN to get a large receptive field, and each frame in the window is labeled according to the annotation csvs.

3.2. Data augmentation

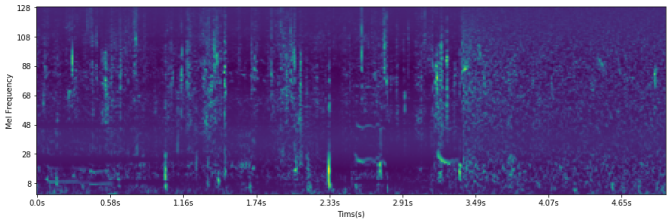
To improve the performance of the model, we used different data augmentation methods in both training and fine-tuning stage.

Training augmentation In the training set preparation, in order to increase the size of the training set, we use Kaldi tools to perturb the training audio with 0.9, 1.0 and 1.1 speed separately to change its timbre and intonation. In the training stage, to improve the generalization of the feature extractor, we apply time-frequency (TF) masking and random Gaussian noise to the PCEN features.

Fine-tuning augmentation Considering that the use of TF masking will lead to information loss which is harmful to the fine-tuning of the model. we use the FilterAugment to augment the PCEN features. Specifically, we randomly split $3 \sim 6$ bandwidths from the frequency of $40 \sim 128$ in PCEN and linearly mask the split bandwidths with $-6dB \sim 8dB$ scale rate. The augmented PCEN feature is shown in Figure 6. The scale rate Filters are shown in Figure 7.



(a) The PCEN feature



(b) The augmented PCEN feature

Figure 6: Illustrations on application of FilterAugment on an example of ME1 PCEN features

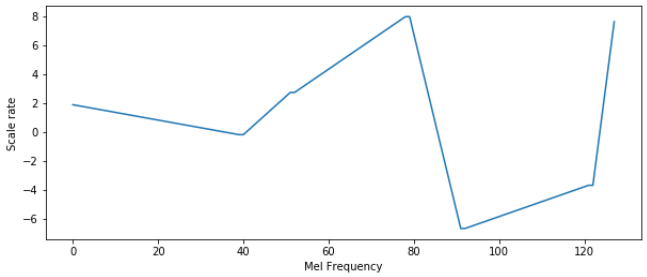


Figure 7: Filters applied on the PCEN features

4. REFERENCES

[1] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," *Advances in neural information processing systems*, vol. 30, 2017.

[2] V. Morfi, I. Nolasco, V. Lostanlen, S. Singh, A. Strandburg-Peshkin, L. F. Gill, H. Pamuła, D. Benvent, and D. Stowell, "Few-shot bioacoustic event detection: A new task at the dcase 2021 challenge," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2021 Workshop (DCASE2021)*, 2021.

[3] I. Nolasco, S. Singh, E. Vidana-Villa, E. Grout, J. Morford, M. Emmerson, F. Jensens, H. Whitehead, I. Kiskin, A. Strandburg-Peshkin, *et al.*, "Few-shot bioacoustic event detection at the dcase 2022 challenge," *arXiv preprint arXiv:2207.07911*, 2022.

[4] A. Mesaros, T. Heittola, T. Virtanen, and M. D. Plumbley, "Sound event detection: A tutorial," *IEEE Signal Processing Magazine*, vol. 38, no. 5, pp. 67–83, 2021.

[5] 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)*. IEEE, 2022, pp. 4308–4312.

[6] J. Tang, Z. Xueyang, T. Gao, D. Liu, X. Fang, J. Pan, Q. Wang, J. Du, K. Xu, and Q. Pan, "Few-shot embedding learning and event filtering for bioacoustic event detection technical report," DCASE2022 Challenge, Tech. Rep., June 2022.

[7] N. Li and P. C. Loizou, "Factors influencing intelligibility of ideal binary-masked speech: Implications for noise reduction," *The Journal of the Acoustical Society of America*, vol. 123, no. 3, pp. 1673–1682, 2008.

[8] J. Ma, Z. Zhao, X. Yi, J. Chen, L. Hong, and E. H. Chi, "Modeling task relationships in multi-task learning with multi-gate mixture-of-experts," in *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, 2018, pp. 1930–1939.

[9] Y. Fujita, N. Kanda, S. Horiguchi, Y. Xue, K. Nagamatsu, and S. Watanabe, "End-to-end neural speaker diarization with self-attention," in *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE, 2019, pp. 296–303.