FEW-SHOT BIOACOUSTIC EVENT DETECTION WITH FRAME-LEVEL EMBEDDING LEARNING SYSTEM

Technical Report

PengYuan Zhao[1], ChengWei Lu[1], Liang Zou[1],

[1] China University of Mining and Technology, XuZhou, China, {pengyuan, chengwei, liangzou }@cumt.edu.cn

ABSTRACT

This technical report presents our frame-level embedding learning system for the DCASE2024 challenge for few-shot bioacoustic event detection (Task 5). We used an open-source multi-task frame-level learning system. In this work, we used log-mel and PCEN for feature extraction of the input audio, Netmamba Encoder as the information interaction network, and adopted data augmentation strategies to improve the generalizability of the trained model as well as multiple post-processing methods. Our final system achieved an F-measure score of 56.4% in DCASE2024 task 5 Validation set.

Index Terms— Dcase, few-shot bioacoustic event detection, embedding learning

1. INTRODUCTION

Few-shot learning (FSL) [1] aims to train models that can accurately predict new data with a small amount of labeled data. Methods such as Meta-learning architectures and data augmentation techniques are commonly used to improve model generalization and increase the diversity of training data. This approach is particularly suitable for scenarios where data is scarce, costly to obtain, or difficult to label. Bioacoustic event detection (BED) field refers to the automated identification and classification of sounds produced by various organisms (such as birds, mammals, etc.) in nature. Deep learning and transfer learning are commonly used to enhance the accuracy and generalization of sound event detection and classification. This technology plays an important role in ecological monitoring and species conservation. However, annotating bioacoustic data usually requires significant time and effort from experts in the field. Few-shot bioacoustic event detection (FSBED) [2, 3] combines the advantages of few-shot learning, requiring only a small amount of supervised data to accomplish bioacoustic event detection, thereby saving significant time and labor costs. This technology has important applications in studying animal populations and their behaviors.

To improve model efficiency, we use the NetMamba Encoder [4] as the Information exchange network instead of the Transformer encoder. Mamba is a linear-time state space model for sequence modeling that has achieved significant success in various fields, including natural language processing [5] and computer vision [6]. Through innovative design, the Mamba network addresses many issues encountered by RNNs, LSTMs, and Transformers [7, 8] when processing long sequential signals, such as high computational complexity, large memory consumption, and gradient vanishing. It excels in capturing long-term dependencies, computational efficiency, and memory efficiency, providing a more efficient and stable solution for long sequential signal processing tasks. This

indicates that applying Mamba to the field of few-shot bioacoustic event detection has great potential. To achieve superior performance, inference efficiency, and few-shot learning capability, we chose NetMamba [9] among the various Mamba variants.

2. METHODOLOGY

The multi-task frame-level detection training model we use is shown in Figure 1. First, the input audio clip is processed and feature extraction is performed. The features are then input into the embedding extraction network. The frame-level embedding representation is fused and then used for sound event detection (SED) and foreground/background classification tasks. Finally, the embedding of the query set is multiplied by W and the prediction result is obtained through the softmax function. In the evaluation set, our method1 system base on logmel feature, which achieves an F1 score of 44.07%. The method2 system base on PCEN feature, which achieves an F1 score of 56.4%. Both systems perform well on validation sets such as HB and ME, but perform poorly on validation sets such as RD and PW.

2.1. Multi-task Training Strategy

The training framework, as depicted in Figure 1, employs a multi-task frame-level model. Initially, the input audio segments are processed to extract Log-mel features. These features, along with the masked Log-mel features, are fed into an embedding extraction network consisting of four CNN Blocks (a sequence of Convolutional Layer, Batch Normalization, and ReLU activation). In the Sound Event Detection (SED) branch, the generated embedding features are utilized. In the Foreground/ Background Classification (FBC) branch, the features from the SED branch are fused with those from the FBC branch and subsequently input into the NetMamba Encoder. Ultimately, we adopt Cross-Entropy (CE) loss in both the SED and FBC branches, denoted as L1 and L2 respectively. These losses are computed using the following formulas:

$$l_1 = -\frac{1}{N} \sum_{i=1}^{n} \sum_{j=1}^{c} ((Y_i = j) \odot M_i) \log(f_\theta(X_i)) \quad (1)$$

$$l_2 = \sum_{i=1}^{n} \sum_{j=0}^{1} ((A_i = j) \odot M_i) \log(f_\theta(X_i, X_t)) \log(f_\theta(X_i)) \quad (2)$$

$$l_{\text{total}} = l_1 + l_2 \quad (3)$$

State Space Models As the key components of Mamba, State Space Models (SSMs) represent a contemporary category of sequence models within deep learning that share broad connections
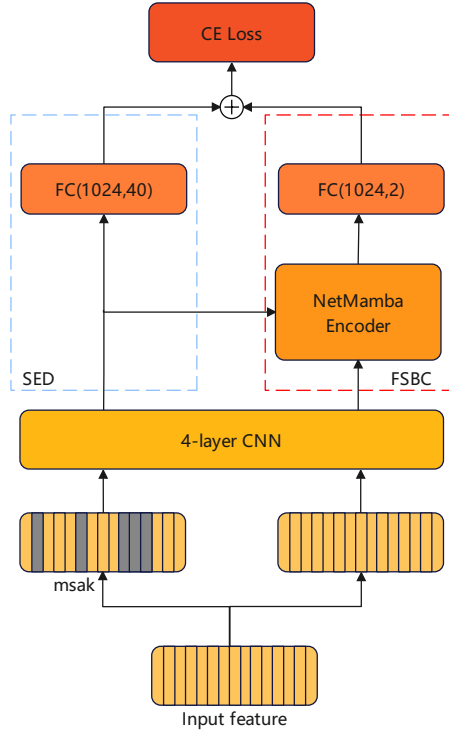
Figure 1: Training Framework

Figure 2: NetMamba Encoder

with Recurrent Neural Networks (RNNs) and Convolutional Neural Networks (CNNs). Drawing inspiration from continuous systems, SSMs are commonly structured as linear Ordinary Differential Equations (ODEs) which establish a mapping from an input sequence $x(t) \in \mathbb{R}^N$ to an output sequence $y(t) \in \mathbb{R}^N$ via an intermediate latent state $h(t) \in \mathbb{R}^N$:

$$\begin{aligned} h'(t) &= \mathbf{A}h(t) + \mathbf{B}x(t) \\ y(t) &= \mathbf{C}h(t) \end{aligned} \qquad (4)$$

where $\mathbf{A} \in \mathbb{R}^{N \times N}$ represents the evolution parameter, while $\mathbf{B} \in \mathbb{R}^{N \times 1}$ and $\mathbf{C} \in \mathbb{R}^{1 \times N}$ are the projection parameters.

NetMamba Encoder The NetMamba decoder use unidirectional Mamba blocks, as illustrated in Figure 2. For a given input token sequence X_{t-1} with a batch size of B and sequence length L from the $(t-1)$-th NetMamba block, the decoder first normalize it and then project it linearly into x and z, both with dimension size of E. It subsequently apply causal 1-D convolution to x, resulting in x'. Based on x', the decoder compute the input-dependent step size Δ, as well as the projection parameters B and C having a dimension size of N. It then discretize A and B using Δ. Following this, it calculate y employing a hardware-aware SSM. Finally, y is gated by z and added residually to X_{t-1}, resulting in the output token sequence X_t for the t-th NetMamba block.

2.2. Multi-task Fine-tuning Strategy

After investigating a large number of existing results, we found that the fine-tuning strategy is helpful in improving the detection accuracy and generalization of the model. So we also fine-tune the SED branch and the SFBC branch respectively on the reconstructed supports.
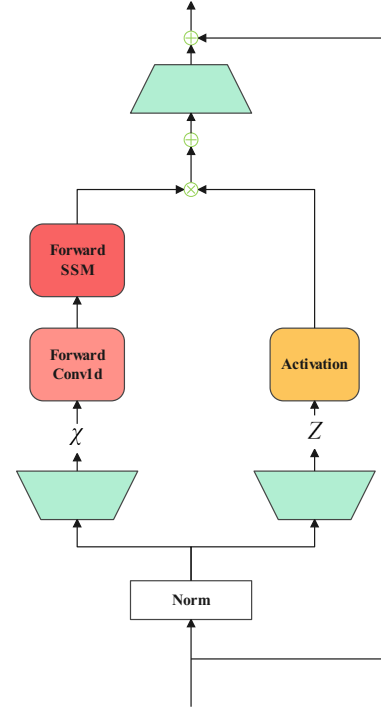
SED Branch Initially, we define a binary classifer based on the reconstructed supports set ($Support_1$ and $Support_2$). Subsequently, we employ the pseudo-labels generated in the initial stage to further refine the model. This two-step process is iterated for a predetermined number of cycles.

SFBC Branch We select the windows in $Support_1$ containing multiple different POS as TC-Vector and feed them into the lightweight feature extractor. We obtain the POS center by meaning the masked POS frames and utilize the NetMamba Encoder Layer transferred from the training stage to build a self-attention with the POS in $Support_2$.

3. EXPERIMENTS

3.1. Data Preparation

We use the official training dataset, and we use an FFT size of 1024, hop_len of 256, a Mel number of 128, and a sampling rate of 22.05kHz to calculate the Logmel spectrum features. A sliding window segmentation of Logmel is performed using a window length of 431 and a window shift of 86, and each frame in the window is labeled according to the annotation csv.

3.2. Data Augmentation

We observed that the quality and quantity of training data have a significant impact on the performance of the model, so we screened the training set. We observed that in the Western Mediterranean Birds (WAW) data set, there are many types of calls and long duration, but we observed that there are some labels are incorrectly labeled, so we first train the model on the training data that does not contain

Table 1: Dcase Task 5 Results

System	$Dcase2023Task5$			$Dcase2024Task5$		
	$Precision(\%)$	$Recall(\%)$	$F1-score(\%)$	$Precision(\%)$	$Recall(\%)$	$F1-score(\%)$
Baseline	36.34	24.96	29.59	56.18	48.64	52.14
Method1	76	67.24	71.35	35.85	57.19	44.07
Method2	75.68	69.39	72.4	61.16	52.32	56.4

WAW, and then use the trained model to predict the data with weak WAW annotation information to enhance the training data.

During the training phase, we added GussionNoise to the Logmel feature. At the same time, for the time domain and frequency domain of the feature, we used the FrequencyMasking object to randomly mask a certain number of frequency channels in the input data, and used the TimeMasking object to randomly mask the time frames in the input data to achieve data enhancement and improve the model's generalization ability for different audio features.

3.3. Post Processing

We extract the predicted probability, minimum detection length (MFL), threshold and average event length of the model evaluation results. In particular, for the adjustment of the threshold, a strategy is set: subtract 0.05 from the threshold of the model output. If the result is lower than 0.5, the threshold is set to 0.5. This can avoid false detections caused by too low threshold. The predicted probabilities are then binarized, setting the probability above the adjusted threshold as 1 and the rest as 0. A convolution operation is used to identify significant change points in the binarized probabilities, which represent the beginning and end of events. In order to further optimize event detection, we introduced the non-maximum suppression (NMS) [10] algorithm and set an IoU threshold of 0.7 to remove events that overlap or are close to each other and retain the best event detection results.

For frequently occurring short events, we have added a step of processing. If the average event length is greater than 2 times the minimum frame length (for example, 2 * 87 frames), and the interval between consecutive events is less than 87 frames, and the average predicted probability is greater than 0.5, we reset the predicted probabilities of these consecutive events to 1 to enhance the model's ability to detect such events. Finally, we implemented a temporal smoothing technique using a moving average with a window size of 5 frames to smooth the prediction probabilities and reduce noise. Additionally, we used median filtering to further reduce outliers in the predictions.

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] I. Nolasco, S. Singh, V. Morfi, V. Lostanlen, A. Strandburg-Peshkin, E. Vidaña-Vila, L. Gill, H. Pamuła, H. Whitehead, I. Kiskin, *et al.*, "Learning to detect an animal sound from five examples," *Ecological informatics*, vol. 77, p. 102258, 2023.

[3] J. Liang, I. Nolasco, B. Ghani, H. Phan, E. Benetos, and D. Stowell, "Mind the domain gap: a systematic analysis on bioacoustic sound event detection," *arXiv preprint arXiv:2403.18638*, 2024.

[4] A. Gu and T. Dao, "Mamba: Linear-time sequence modeling with selective state spaces," *arXiv preprint arXiv:2312.00752*, 2023.

[5] W. He, K. Han, Y. Tang, C. Wang, Y. Yang, T. Guo, and Y. Wang, "Densemamba: State space models with dense hidden connection for efficient large language models," *arXiv preprint arXiv:2403.00818*, 2024.

[6] L. Zhu, B. Liao, Q. Zhang, X. Wang, W. Liu, and X. Wang, "Vision mamba: Efficient visual representation learning with bidirectional state space model," *arXiv preprint arXiv:2401.09417*, 2024.

[7] J. Schäfer-Zimmermann, V. Demartsev, B. Averly, K. Dhanjal-Adams, M. Duteil, G. Gall, M. Faiß, L. Johnson-Ulrich, D. Stowell, M. Manser, *et al.*, "animal2vec and meerkat: A self-supervised transformer for rare-event raw audio input and a large-scale reference dataset for bioacoustics," *arXiv preprint arXiv:2406.01253*, 2024.

[8] Z. Liang, Y. Genwei, W. Ruoyu, D. Jun, L. Meng, G. Tian, and F. Xin, "Multitask frame-level learning for few-shot sound event detection," 2024.

[9] T. Wang, X. Xie, W. Wang, C. Wang, Y. Zhao, and Y. Cui, "Netmamba: Efficient network traffic classification via pre-training unidirectional mamba," *arXiv preprint arXiv:2405.11449*, 2024.

[10] A. Neubeck and L. V. Gool, "Efficient non-maximum suppression," in *18th international conference on pattern recognition (ICPR'06)*. IEEE, 2006.