FEW-SHOT BIOACOUSTIC EVENT DETECTION AT THE DCASE 2024 CHALLENGE

Wei Liu[1], Hy Liu[1], Fl Lin[1], Hs Liu[1], Tian Gao[1], Xin Fang[1], Jh Liu[1]

Xuyao Deng[2], Yanjie Sun[2], Kele Xu[2], Yong Dou[2]

[1] iFLYTEK Research Institute, HeFei, China, {weiliu87, hyliu41, fllin, hsliu5, tiangao5, xinfang, jhliu)@iflytek.com

[2] National University of Defense Technology, ChangSha, China, {dengxuyao, sunyanjie21, xukelele, yongdou}@nudt.edu.cn

ABSTRACT

In this technical report, we describe the submission system for DCASE2024 Task 5: Few-shot Bioacoustic Event Detection. In previous work, we proposed a frame-level embedding learning system and achieved the best performance in DCASE2022 Task 5. In this task, we propose several methods to improve the representational capacity of embeddings under limited positive samples. Three methods are proposed based on the pre-training fine-tuning process, including the AAPM segment-level embedding learning method, the Baseline framework-level embedding learning method, and the Unet network-based framework-level embedding learning method. Compared to our previous work, our new system achieved better results on the official 2023 validation set (F-measure 76.8%, No ML). The proposed system was evaluated on the newly released official 2024 validation set, with a best overall F-measure score of 70.56%.

Index Terms— DCASE2024, few-shot bioacoustic event detection, AAPM segment-level, Unet network

1. INTRODUCTION

Few-shot learning (FSL) [1] is a branch of machine learning that aims to develop effective models using very limited labeled data. Unlike traditional methods requiring extensive labeled datasets, FSL seeks to generalize well with just a few training samples per class (often just a few images or data points). This is typically explored through N-way-k-shot classification, where N denotes the number of classes and k denotes the number of examples per class.

Few-shot bioacoustic event detection (FSBED) is a relatively new research area focusing on utilizing limited vocalization data of animals (mammals and birds) to perform sound event detection (SED) through few-shot learning (FSL) [2,3]. Essentially, FSBED can be seen as a few-shot image classification (FSIC) task, where a large query set for each audio is accessible. Researchers are improving detection performance in few-shot scenarios using methods like meta-learning, contrastive learning, and transfer learning.

Meta-learning is a key method in few-shot learning, training models to quickly adapt to new tasks. Model-Agnostic Meta-Learning (MAML) [4] initializes model parameters for rapid fine-tuning with limited data. ProtoNet [1] classifies by learning prototypes in the feature space. Sabiron et al. used ProtoNet for bat sound event detection, achieving notable results. Contrastive learning enhances feature representation through similar and dissimilar sample pairs. SimCLR [5] uses contrastive loss, generating positive sample pairs via data augmentation and contrasting with random negatives for robust feature learning. Zakszeski et al. studied SimCLR for animal sound event detection. Transfer learning involves transferring knowledge pre-trained on large-scale datasets to few-shot tasks. By leveraging pre-trained models (typically trained on large datasets) to extract features, transfer learning requires less data for the target task. For few-shot event detection, this means achieving good performance even with limited data. Therefore, based on the pre-training-fine-tuning process, this paper proposes three methods. First, the Baseline framework-level embedding learning method. Second, the Unet network-based framework-level embedding learning method. Third, the AAPM segment-level embedding learning method.

2. METHODOLOGY

Our detection process is roughly as shown in Figure 1. According to the positive label and the specific negative segments selection method, each audio in the test set is divided into positive segments, negative segments, and query sets. The LOGMEL features are performed on the spectrograms of audio segments. Hereafter, the above LOGMEL are input into embedding extraction network to obtain the segment-level (or frame-level for frame-level method) embedding representation. Then the prototype features are obtained by taking embdding mean of "postive" and "negative". The positive and negative central embedding are spliced to form a 1024*2 feature vector, which is used as an initialization parameter of the softmax binary classifier, that is $W \in R^{2\times d}$. Finally, the embedding of query set is multiplied by W, and the prediction results is obtained through a softmax function. Finally, the F1 results of our four systems are 66.66%, 70.56%, 61.69%, and 63.23%, respectively. Detail results are shown in Table 1.

Table 1:Detailed validation results of four systems

System	Precision (%)	Recall (%)	F1 (%)	F1_PB (%)	F1_HB (%)	F1_ME (%)	F1_pw (%)	F1_RD (%)
Unet-Frame-Level	64.78	68.65	66.66	65.29	86.06	94.34	51.48	55.40
logmelBase-Frame-Level	76.00	65.70	70.56	67.00	79.00	91.00	67.00	56.00
pcenBase-Frame-Level	74.27	52.76	61.69	62.88	79.68	90.19	59.22	40.62
AAPM-Seg-Level	66.24	60.49	63.23	42.6	88.00	83.64	71.08	54.92

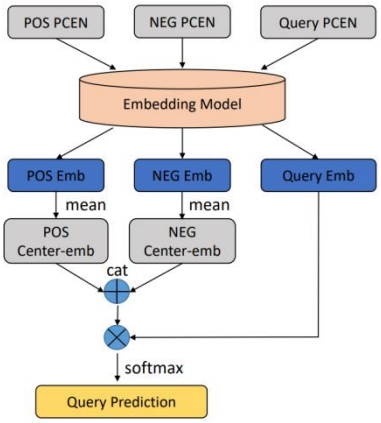

Figure 1: The system framework of few-shot bioacoustic event detection.

2.1. The Baseline framework-level embedding learning method

Figures 2 and 3 show the training and testing frameworks of the embedding learning system based on the Baseline framework. The frame-level method leverages the similarity between adjacent frames. Since variable-length audio can extract stable feature embeddings, we found that the frame-level framework achieves better adaptability during fine-tuning. Additionally, a two-step fine-tuning scheme was designed for the testing phase, allowing the use of both the training set and testing set to obtain improved feature embeddings.

The embedding system based on the Baseline framework has its network structure illustrated in Figure 2. It consists of 2 BasicBlock layers, 2 CNN layers, and 1 linear layer, with detailed configurations shown in Table 2. The BasicBlock is the core building unit in ResNet [6] (Residual Network) and has notable advantages over traditional convolutional layers. It incorporates a skip connection that directly adds the input to the output. This direct path allows gradients to bypass several layers during backpropagation, significantly mitigating the vanishing gradient problem and enabling effective training of deeper networks. The skip connection also helps the network converge faster by reducing the training loss and provides a regularization effect that reduces overfitting and enhances the model's generalization ability. Log-Mel and PCEN (Per-Channel Energy Normalization) are two common feature extraction methods in audio signal processing. Log-Mel is effective in areas like speech recognition and music information retrieval, being simple, easy to use, and computationally efficient. PCEN is more suited for tasks requiring high robustness, such as environmental sound detection and classification, performing particularly well in noisy environments. We extract Log-Mel and PCEN features from 128-bin Mel spectrograms, using 1024 FFT samples and a hop size of 256 samples. In the training phase, we use a simple CE loss function rather than few-shot loss.

Table 2:The framework-level embedding learning method network architecture based on the Baseline network

Block	kernel_stride	Activate
BasicBlock1	Conv, 3×3, (1,64)	BN+ReLU
BasicBlock2	Conv, 3×3, (64,64)	BN+ReLU
CNN_Block3	Conv, 3×3,(64,64)	BN+ReLU
CNN_Block4	Conv, 3×3,(64,64)	BN+ReLU
FC	Fc(1024,20)	softmax
Decoder2	Fc(1024,2)	softmax

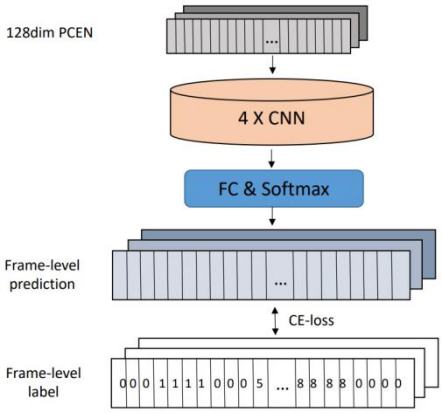

Figure 2: The framework of frame level model.

Two-step adaptive strategy. As shown in Figure 3, there are two-steps during fine-tuning. First, the 5-shot labeled support segments are selected as positive set, and the four segments between two positive samples are selected as negative set. We use cross entropy loss function to distinguish the two classes. In order to obtain better feature representation, we combine training set and the positive samples to define a 20-classification task.The second-step, we can get posterior probability of query set by the first-step model.Through method of fixed threshold selection, we set a high threshold to filter query results with high confidence into the positive set, thereby increasing the number of positive examples for training. Then repeat step1 and step2 until a set number of iterations. With the frame-level embedding learning, we got a powerful result in the development set as table 1, which F1-score is 70.56%.

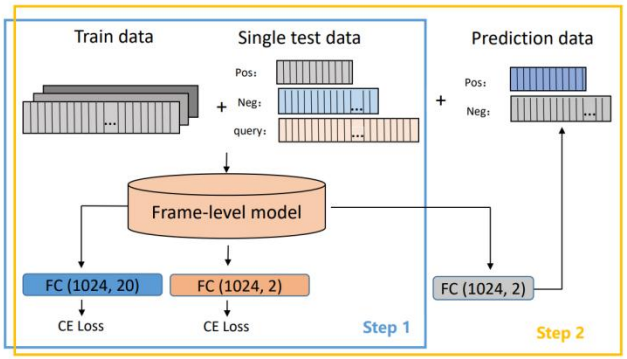

Figure 3: The framework of semi-supervised learning.

2.2. The Unet network-based framework-level embedding learning method

U-Net [7] is a Convolutional Neural Network (CNN) architecture mainly used for semantic segmentation. It features a symmetric encoder-decoder structure, where the encoder reduces the input image's spatial dimensions while increasing feature channels, and the decoder upsamples these feature maps to generate a segmentation mask. This architecture allows the extraction of both low-level details and high-level semantics, which is vital for few-shot learning. In such cases, making the most of limited data is essential for accurate classification or detection. U-Net's skip connections enable direct transfer of low-level features to the decoder, improving information flow during training and addressing the vanishing gradient issue, which is particularly important in few-shot learning.

Analysis of the validation and test sets revealed that the duration of animal sounds varies significantly. Some animals vocalize infrequently (e.g., BP, BP24), while others do so frequently (e.g., pw). To extract both low-level detail features and high-level semantic information, we modified the U-Net network for feature extraction, using it as the backbone for the framework-level embedding learning method. The structure of this method, based on the U-Net network, is detailed in Table 2.

Table 3:The framework-level embedding learning method network architecture based on the U-Net network

Block	kernel_stride	Activate
Inc	Conv, 3×3, (1,32)	BN+ReLU
Down1	Conv, 3×3, (32,32)	BN+ReLU
Down2	Conv, 3×3,(32,32)	BN+ReLU
Down3	Conv, 3×3,(32,64)	BN+ReLU
Down4	Conv, 3×3,(64,64)	BN+ReLU
Up1	Conv, 3×3,(64,64)	BN+ReLU
Up2	Conv, 3×3,(64,32)	BN+ReLU
Up3	Conv, 3×3,(32,32)	BN+ReLU
Up4	Conv, 3×3,(32,32)	BN+ReLU
FC	Fc(1024,20)	softmax
Decoder2	Fc(1024,2)	softmax

2.3. The AAPM segment-level embedding learning method

Inspired by BirdNET [8] and SSAST [9], we pretrained a comprehensive animal acoustic pretraining model(AAPM) on four Titan XP graphics cards using animal audio data within the allowable range of the rules. In the pretraining process, we followed the pretraining framework of SSAST and adopted the ViT model structure as the core. It is worth noting that we did not directly use the pretraining model of SSAST on audioset and librispeech datasets, but made full use of the pretraining method of SSAST to carry out a new pretraining on the regular animal audio datasets.

Unlike BirdNET and other CNN models that focus on the audio recognition of a certain kind of animals, our model uses a transformer based pretraining method to build a comprehensive model containing the acoustic characteristics of a variety of animal categories from AudioSet [10] and Xeno-Canto [11]. These animal categories include terrestrial animals (such as dogs, cats, pigs, crickets, etc.), aquatic animals (such as whales, dolphins, etc.), flying animals (such as birds, bats, etc.) and amphibians (such as frogs, etc.).

In order to ensure the generalization ability and robustness of the model, in addition to using rich animal audio data, we also introduced a variety of background environment sounds from AudioSet. These background sounds not only enrich the training data of the model, but also make the model better adapt to various complex acoustic environments in practical applications.

Through this series of pretraining and optimization, we have constructed a powerful and adaptive animal acoustic pretraining model, which provides a solid foundation for the subsequent task of few-shot bioacoustic event detection.

2.3.1. Animal Acoustic Dataset

In order to support the pretraining of animal acoustic models, we carefully constructed a training dataset containing a wide range of animal audio. This data set integrates the rich animal audio recorded by Xeno-Canto so far, as well as the animal audio in AudioSet and diversified background environment sounds. In order to ensure the practical application ability of the model, we used animal and background audio in TUT2016 [12]. and ESC50 [13] datasets as validation sets to test our pretrained animal acoustic model.

In the processing of Xeno-Canto dataset, we use BIRB [14] processing strategy, but considering the compatibility with AudioSet dataset, we adopt a fixed 10 second audio length. In addition, we adhere to high-quality data standards, so we filter out data below quality level B. In view of the weak label characteristics of Xeno-Canto dataset, we further assume that the longer the audio time, the lower the label quality and the higher the proportion of non event audio. Therefore, we eliminate the audio with a duration of more than 2 minutes to avoid adverse effects on model training. For audio with a duration of less than 10 seconds, we use the cycle filling technology to ensure that all audio samples reach the standard length of 10 seconds.

When using SSAST framework for pretraining, we followed the SSAST approach in processing the original data. Specifically, we retain the sampling rate of the original audio, and use the frame length of 25 milliseconds, the frame shift of 10 milliseconds and the number of 128 Mel bands to extract fbank features. This process aims to capture the key acoustic information in audio and provide high-quality feature input for subsequent model training.

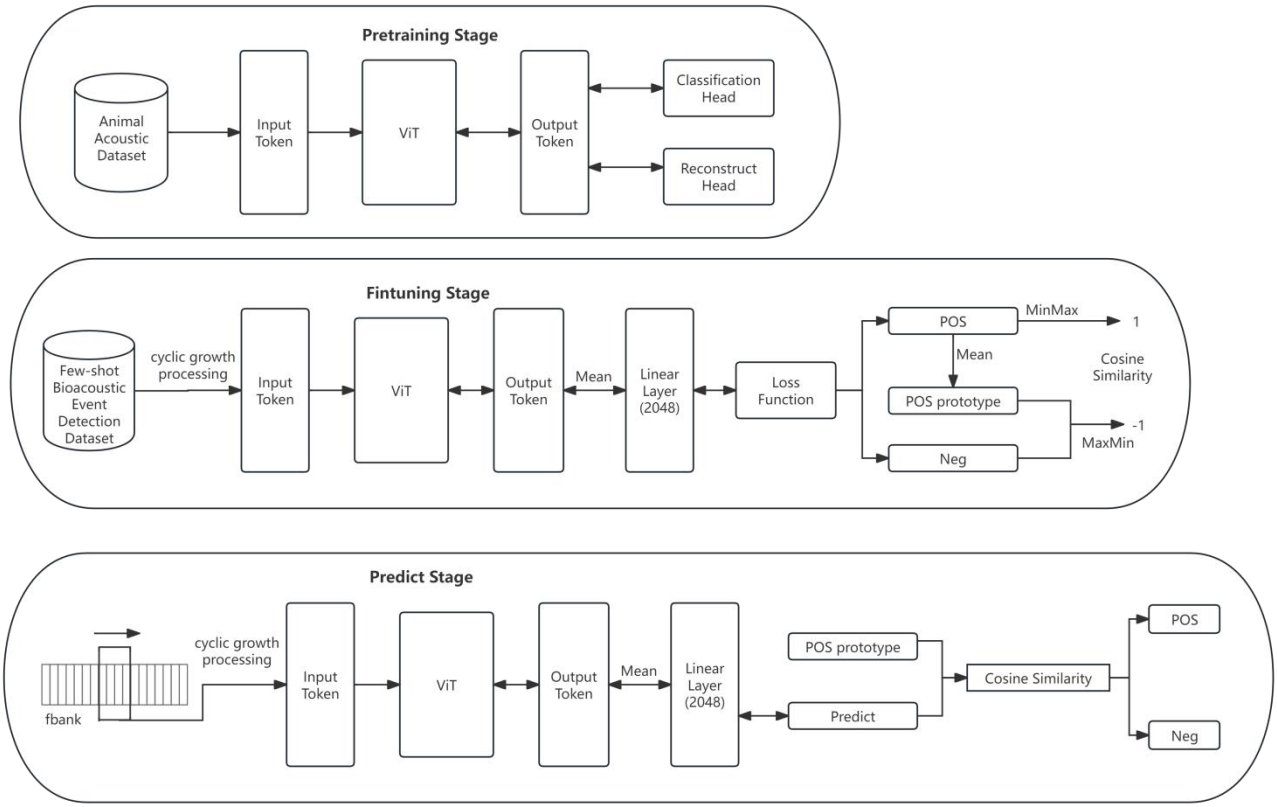

Figure 4: The framework of the whole system.

2.3.2. Animual Acoustic Pre-trained Model Finetuning

In order to adapt the pretrained animal acoustic model to the new field, we use the first five positive cases and the negative cases between the positive cases to finetune the animal acoustic pretraining model, so that the model can predict more accurately.
In the task of animal acoustic audio detection, in order to maintain the consistency with the pretraining stage, we followed the parameter setting of pretraining in the audio framing, that is, using the frame length of 25ms and the frame shift of 10ms to extract the fbank feature of 128 Mel bands, and retain the sampling rate of the original audio. However, due to the short time length of some animal sound events, there is a significant difference from the 10 second segment used in the pretraining, which may lead to the decline of model performance. In order to solve this problem, we performed cyclic growth processing on short events in the fine-tuning stage to enhance the detection ability of the model.

We implement the task of animal acoustic audio detection by classifying the frames. The framework of the whole system is shown in Figure 4, which clearly shows the process from feature extraction to classification decision.

Through experimental verification, we found that animal acoustic pretraining model is not suitable for single frame classification detection for all data set categories in few-shot animal sound detection task. Therefore, when using animal acoustic pretraining model for downstream fine-tuning classification, in order to ensure the robustness of the results, we adopted different frame number settings for different datasets, as shown in Table 4. This adaptive adjustment strategy effectively improves the detection performance of the model on different datasets

Table 4: Adaptive adjustment strategy of frame number

tmin(s)	[0,0.1]	(0.1,0.2]	(0.2,0.4]	(0.4,0.8]	(0.8,+inf)
frame num	2	10	20	40	80

2.3.3. Loss Function

In animal acoustic audio detection tasks, positive cases (i.e. target sounds) account for a relatively small proportion in the dataset and usually belong to a single type, while negative cases (i.e. background sounds or other non target sounds) account for the majority and contain a variety of possible sounds. Considering that the positive cases should have high similarity in acoustic characteristics, while the positive cases and negative cases should have significant differences, we designed a loss function based on cosine similarity to guide the training of the model.

Specifically, we define the cosine similarity function as $f(x, y)$, which is used to measure the cosine value of the angle between two vectors, so as to reflect the similarity between them. For positive case sets, because positive cases account for a

relatively small proportion in the dataset and usually belong to a single type, we expect that after model feature extraction, the feature vectors of any two positive cases have high similarity. Note that the set of positive cases is $\{p_i\}, i = 1 \ldots N_p$, where N_p is the number of positive cases, and the feature extraction process of the model for positive cases is g (P_i). We set the goal that is close to 1 for any i, $j (i \neq j)$.

When calculating the loss in each iteration, we focus on the two most dissimilar positive examples at present, that is, calculate the cosine similarity between all positive example pairs and take the minimum value. In this way, we get the loss function of the positive example part as:

$$loss_p = 1 - \min_{i,j \in \{1,\ldots,N_p\}, i \neq j} f(g(p_i), g(p_j)) \quad (1)$$

For negative examples, because they contain a variety of possible sounds and a large number, we do not make specific requirements for the similarity between negative examples. However, we expect the minimum similarity between negative cases and positive cases. For this purpose, we calculate the cosine similarity between the average eigenvector $p = \sum g(p^i) / N_p \quad i = 1 \ldots N_p$ of the positive case set and each negative case in the negative case set $\{n_i\}\, i = 1 \ldots N_n$ (where N_n is the number of negative cases). In each iteration, we focus on the negative case that is most similar to the positive case, that is, calculate the cosine similarity between all negative cases and P and take the maximum value. The loss function between negative and positive cases is

$$loss_n = 1 + \max_{i \in \{1,\ldots,N_n\}} f(g(n_i), P) \quad (2)$$

Finally, our total loss function is the sum of positive case loss and negative case loss:

$$loss = loss_p + loss_n \quad (3)$$

By optimizing this loss function, we can make the model better distinguish positive cases and negative cases in the feature space, so as to improve the accuracy of few-shot acoustic audio detection.

We evaluated our method on the validation set in 2024, and the specific results are shown in Table 1. It is worth emphasizing that our method does not use the training set data of 2024, but directly processes each audio file to be predicted separately. Specifically, we directly use the first five events of each audio file and its background-information, use the cosine similarity above tofinetune the model, and then predict the subsequent audio. In addition, our method does not adopt the strategy of conductive learning, which further highlights the effectiveness of our method.

3. EXPERIMENTS

3.1. Experimental setup

We use the Adam[15] optimizer for 20-class pre-training on the training data with a learning rate of 0.0003. The learning rate is decayed using StepLR with gamma=0.5 and a step-size of 10. The network is trained on 80% of the randomly split training data and validated on the remaining 20%. Training continues until there is no reduction in validation loss over the last 10 epochs, and the model with the highest accuracy is selected as the best model. During fine-tuning, only the last two convolutional layers and the FC layer are adjusted, with learning rates of 0.0001 and 0.001.

3.2. Data augmentation

Mixup: Mixup [16] is a data augmentation technique that interpolates the inputs and targets of two audio clips in the dataset. For instance, if the inputs of two audio clips are represented as x1 and x2, and their targets as y1 and y2, the augmented inputs and targets are calculated as x = λx1 + (1 − λ)x2 and y = λy1 + (1 − λ)y2, where λ is sampled from a Beta distribution[16]. By default, mixup is applied to frame-level feature maps.

SpecAugment [17] is a data augmentation technique specifically designed for speech data, particularly used for training Automatic Speech Recognition (ASR) systems. The main goal of SpecAugment is to improve the robustness and performance of ASR models by augmenting the spectrogram of the audio signal. SpecAugment operates on frame-level features using frequency masking and time masking. Frequency masking involves masking f consecutive mel-frequency bins [f0; f0 + f], where f is chosen from a uniform distribution from 0 to a frequency mask parameter F0, and F0 is from [0; F−F], where F is the number of mel-frequency bins. Time masking is similar to frequency masking but is applied in the time domain.

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. StrandburgPeshkin, 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] Finn. C, Abbeel. P, and Levine. S. “Model-agnostic meta-learning for fast adaptation of deep networks,” International conference on machine learning. PMLR, 2017.

[5] Chen. T, Kornblith. S, Norouzi. M, et al. “A simple framework for contrastive learning of visual representations,” International conference on machine learning. PMLR, 2020.

[6] He. K, Zhang. X, Ren. S, et al. “Deep residual learning for image recognition,” Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

[7] Ronneberger. O, Fischer. P, and Brox. T. “U-net: Convolutional networks for biomedical image segmentation,” Medical image computing and computer-assisted intervention−MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18. Springer International Publishing, 2015.

[8] Kahl. S, Wood. C M, Eibl. M, et al. “BirdNET: A deep learning solution for avian diversity monitoring,” Ecological Informatics 61 (2021): 101236.

[9] Gong. Y, Lai. C I, Chung. Y A, et al. “Ssast: Self-supervised audio spectrogram transformer,” Proceedings of the AAAI Conference on Artificial Intelligence. Vol. 36. No. 10. 2022.
[10] Gemmeke. J F, Ellis. D P W, Freedman. D, et al. “Audio set: An ontology and human-labeled dataset for audio events,” 2017 IEEE international conference on acoustics, speech and signal processing (ICASSP). IEEE, 2017.
[11] Vellinga. W P, and Planqué. R. “The Xeno-canto Collection and its Relation to Sound Recognition and Classification,” CLEF (Working Notes). 2015.
[12] Mesaros. A, Heittola. T, and Virtanen. T. “TUT database for acoustic scene classification and sound event detection,” 2016 24th European Signal Processing Conference (EUSIPCO). IEEE, 2016.
[13] Kumar. A, Khadkevich. M, and Fügen. C. “Knowledge transfer from weakly labeled audio using convolutional neural network for sound events and scenes,” 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2018.
[14] Hamer. J, et al. “BIRB: A Generalization Benchmark for Information Retrieval in Bioacoustics,” arXiv preprint arXiv:2312.07439 (2023).
[15] Kingma. D P, and Ba. J. “Adam: A method for stochastic optimization.” arXiv preprint arXiv:1412.6980 (2014).
[16] Zhang. H, Cisse. M, et al. “mixup: Beyond empirical risk minimization,” arXiv preprint arXiv:1710.09412 (2017).
[17] Park. D S, Chan. W, Zhang. Y, et al. “Specaugment: A simple data augmentation method for automatic speech recognition,” arXiv preprint arXiv:1904.08779 (2019).