Weakly Supervised Sound Event Detection

Project of AI3611 Intelligent Perception and Cognition Practice, 2024 Spring, SJTU

Xiangyuan Xue (521030910387)

School of Electronic Information and Electrical Engineering

Abstract

Weakly supervised sound event detection is a challenging task in the field of audio signal processing, which requires the model to learn sound events detection with only weak labels. In this project, we reveal the challenges of the unsupervised setting and implement a mainstream framework based on CRNN to solve the problem. We conduct experiments to research how the hyperparameters affect the performance of the model. We also investigate various data augmentation methods and conduct ablation studies to verify the effectiveness of these methods.

1 Introduction

Sound event detection (SED) [8] is a fundamental task in the field of audio signal processing, which aims to recognize what is happening in an audio clip and when it is happening. To be specific, given an audio clip, the goal is to recognize at what temporal instances different sounds are active, assigning each sound event a label along with an onset and offset time, which is illustrated in Figure 1. SED can be used in various real-world applications, such as surveillance, wildlife monitoring, and human-computer interaction. Different from the traditional sound classification task, multiple different sound events may occur simultaneously in a single audio clip. The localization of sound events also increases the difficulty of the SED task.

Figure 1: An overview of the sound event detection task. The input is an audio clip, and the output is a sequence of sound events with their labels and temporal locations.

There are two main approaches to train an effective SED model: fully supervised learning and weakly supervised learning. Fully supervised SED potentially performs better than weakly supervised SED [2]. However, it requires fine-grained manual labeling of sound events, which is extremely expensive since annotating the onset and offset time of each sound event in a long audio clip is time-consuming and labor-intensive [8]. In contrast, weakly supervised SED only has access to clip-level labels as is shown in Figure 2, yet still aims to localize sound events in the audio clip. Such approach reduces the requirement for the dataset while adding additional challenges to the model.

In this project, we will focus on the approach of weakly supervised SED. As is described in the previous paragraphs, the main challenge of weakly supervised SED comes from the lack of strong

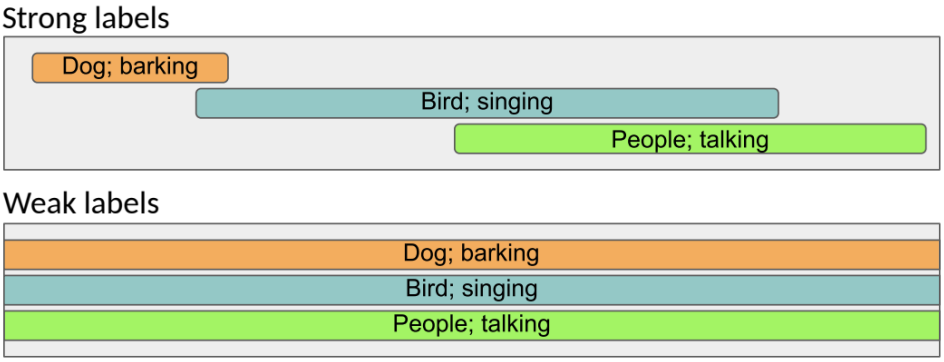

Figure 2: Different types of labels in sound event detection. Fully supervised SED requires fine-grained annotations of sound events. Weakly supervised SED only has access to clip-level labels.

labels. The model has no access to the fine-grained temporal labels but is expected to learn how to accurately localize the sound events [8]. Therefore, the model has to learn from the clip-level labels and generalize to the frame-level predictions. To address this problem, we introduce a weakly supervised framework, covering the feature extraction, model architecture, evaluation metrics, and data augmentation. In Section 2, we will introduce the pipeline of the weakly supervised SED model. In Section 3, we will present the experimental results and analyze the model performance. Finally, we will summarize our findings and provide some insights in Section 4.

2 Method

In this section, we will describe our pipeline from four main aspects: feature extraction, model architecture, evaluation metrics, and data augmentation. The feature extraction part is responsible for transforming the raw audio data into a format that can be fed into the model. Then the model will learn the feature representation and predict the sound events. Multiple evaluation metrics are introduced to comprehensively evaluate the model performance. We also try different data augmentation techniques to improve the robustness of the model. The overall pipeline is presented in Figure 3.

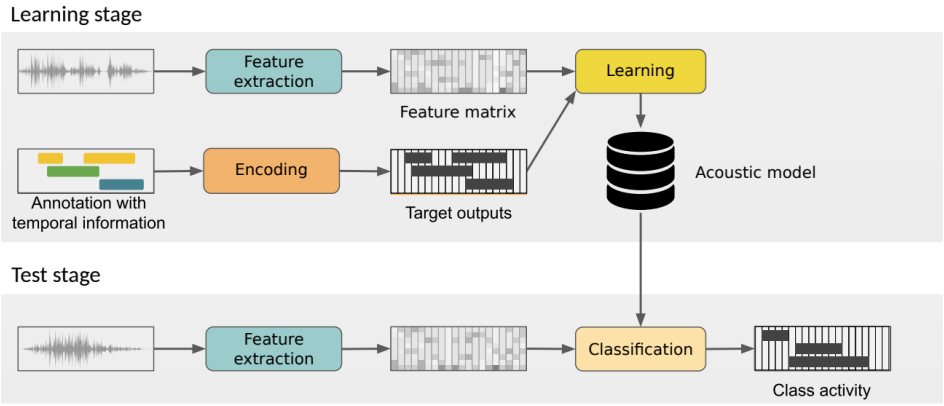

Figure 3: The overall pipeline of our framework. The raw audio data is first transformed into the log mel spectrogram. Then the model predicts the sound events based on the feature representation. Annotations are encoded into clip-level labels to supervise the model training.

2.1 Feature Extraction

To transform the raw audio data into learnable features, we apply the log mel spectrogram (LMS) as the input feature [11, 9]. LMS is a widely used feature representation in the field of audio signal processing. To compute the LMS, the audio clip is first divided into frames with a fixed length

and overlap. Then we apply the short-time Fourier transform (STFT) to each frame to obtain the spectrogram. Finally, we apply the mel filterbank to the spectrogram and take the logarithm to get the LMS. The computed LMS is a matrix with two dimensions, which is capable of representing the audio content in both time and frequency domains. An LMS with waveform is shown in Figure 4.

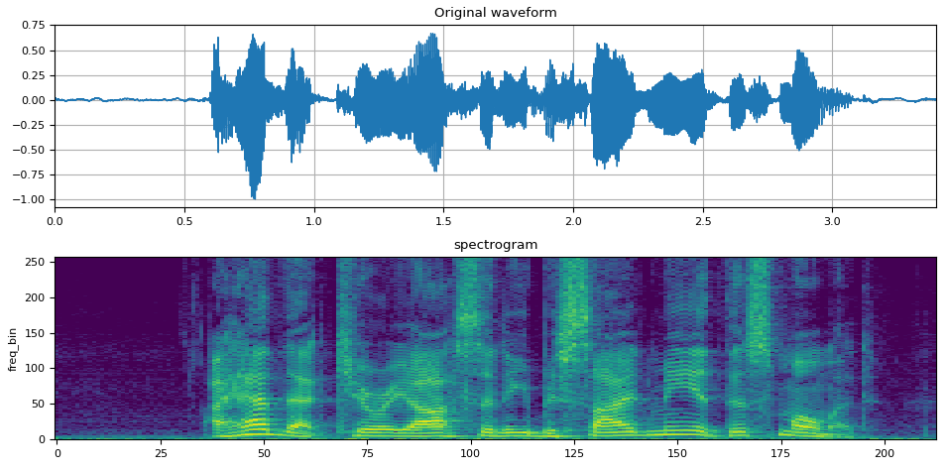

Figure 4: An illustration of the waveform of an audio clip and its corresponding log mel spectrogram. The two axes represent time and frequency respectively. The color indicates the intensity.

2.2 Model Architecture

Previous works [1, 3] have shown that convolutional recurrent neural networks (CRNN) can achieve state-of-the-art performance in sound event detection tasks. In this project, we adopt a similar architecture, which consists of multiple convolutional layers followed by a recurrent layer. The model is first fed with the LMS of the audio clip, which can be regarded as an image with only one channel. The convolutional layers extract the local features and produce a feature map. Then an average pooling layer is applied to aggregate the features along the frequency axis, which turns the feature map into a typical sequence. Finally, the recurrent layer processes the sequence and the fully connected layer together with a sigmoid function outputs the frame-level prediction.

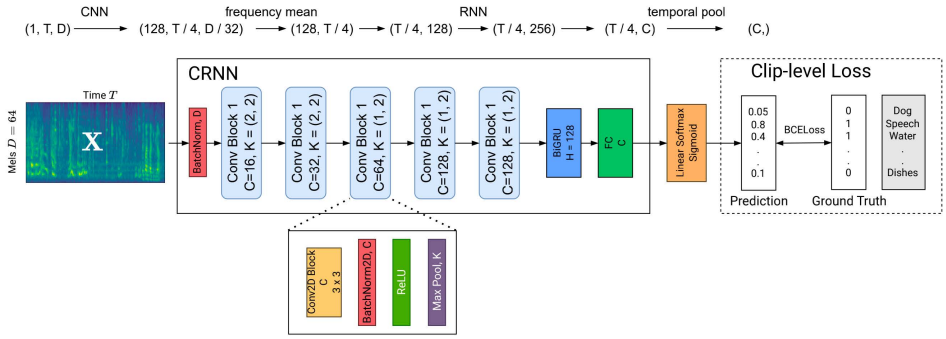

Figure 5: An overview of the model architecture. The model follows the CRNN structure, which consists of convolutional layers, recurrent layers, and fully connected layers. The output of the model is the frame-level prediction, which will be further processed to form the clip-level prediction.

However, the frame-level labels are not available in the weakly supervised setting, so the frame-level prediction will be fed into a linear softmax layer to form the clip-level prediction. The linear softmax layer can be interpreted as a self-weighted average function, which is formulated as

$$y(e) = \frac{\sum_{t=1}^{T} y_t^2(e)}{\sum_{t=1}^{T} y_t(e)} \tag{1}$$

Then the clip-level prediction can be aligned with the clip-level labels to supervise the model training. We use binary cross entropy (BCE) as the loss function, which is defined as

$$\mathcal{L}(y, \hat{y}) = -\hat{y}\log(y) - (1-\hat{y})\log(1-y) \tag{2}$$

For better frame-level prediction, we introduce the median filtering technique as a post-processing step to smooth the outputs. Considering the temporal continuity of the sound events, the median filtering before thresholding is capable of removing the isolated false positives and false negatives, which is beneficial for the detection performance.

2.3 Evaluation Metrics

To evaluate the performance of the model comprehensively, we introduce multiple metrics including the event-based metrics, segment-based metrics, and tagging-based metrics [8]. Each of these metrics consists of common binary classification metrics [12] such as precision (P), recall (P), and F-score (F), which can be formulated as

$$\mathrm{P} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}}, \quad \mathrm{R} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}, \quad \mathrm{F} = \frac{2\times\mathrm{P}\times\mathrm{R}}{\mathrm{P}+\mathrm{R}} \tag{3}$$

where TP, FP, and FN represent the number of true positives, false positives, and false negatives respectively. Now we will introduce the specific metrics in detail.

Event-based metrics. The event-based metrics provide the most fine-grained evaluation of the sound event detection performance while bringing about the greatest difficulty, which checks every existing sound event instance according to the manually annotated reference. As is shown in Figure 6, the subjectivity of the temporal boundaries is alleviated by allowing some degree of misalignment called a collar between the compared events in the reference and system output. An event instance is counted as a true positive if it has the same label as the corresponding reference event, and its temporal boundaries lie within the permitted temporal collar with respect to the reference event.

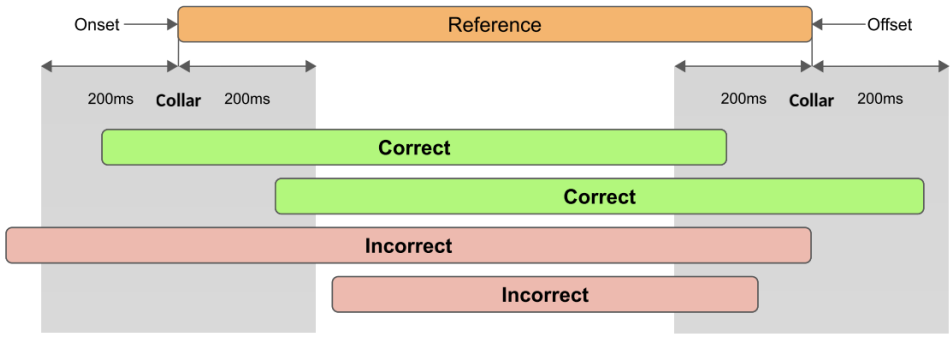

Figure 6: An illustration of the event-based evaluation. The reference and system output are compared with an error margin allowed to account for the temporal misalignment.

Segment-based metrics. The segment-based metrics are more relaxed compared to the event-based metrics, which divide the audio clip into segments and compare the reference and system output on a fixed temporal grid. The evaluation process will quantize sound event onsets and offsets, extending the sound event length such that the activity indicators cover all segments in which the sound is active, even for a very short time. Such a strategy is more robust to the subjectivity of the manual annotations, which is intuitively illustrated in Figure 7.

Tagging-based metrics. The tagging-based metrics are the coarsest evaluation metrics, which only check the presence or absence of sound events in the audio clip. Such a strategy ignores the temporal information of the sound events and directly reflects the effectiveness of supervised learning. In addition, we will compute the mean average precision (mAP) [12] based on the tagging-based metrics, which generally reflects the ability of the model to detect multiple sources of sound events.

2.4 Data Augmentation

To further improve the robustness of the model, we investigate multiple data augmentation techniques [10, 5] and apply them to the training data. Methods for data augmentation used in sound

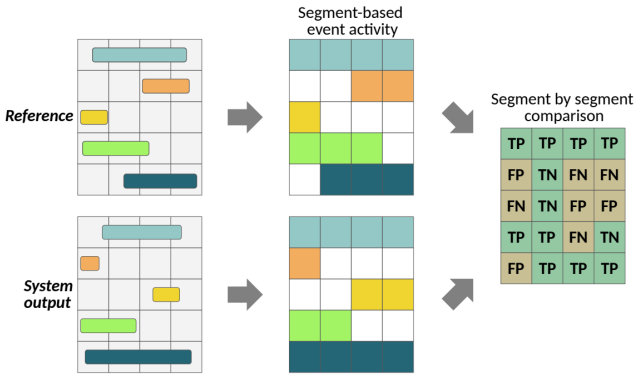

Figure 7: An illustration of the segment-based evaluation. The reference and system output are compared on a fixed temporal grid, which is more robust to the temporal misalignment.

event detection tasks range from basic signal manipulations such as time stretching, pitch shifting, and dynamic range compression, to more complex ones such as convolution with various impulse responses, block mixing, and simulating a set of noise conditions. Since we use the LMS as the input feature, we mainly focus on the time-frequency domain augmentation techniques based on the spectrogram, including Gaussian noise (GN), time shift (TS), time mask (TM), frequency shift (FS), and frequency mask (FM), which are intuitively illustrated in Figure 8.

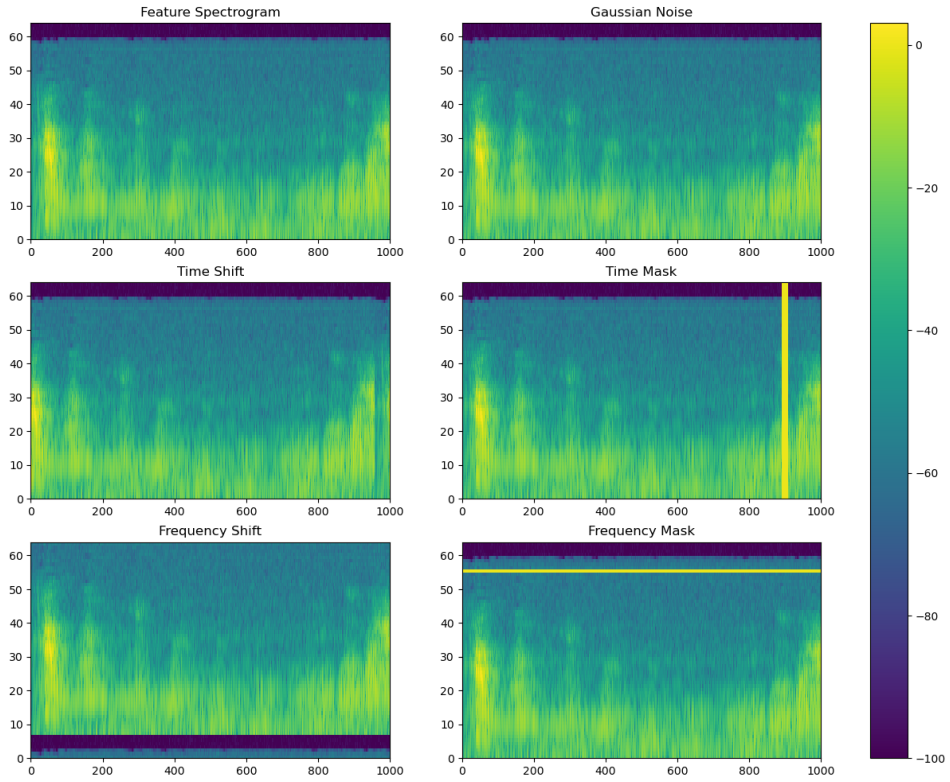

Figure 8: An illustration of the data augmentation techniques based on the spectrogram. Gaussian noise, time shift, time mask, frequency shift, and frequency mask are included.

Gaussian noise. For each element in the spectrogram, we add a random value sampled from an independent Gaussian distribution with zero mean and a certain standard deviation. It is capable of simulating the background noise in the real-world environment, improving the model's robustness.

Time shift. We shift the spectrogram along the time axis by a random number of frames within a certain range. It is capable of simulating the temporal misalignment of the sound events, which is beneficial for the model to learn the temporal invariance of the sound events.

Time mask. We randomly select a continuous segment along the time axis and set the corresponding elements in the spectrogram to zero. It draws inspiration from the idea of ensemble learning, which is capable of improving the generalization ability of the model.

Frequency shift. We shift the spectrogram along the frequency axis by a random number of bins within a certain range. It is capable of simulating the frequency shift of the sound events, which is beneficial for the model to learn the frequency invariance of the sound events.

Frequency mask. We randomly select a continuous segment along the frequency axis and set the corresponding elements in the spectrogram to zero. Similarly, it draws inspiration from the idea of ensemble learning, which is capable of improving the generalization ability of the model.

3 Experiments

In this section, we will implement and train the models on the given dataset and compare the performance of different settings. We will research how the hyperparameters affect the performance and try to optimize the detection results. We also conduct ablation studies to explore the effectiveness of different data augmentation methods. In the following parts, we will present the results in detail.

3.1 Experiment Setup

We employ the DCASE2018 dataset [6] to train and evaluate the models. The dataset is built for the challenge of large-scale weakly labeled semi-supervised sound event detection in domestic environments, containing 10 unique events. The entire training dataset consists of 55990 clips, but only 1578 clips contain labels. The rest 54412 clips are split into in-domain data of 14413 clips and out-domain data of 39999 clips. Most clips in this dataset contain one or two distinct events.

In the default setting, we do not employ any data augmentation method. We use a CRNN model with 5 convolutional layers and 1 recurrent layer, and the hidden size is set to 128. The reduced time resolution will be restored with a linear interpolation. We train the model for 100 epochs with a batch size of 64 using AdamW [4, 7] optimizer with a learning rate of 10^{-3}. We reduce the learning rate by a factor of 0.2 if the validation loss fails to decrease for 3 epochs. We will early stop the training if the validation loss does not decrease for 10 epochs. The model with the best validation loss will be selected for evaluation. For evaluation, we post-process the outputs with a median filter and a binary threshold. The window size is set to 7 and the threshold is set to 0.5.

3.2 Model Performance

To compare the performance of the models with different architectures, we implement the CRNN models with various layer numbers and hidden sizes. All the hyperparameters follow the default setting except for the layer number and the hidden size. The event-based, segment-based and tagging-based metrics together with the mean average precision are reported in Table 1.

From the table we can see that the model with 4 convolutional layers, 1 recurrent layer and 128 hidden units achieves the best performance in terms of mean average precision. The model with 5 convolutional layers, 2 recurrent layer and 256 hidden units achieves the best performance in terms of event-based metrics. The model with 5 convolutional layers, 1 recurrent layer and 128 hidden units achieves the best performance in terms of segment-based and tagging-based metrics. We can observe that as the number of convolutional layers increases, the performance of the model significantly improves. In contrast, the number of recurrent layers show no significant impact on the performance. An appropriate hidden size can also improve the performance, but it should match the number of convolutional and recurrent layers.

3.3 Ablaion Study

To explore the effectiveness of different data augmentation methods, we conduct ablation studies based on the default setting. We compare the performance of the models with different data aug-

Table 1: The performance of the models with different architectures. The numbers in the model name represent the layer number and the hidden size (e.g., 3-1-64 denotes a CRNN model with 3 convolutional layers, 1 recurrent layer and 128 hidden units). The event-based, segment-based and tagging-based metrics together with the mean average precision are reported.

Model	mAP	Event-based			Segment-based			Tagging-based		
		F	P	R	F	P	R	F	P	R
3-1-64	.615	.108	.104	.130	.542	.575	.516	.549	.672	.483
3-1-128	.664	.108	.106	.125	.562	.576	.560	.598	.655	.561
3-2-128	.651	.164	.158	.199	.567	.573	.576	.617	.629	.616
3-2-256	.637	.148	.136	.202	.549	.562	.574	.589	.638	.577
4-1-64	.659	.116	.122	.116	.567	.592	.553	.605	.659	.580
4-1-128	**.693**	.166	.159	.200	.609	.637	.591	.643	.702	.607
4-2-128	.676	.149	.146	.189	.575	.571	.600	.620	.628	.630
4-2-256	.660	.170	.158	.223	.582	.594	.591	.610	.674	.586
5-1-64	.684	.144	.150	.154	.574	**.676**	.509	.638	**.722**	.585
5-1-128	.692	.180	.167	.229	**.611**	.624	.619	**.663**	.673	.660
5-2-128	.686	.155	.153	.189	.603	.602	.624	.647	.655	.647
5-2-256	.686	**.181**	**.169**	**.244**	.603	.593	**.645**	.660	.664	**.672**

mentation methods. The event-based, segment-based and tagging-based metrics together with the mean average precision are reported in Table 2.

Table 2: The performance of the models with different data augmentation methods. The method names are abbreviated as NA (No Augmentation), GN (Gaussian Noise), TS (Time Shift), TM (Time Mask), FS (Frequency Shift), and FM (Frequency Mask). The event-based, segment-based and tagging-based metrics together with the mean average precision are reported.

Method	mAP	Event-based			Segment-based			Tagging-based		
		F	P	R	F	P	R	F	P	R
NA	**.704**	.188	**.177**	.230	**.612**	**.644**	.598	**.653**	**.702**	.626
GN	.689	.169	.156	.205	.609	.634	.601	.646	.696	.614
TS	.686	**.188**	.172	**.239**	.606	.614	**.610**	.644	.655	**.642**
TM	.694	.183	.172	.232	.607	.634	.598	.643	.668	.640
FS	.687	.168	.157	.201	.608	.634	.594	.633	.699	.598
FM	.686	.162	.154	.188	.606	.630	.592	.644	.692	.615

From the table we can see that different data augmentation methods have different impacts on the performance compared to the default setting, among which the time shift method achieves the best performance with event-based metrics slightly improved. The time mask method also shows satisfactory performance in terms of multiple metrics. The Gaussian noise method fails to improve the performance, which may be due to the fact that the audio clips already contain environmental noise. The frequency shift and frequency mask methods also show no significant improvement. We owe this result to the terrible interpretability of the transformation in the frequency domain, where the low frequency components are indispensable as the basic features of the sound events.

4 Conclusion

In this project, we research on the approach of weakly supervised sound event detection. We first discuss the challenges of the weakly supervised setting and the potential solutions. We propose a valid framework which can be divided into multiple stages. We introduce the pipeline from four main aspects including the feature extraction, model architecture, evaluation metrics, and data augmentation. Then we implement the proposed framework and conduct experiments on the DCASE2018 dataset. We compare the performance of the models with different architectures to investigate how the hyperparameters affect the results. We find that the number of convolutional layers has a sig-

nificant impact on the performance, while the number of recurrent layers and the hidden size show no significant impact. The best event-based metrics are achieved by the model with 5 convolutional layers, 2 recurrent layers and 256 hidden units.

Besides, we conduct ablation studies to investigate the effectiveness of different data augmentation methods. We find that though the model with no augmentation achieves satisfactory performance, the time shift and time mask methods can further improve the event-based metrics. However, the Gaussian noise method shows no significant impact on the performance, which indicates that the environmental noise is already included in the audio clips. The frequency shift and frequency mask methods also show no significant improvement, where we owe this result to the terrible interpretability of the transformation in the frequency domain.

References

[1] Emre Cakır, Giambattista Parascandolo, Toni Heittola, Heikki Huttunen, and Tuomas Virtanen. Convolutional recurrent neural networks for polyphonic sound event detection. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 25(6):1291–1303, 2017.

[2] Yin Cao, Qiuqiang Kong, Turab Iqbal, Fengyan An, Wenwu Wang, and Mark D Plumbley. Polyphonic sound event detection and localization using a two-stage strategy. *arXiv preprint arXiv:1905.00268*, 2019.

[3] Heinrich Dinkel, Mengyue Wu, and Kai Yu. Towards duration robust weakly supervised sound event detection. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:887–900, 2021.

[4] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[5] Tom Ko, Vijayaditya Peddinti, Daniel Povey, and Sanjeev Khudanpur. Audio augmentation for speech recognition. In *Interspeech*, volume 2015, page 3586, 2015.

[6] Qiuqiang Kong, Turab Iqbal, Yong Xu, Wenwu Wang, and Mark D Plumbley. Dcase 2018 challenge surrey cross-task convolutional neural network baseline. *arXiv preprint arXiv:1808.00773*, 2018.

[7] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[8] Annamaria Mesaros, Toni Heittola, Tuomas Virtanen, and Mark D Plumbley. Sound event detection: A tutorial. *IEEE Signal Processing Magazine*, 38(5):67–83, 2021.

[9] Md Sahidullah and Goutam Saha. Design, analysis and experimental evaluation of block based transformation in mfcc computation for speaker recognition. *Speech communication*, 54(4):543–565, 2012.

[10] Connor Shorten and Taghi M Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of big data*, 6(1):1–48, 2019.

[11] Vibha Tiwari. Mfcc and its applications in speaker recognition. *International journal on emerging technologies*, 1(1):19–22, 2010.

[12] Zhi-Hua Zhou. *Machine learning*. Springer nature, 2021.