ATTENTION-BASED CONVOLUTIONAL NEURAL NETWORK MODEL AND ITS COMBINATION WITH FEW-SHOT LEARNING FOR AUDIO CLASSIFICATION

A Dissertation
Presented to
The Academic Faculty

By

You Wang

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Electrical and Computer Engineering

Georgia Institute of Technology

August 2022

ATTENTION-BASED CONVOLUTIONAL NEURAL NETWORK MODEL AND ITS COMBINATION WITH FEW-SHOT LEARNING FOR AUDIO CLASSIFICATION

Thesis committee:

Dr. David V. Anderson, Advisor
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Mark A. Davenport
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Christopher J. Rozell
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Thomas Ploetz
College of Computing
Georgia Institute of Technology

Dr. Eva L. Dyer
Department of Biomedical Engineering
Georgia Institute of Technology

Date approved: July 21, 2022

ACKNOWLEDGMENTS

First and foremost, I would like to thank my advisor Dr. David Anderson. He was the teacher of my very first class in Georgia Tech and I feel extremely lucky to have him as my advisor throughout my whole career as a graduate student. He is very caring, kind hearted and always there to help. Furthermore, he is more than just an advisor, he is also a mentor, who has provided me with considerable life advises and encouraged me whenever I am down. He also often invites us to his place for parties, boardgame nights, and other fun activities, which really makes me feel like home even though I'm thousands of miles away from my hometown. Without Dr. Anderson, my PhD experience would have become completely different and I really appreciate his help with all my heart.

Secondly, I would like to thank my parents who have always supported me even though they are in China. My mom is always concerned about how I'm living my life here in the US and my father always offers valuable suggestions regarding my PhD career, not to mention their constant financial support before I had my own stipends as a student. It's been three years since last time I saw them in person because of the pandemic, and I really hope that I can have my family reunion in the near future.

In addition, I am very grateful that I have the opportunity to collaborate with Dr. Antonio Cicone, from whom I learned a lot of knowledge about mathematics and many life lessons. I really appreciate his help during the BMW project, and I hope that I can work together with him again in the future.

Last but not least, I would like to give special thanks to my labmates, especially Chuyao Feng and Nicolas Shu. They've always been my close friends and both of them have taught me a lot about coding and linux system. Sharing thoughts with them is always fun and refreshing, which offers many inspirations for my own research.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

Environmental sound and acoustic scene classification are crucial tasks in audio signal processing and audio pattern recognition. In recent years, deep learning methods such as convolutional neural networks (CNN), recurrent neural networks (RNN), and their combinations, have achieved great success in such tasks. However, there are still numerous challenges left to be addressed in this domain. For example, in most cases, the sound events of interest will be present through only a portion of the entire audio clip, and the clip can also suffer from the background noise. Furthermore, in many application scenarios where the amount of labelled training data can be very limited, the application of few-shot learning methods especially prototypical networks have achieved great success. But metric learning methods such as prototypical networks often suffer from bad feature embeddings of support samples or outliers, or may not perform well on noisy data. Therefore, the proposed work seeks to overcome the above limitations by introducing a multi-channel temporal attention-based CNN model and then introduce a hybrid attention module into the framework of prototypical networks. Additionally, a Π-model is integrated into our model to improve performance on noisy data, and a new time-frequency feature is explored. Various experiments have shown that our proposed framework is capable of dealing with the above mentioned issues and providing promising results.

CHAPTER 1
INTRODUCTION

1.1 Overview

Environmental sound classification (ESC) recognizes an audio clip as primarily containing a particular type of environment sound. It is an important area of sound event detection and classification with applications in many areas including audio surveillance systems [1], hearing aids [2], smart rooms [3], smart cars, etc. Environmental sounds are a very diverse group of everyday audio events that can neither be described as speech nor as music [4]. Because of their non-stationary characteristic and complex patterns, the performance of conventional music or speech recognition techniques such as K-nearest neighbors [5], support vector machines [6], Gaussian mixture models [7], and hidden Markov models [8], in environmental sound classification is limited.

As an extension of ESC, acoustic scene classification (ASC) is an essential part of auditory scene analysis and involves labeling an entire recorded audio signal using a pre-defined semantic description such as "office" or "airport" [9]. The nature of an acoustic scene containing multiple different sound events simultaneously makes this task more complex and challenging. ASC has become an increasingly important research topic in recent years thanks to the Detection and Classification of Acoustic Scenes and Events (DCASE) challenge [10]. Figure 1.1 shows the overview of an acoustic scene classification system.

In general, audio classification systems consist of two major parts, data preparation and data modelling [9]. In the data preparation phase, audio signals are converted into different representations by feature extraction and one or more pre-processing techniques are applied, and sometimes data augmentation is applied if the amount of data is limited. When the inputs are prepared, a model combined with a learning paradigm is designed for a

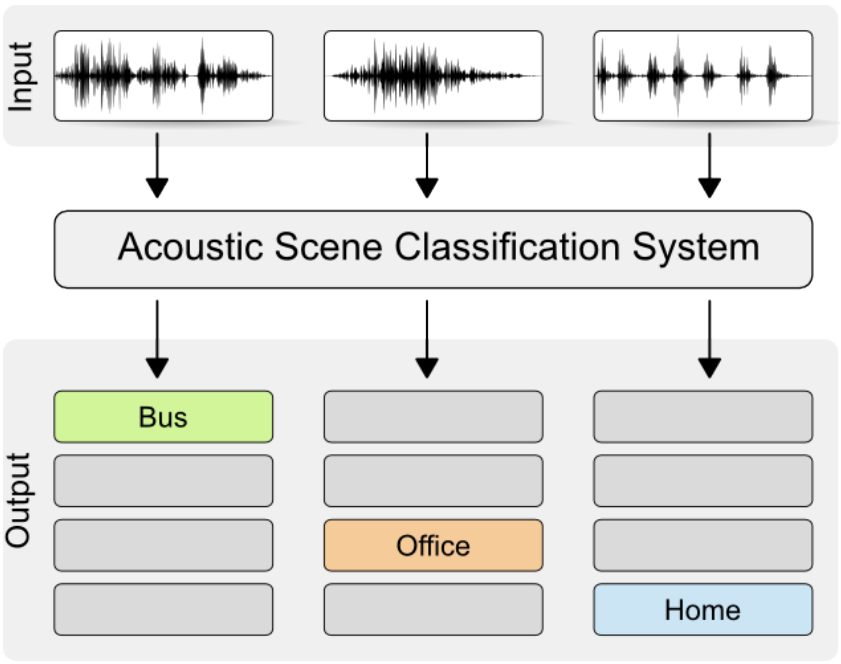

Figure 1.1: Overview of an acoustic scene classification system, provided by DCASE 2016 challenge [11]

particular classification task. Figure 1.2 shows the overall flowchart of how a typical audio classification problem is addressed.

Since digital audio signals are essentially 1-dimensional temporal sequences, unlike images which have 2 dimensions, the information provided is limited. Therefore, for most audio classification tasks, features of the audio signals are extracted. Such features usually add frequency or spatial information and serve dimension reduction purposes. Furthermore, the extracted features will go through several types of pre-processing steps so that they become more proper inputs for various classification models. Some of the most commonly used audio features and pre-processing methods are introduced in the following section.

1.2 Data Preparation

1.2.1 Audio features

Some of the most basic features for audio classification, especially for speech/music recognition, include zero-crossing rate (ZCR), pitch, short-time energy (STE), sub-band energy

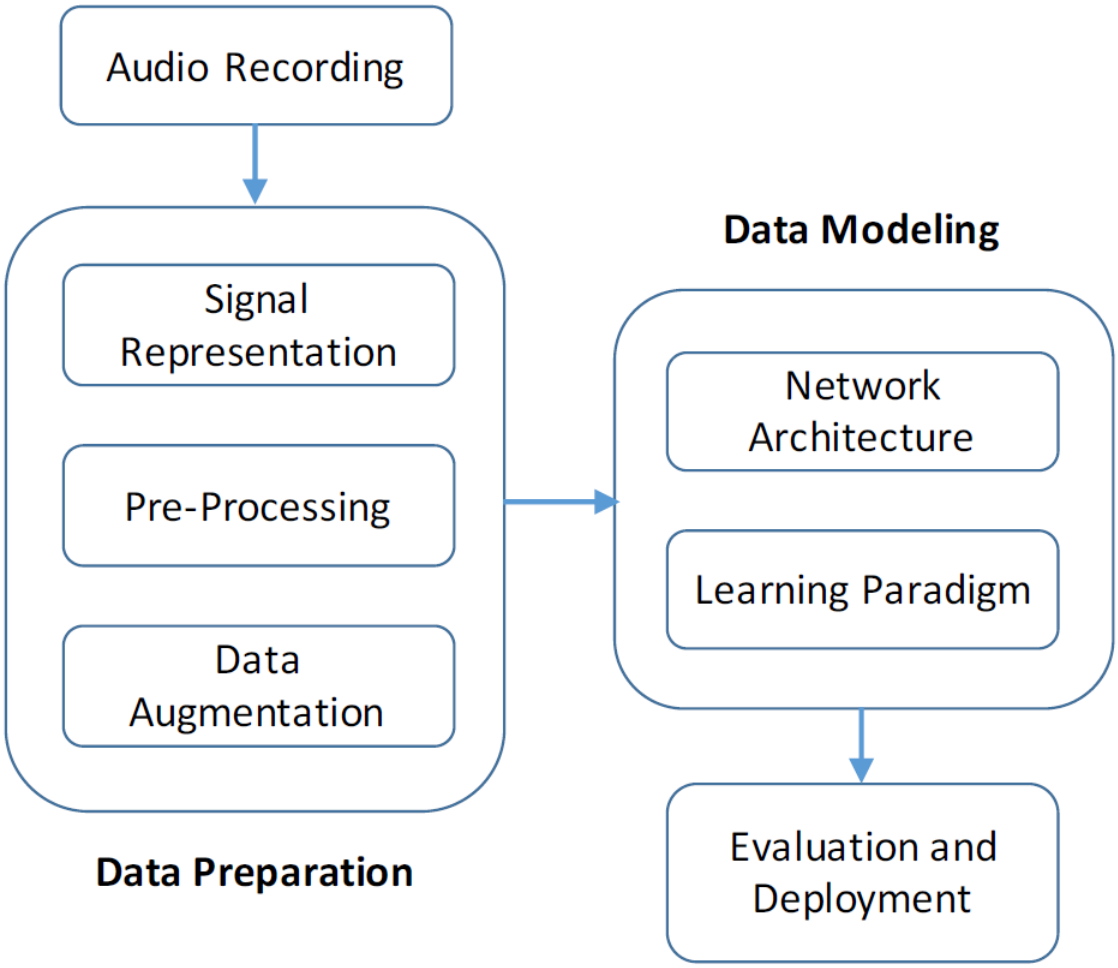

Figure 1.2: The typical processing flow of an audio classification system [9]

radio, etc [12]. These features can provide general temporal and spectral information of audio signals [12]. One of the most widely used features is the set of cepstral features, including Mel-frequency cepstral coefficients (MFCCs), log Mel-spectrograms and so on. These features are designed to mimic the behavior of how human perceives sound. Therefore, they provide a more compact temporal-spectral representation than a plain short-time-fourier-transform [9]. To compute these features from a raw audio signal, the frequency axis of the short-time-fourier-tranform needs to be firstly mapped to the Mel scale. Mel scale is a scale of pitches perceived by listeners that neighboring pitches have equal distances, and was proposed by Stevens *et al.* in 1937 [13]. This scale is designed to relate to the fact the humans can better differentiate low frequencies than high frequencies. Mel scale is logarithmic and the conversion from Hz to Mel can be described as:

$$Mel_f = 2595 \times log_{10}(1 + \frac{f_{Hz}}{700}) \tag{1.1}$$

MFCCs were introduced by Davis and Mermelstein in the 1980's [14] and have been state-of-the-art especially in speech recognition since then. The steps of computing MFCCs include: 1) Firstly compute the short-time-fourier-transform (STFT) of the audio signal; 2) Apply the Mel filterbank (Figure 1.3) to the power spectra of each time frame, and then compute the sum of energy within each filter; 3) Compute the logarithm of all filterbank energies; 4) Apply DCT to the results of the previous step, and take the pre-defined number of coefficients. Log Mel-spectrogram is the outcome of step 3) and it is commonly used in sound classification tasks, and it was also used in this work as the primary feature. Moreover, since digital audio signals are time series, it is beneficial to gain information of the dynamics of the power-based features. Therefore, deltas and delta-deltas, which stand for the first and second temporal derivatives were introduced as dynamic features for this purpose. It has been shown that combining original features with their corresponding dynamic features can improve classification performance [15].

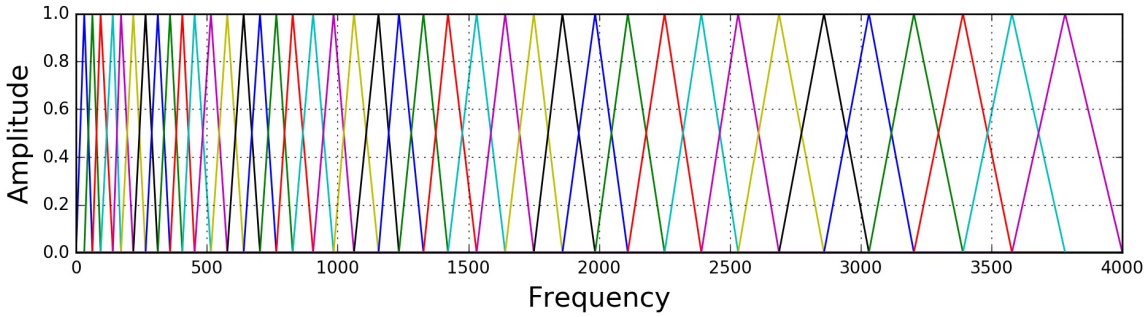

Figure 1.3: The Mel-scale filterbank [16].

In addition, some autoregression based features such as Linear Prdiction Coefficients (LPC) [17] and its alternative representation, Linear Prediction Cepstrum Coefficients (LPCC) [18] are also commonly used, especially for speech processing [12]. Some features are extracted based on signal decomposition methods such as Empirical Mode Decomposition (EMD) [19], which decomposes a signal into simple components called Intrinsic Mode Functions (IMFs), and its more stable version Ensemble Empirical Mode Decomposition (EEMD) [20] along with Iterative Filtering (IF) [21]. Other features that are also used by audio classification systems include wavelets [22, 23], scalogram [24], sub-band power distribution [25], and gammatone frequency cepstral coefficients (GFCCs) [26] etc. Occasionally, to avoid fixed pre-defined signal transformations, features can be directly learned from the raw sound waves. For example, Jaitly and Hinton [27] used Restricted Boltzmann Machines (RBMs) [28] to learn higher dimensional representations of audio signals to improve classification performance. Additionally, some end-to-end learning systems which directly take raw audio signals as inputs and combine feature extraction and classification together are used. These systems include AclNet [29] and AclSincNet [30], EnvNet [31], and SoundNet [32] etc. As opposed to time-frequency representation based features, raw audio signals contain the most original information and the phases are not discarded. But due to their one-dimension characteristic, extracting features becomes more challenging and they are usually outperformed by spectrogram-like representations such as Log Mel-spectrograms.

1.2.2 Standardization and Denoising

Machine learning techniques, especially those gradient-decent based algorithms, usually require feature standardization to facilitate the training process. It changes the feature distributions to have zero mean and unit standard deviation. Alternatively, logarithmic scaling is commonly applied on spectrogram-based features to compensate for the large dynamic range in environmental sound recordings [9]. In addition, dereverberation and low-pass filtering, which are most commonly applied to speech, can also become potential pre-processing methods for environmental audio scenes.

Audio pattern recognition always suffers from the contamination of the foreground information by background noise. One of the most basic methods is spectral subtraction proposed by Steven in 1979, where the noise spectrum level calculated during noise signal segments is subtracted from the whole spectrum [33]. Per-channel energy normalization (PCEN) [34] was used by Lostanlen *et al.* [35] to suppress stationary noise and enhance transient sound events in environmental audio clips. Wu *et al.* made the sound texture more salient by applying the Difference of Gaussian (DoG) and Sober filtering from image processing on feature maps and enhanced the edge information [36]. Han *et al.* [37] removed unwanted background noises by using background subtraction and applying median filtering. Additionally, several filtering approaches such as repeating pattern extraction technique (REPET) algorithm [38] and harmonic-percussive source separation (HPSS) [37, 39, 40] have also been used as pre-processing for ASC. So far, while most of the methods mentioned above have been proposed very recently, logarithmic magnitude scaling has been the only well-established method which is widely used among the best performing ESC and ASC algorithms [9].

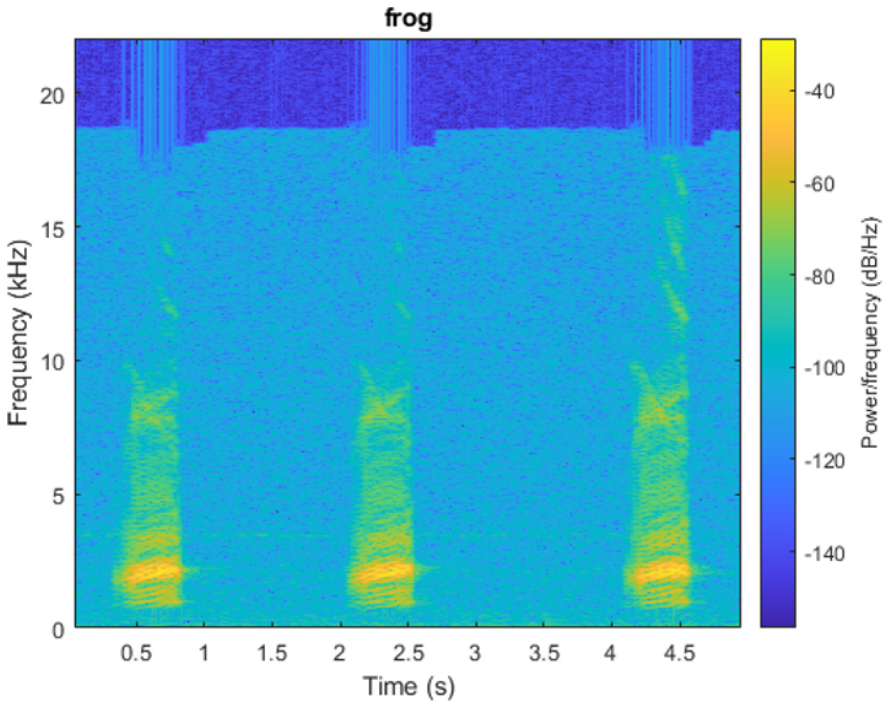

Figure 1.4: Log Mel-spectrogram of a 5-second-long frog sound.

1.3 Challenges

There are mainly three challenges faced in environmental sound and acoustic scene classification. First, for environmental sound recordings, the training samples are typically weakly labelled, meaning that the sound events of interest may have short duration and the location information of them is usually missing. As is shown in Figure 1.4, in a 5-second-long frog sound recording, the total duration of the sound events is less than 1.5 seconds. And between those sound events are long eventless intervals. Similarly in Figure 1.5, the actual sneezing event only occurs within the beginning 1.3 seconds, For both cases, the only information available are the clip level annotations “frog” and “sneezing”. Likewise for acoustic scenes, weakly labeled audio recordings are dealt with in almost every case. Since an acoustic scene contains different sets of sound events but is described by only a single pre-defined clip-level label, finding those that can better summarize the whole acoustic scene is key to the success of classification.

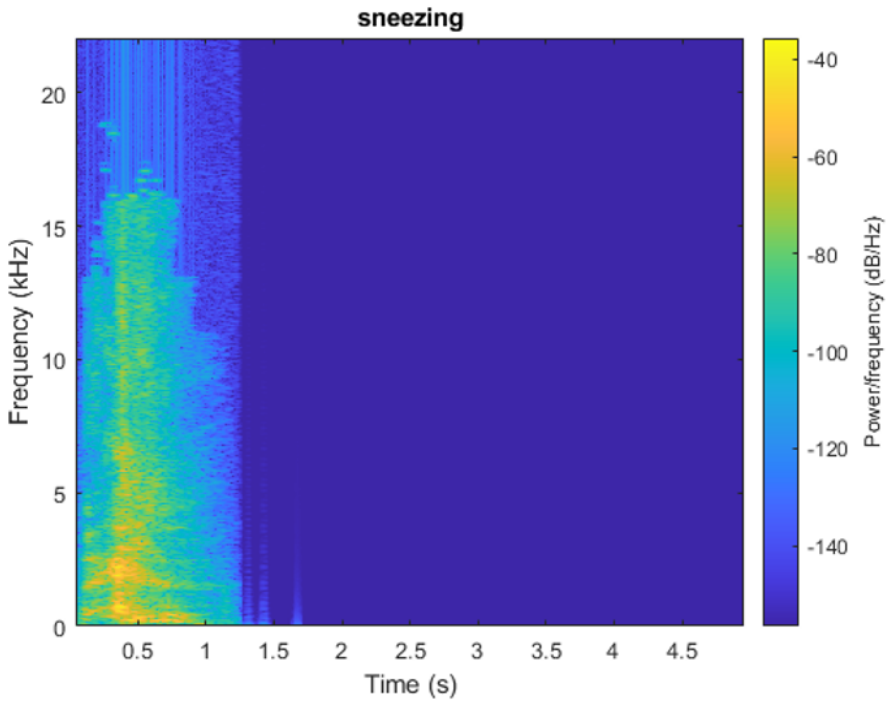

Figure 1.5: Log Mel-spectrogram of a 5-second-long sneezing sound.

The second challenge is irrelevant or unrelated information. Firstly, background noise will significantly affect the final classification performance and its impact should be suppressed. An example is shown in Figure 1.6, where the log Mel-spectrogram of a 5-second-long rain sound is plotted. Even though the rain event is present throughout the clip, it is heavily affected by noise, preventing classification models from correctly recognizing it. Some methods introduced in the previous section can be applied to suppress background noise, but in many cases these methods may also corrupt foreground sound events as well. Moreover, irrelevant sound events are also considered noise for weakly labeled data, especially for acoustic scenes. For example, the acoustic environment of an office should include sound events such as keyboard typing, mouse clicking, and air conditioning, etc. Whereas bird singing or train engine should be considered irrelevant.

Another challenge is data scarcity. The success of deep learning methods is built on the availability of large supervised training datasets, but in many applications the amount of training data can be very lacking. In some cases, labeling unsupervised data can cost

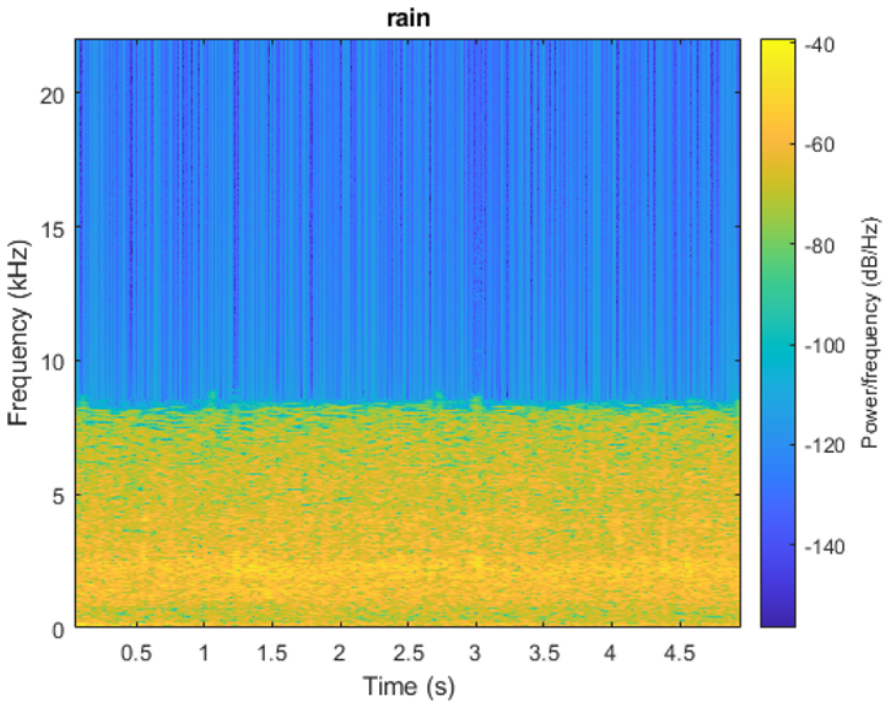

Figure 1.6: Log Mel-spectrogram of a 5-second-long rain sound.

too much human effort and be too time consuming; therefore, the available training data is extremely limited. Or some categories have no or very little amount of data, resulting in a overly unbalanced dataset. In these circumstances, the application of deep neural networks is basically impractical. Also, many neural network structures are built to be deep and heavy in accordance with big datasets, but their performance is not as good with smaller ones.

In the next section, the main contributions of this work are introduced and the outline of this thesis is presented.

1.4 Contributions and Thesis Outline

This thesis explores methods of classifying environmental sound events and acoustic scenes under the limitations mentioned above including, weak labels, noise and irrelevant information, and only a small amount of labeled data for the classes of interest.

To tackle the first two challenges, weakly labeled data and unrelated information, our

first contribution is the development of an multi-channel temporal attention CNN model for environmental sound classification. We believe that temporal attention in audio signals is helpful to emphasize key temporal features that are relative to the clip-level labels. We proposed an effective convolutional neural network structure with a multi-channel temporal attention (MCTA) block, which applies a temporal attention mechanism within each channel of the embedded features to extract channel-wise relevant temporal information. This multi-channel temporal attention structure will result in a distinct attention vector for each channel, which enables the network to fully exploit the relevant temporal information in different channels. The model was tested on some moderate sized datasets such as ESC-50, its subset ESC-10, and UrbanSound8K, along with development sets of DCASE 2018 and 2019. In our experiments, MCTA performed better than the single-channel temporal attention model and the non-attention model with the same number of parameters. Furthermore, we compared our model with some successful attention-based models and obtained competitive results with a relatively lighter network.

Lack of training data is often the reason that prevents a deep learning model from achieving satisfactory performance, and few-shot learning is commonly used to deal with such data scarcity. Therefore, the second contribution of this thesis is the exploration of prototypical networks, one of the most successful few-shot learning methods, with hybrid attention for few-shot audio classification. We introduced a hybrid attention module and combined it with the prototypical networks framework. This hybrid attention module consists of two blocks: a feature-level attention block, and an instance-level attention block. The feature-level attention block makes use of the previous MCTA model, which can highlight key embedded features to locate important information in weakly labeled data. And the instance-level attention block can emphasize crucial support instances to diminish the effect of bad support samples or outliers that contain irrelevant sound events. The performance of our model was also evaluated using the ESC-50 dataset and the $noise$ESC-50 dataset. The model was trained in a 10-way 5-shot scenario and tested in four few-shot

cases, namely 5-way 1-shot, 5-way 5-shot, 10-way 1-shot, and 10-way 5-shot. The results demonstrate that by adding the hybrid attention module, our model outperforms the baseline prototypical networks in all four scenarios. Also, our models were tested on a blended ESC-50 dataset where during 5-shot training, 2 of the 5 clean support samples of each class were randomly picked to be replaced by their noisy counterparts. And it was proved that with this training scenario, the performance gain by adding the instance-level block was amplified. In addition, we created a dataset which contains 108 environmental sound classes drawn from the FSD50K dataset. We tested our models on it and obtained promising results as well.

Additionally, to further improve the classification results on the *noise*ESC-50 dataset. We borrowed the idea of Π-model and incorporated it into our proposed hybrid attention prototypical network framework. Π-model is a self-ensembling method which was initially designed for semi-supervised learning problems. We made a few modifications to the original model, penalizing prediction difference between clean and noisy inputs in the loss function, attempting to enhance the robustness of our models over noise. We named our final model HAPPi, which stands for Hybrid Attentional Prototypical Networks with Π-model, and the results show that with Π-model, the performance on *noise*ESC-50 was indeed boosted.

Finally, we explored an innovative time-frequency representation named IMFogram and compared its performance on environmental sound classification with log Mel-spectrograms. We obtained similar results and the outcome laid foundation for an interesting future direction.

The thesis is organized as follows: Chapter 2 describes the multi-channel temporal attention CNN model; Chapter 3 provides a closer look at the hybrid attentional prototypical networks for few-shot audio classification; Chapter 4 introduces Π-model and the details of HAPPi; Chapter 5 explores IMFogram as a new feature; Chapter 6 presents the conclusion of this work and discuss the potential future research directions and plans.

CHAPTER 2
MULTI-CHANNEL TEMPORAL ATTENTION CONVOLUTIONAL NEURAL NETWORK MODEL

In this chapter, the details of the multi-channel temporal attention (MCTA) convolutional neural network model are presented. We start with a brief survey of and conventional deep learning approaches for audio classification, followed by an introduction of the attention mechanism and its application in image classification and audio classification models. Then the actual structure of our MCTA CNN model is showed. Afterwards, experiments on ESC, UrbanSound8K and DCASE datasets are presented and their results are discussed and analysed afterwards.

2.1 Deep Learning Approaches for Sound Classification

Traditional algorithms for audio classification include K-nearest neighbors [41, 42], support vector machines [43, 44], Gaussian mixture models [45, 46], and hidden Markov models [47, 48], etc. But with the support of more available labelled datasets, deep learning methods have become the most popular method that can achieve superior performance over the traditional approaches. Some popular datasets for environmental sound classification include ESC-50 [4], UrbanSound8k [49], and Google Audioset [50]. And for acoustic scene classification, many datasets were proposed by the challenges of Detection and Classification of Acoustic Scenes and Events (DCASE) [51, 52]. However, comparing with image datasets in Computer Vision community, the amount of audio datasets is still not enough and in many application cases, collecting a large amount of audio data is impractical and requires too much human labor.

Piczak [53] is one of the earliest researchers who applied CNN to environmental sound classification (ESC) and also developed the widely used ESC-50 dataset. By extracting

time-frequency representation based audio features such as log Mel-spectrograms and treating them as images, this deep CNN-based model outperformed the traditional baseline methods via mimicking the training process of image classification, and therefore laid the foundation for the following deep learning based audio classification models. Tokozume *et al.* [31] proposed an end-to-end model called EnvNet, which can perform feature extraction and classification together. It offered an alternative way of utilizing CNN to learn features directly from raw audio waves instead of creating handcrafted features. Similarly, Sailor *et al.* [54] also used a convolutional Restricted Boltzmann Machine (ConvRBM) as a front-end to extract features directly from raw audio waveforms, and used a supervised CNN as a back-end classifier. To overcome the shortage of labelled audio data, Salamon [55] explored the influence of different data augmentation techniques on deep CNN classification performance. After the release of the Google Audioset, Kumar [56] proposed a deep CNN structure that can effectively transfer knowledge from Audioset and be used for solving other target tasks such as environmental sound classification.

Similar as ESC, most state-of-the-art ASC neural network models use CNN based structures [9]. Traditionally, input features go through a series of convolutional layers for further feature extraction and pooling layers for dimension reduction. Alternatively, Ren *et al.* proposed an atrous CNN structure based on dilated convolutional kernels [57], where the receptive field is large and local pooling layers are removed. Koutine *et al.* stated that restricting the size of the receptive field helps improve ASC performance [58], and the same authors further proposed a Frequency-aware Convolution layer which concatenates a new channel containing pixel-wise frequency information [59]. In addition, Basbug and Sert [60] proposed a cascaded CNN architecture using spatial pyramid pooling method to pool features on multiple spatial resolutions.

2.2 Attention Models

In most cases, we are provided with audio datasets that only have clip-level labels, or weakly labeled. One of the common clip-level sound classification processing schemes is segment based processing [61], where an audio clip is split into shorter segments and each of them shares the same label as the clip-level label. Each segment is then considered as one input instance and a classifier is trained on these instances. Afterwards, during inference, the final clip-level prediction decision is made based on the aggregation of segment-level predictions by majority voting or probability voting [62, 53]. This type of method is simple and can be efficient [61], however, it is often incorrect to assume that every segment bears the same label as the whole recording. Furthermore, if the sound events of interest have short durations, it can be easy for long sections of background noise to "outvote" them, resulting in wrong predictions. Considering the frog sound (Figure 1.4) in **CHAPTER 1** again as an example, since the frog sound events are very short, most segments can be labeled "noise" instead of "frog". In addition, another major drawback of this type of method is that it is hard to determine the optimal length of each frame or segment [12]. If the length is too small, the long-term variations of the sound events may not be well captured, and if it is too big, then irrelevant information will be included which can result in bad predictions [12]. Sailor *et al.* [54] applied silence removal algorithm before training to alleviate this issue, but it would take extra processing time and the algorithm itself may not be highly reliable. Some of the models described in the previous section take each audio clip as a whole, but they are still based on an underlying assumption that every frame or segment of an audio clip carries relevant information corresponding to the clip-level label.

To solve this problem, the attention mechanism is then introduced. The concept of attention was originally inspired by the cognitive process of attention in organisms. For example, when we are looking at an image, our visual attention system will allow us to focus on certain regions to help us perceive key features. Figure 2.1 [63] shows a Shiba

Inu wearing a men's outfit. We can recognize that this is a dog by focusing on the pointy ears, the watery doggy eyes, and a dog's nose. However, if the face of the dog is removed, we would not be able to know what this photo is about only based on the sweater and the blanket [63]. Similarly, as is shown in Figure 2.2 [63], in a sentence we would expect a food word after the appearance of "eating", while the color word is more likely to describe the food than "eating" [63]. In this case, we would say that "green" tends to "attend" to "apple" rather than "eating".

When it comes to implementing attention in deep learning models, the main goal of the many proposed models is to discover the best approach for identifying the salient regions or features. In other words, the essence is assigning contribution weights on different parts of features, namely channels, spectral or spatial contents, and temporal frames. The implementation of attention in deep learning was first introduced in [64] for machine translation, as is shown in Figure 2.3. This model inserted an attention layer between a bidirectional RNN encoder and an RNN decoder, by creating a context vector as a weighted sum of a sequence of annotations h_i. And the weights are based on the RNN hidden states and the annotations [64]. Since then, attention-based neural networks have been widely used in computer vision [65], natural language processing [66] and speech recognition [67]. Some architectures that are very successful in computer vision systems, such as squeeze-and-excitation networks (SENet) [68], convolutional block attention module (CBAM) [69], and dual attention network (DANet) [70], also have significant impact on audio. In SENet [68], an attention vector is formed where each element is a weight that is assigned to a respective channel, which is referred to as channel attention. This model laid the foundation for many following attention models and its structure is shown in Figure 2.5. CBAM [69] extended the idea of SENet, by concatenating the channel attention and the spatial attention to form a sequential model. DANet [70] is a variant of CBAM, where deep embedded feature maps go through spatial attention and channel attention in parallel, instead of sequentially.

Just like images and texts, attention mechanisms can also be applied to audio signals.

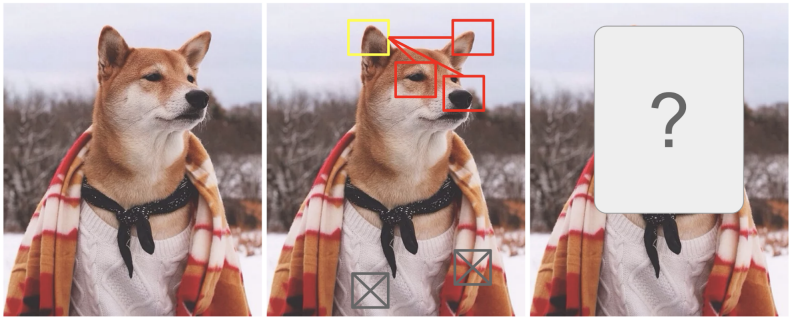

Figure 2.1: A Shiba Inu in a men's outfit. Source: Instagram @menseardog [63]

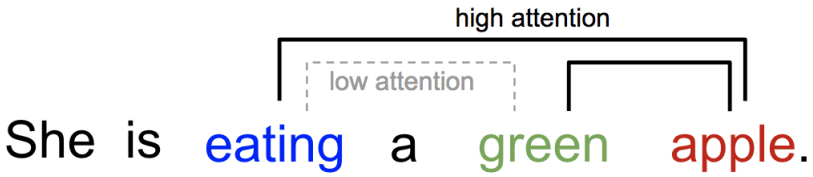

Figure 2.2: One word "attends" to other words in the same sentence differently [63].

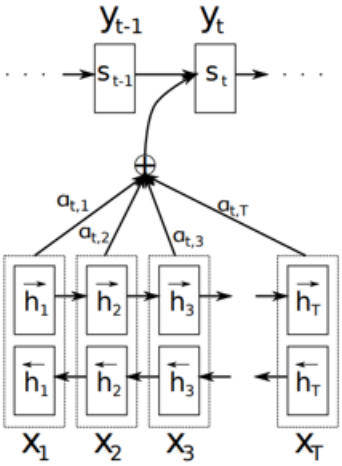

Figure 2.3: Attention implementation by *Bahdanau et al.* [64].

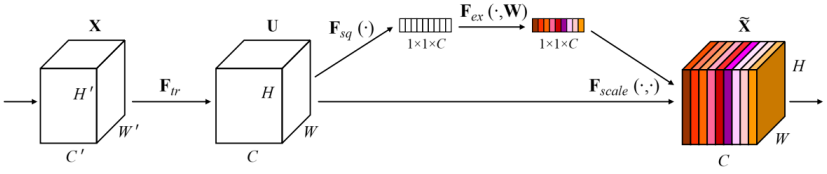

Figure 2.4: Structure of squeeze-and-excitation networks [68].

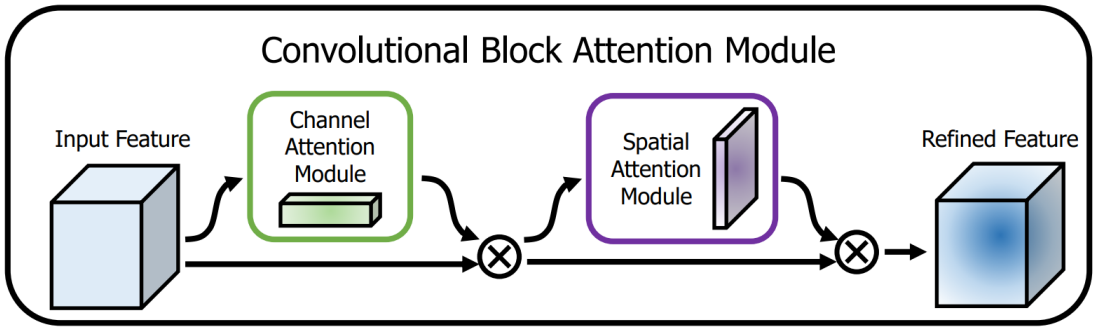

Figure 2.5: The overview of convolutional block attention module (CBAM) [69].

Because of the unique temporal structure of audio signals, similar to the previous channel or spatial attention, temporal attention has been very widely used for audio classification in recent years. Figure 2.6 shows an example of three feature maps of a 5-second-long sneezing sound, namely log Mel-spectrogram, deltas and delta-deltas. From this figure it can be seen that the sneezing sound event only occurs within a small part of the whole signal duration, and the relevant information presented in these three feature maps are temporally different. Therefore, it is plausible to focus on these time regions and pay less attention to the remaining areas. For sound classification tasks, numerous research projects have explored different attentional deep learning architectures over the past few years. Yu *et al.* [71] proposed a multi-level attention model for weakly audio classification with one-channel inputs where multiple attention modules are applied after intermediate layers as well as the final fully connected layer. Li *et al.* [24] proposed a multi-stream network with temporal attention of which the structure has three streams with a single temporal attention vector on all of them. Zhang *et al.* [72] integrated temporal attention into its CRNN architecture and performed well on the ESC-50 dataset. Figure 2.7 shows the structure of a temporal attention vector generation for CNN layers [72]. Zhang *et al.* [73] also borrowed the idea of CBAM and replaced the spatial attention part with temporal attention.

For audio scene classification, locating the salient regions or features are particularly appropriate since salient characteristics of an acoustic scene may be only briefly present or

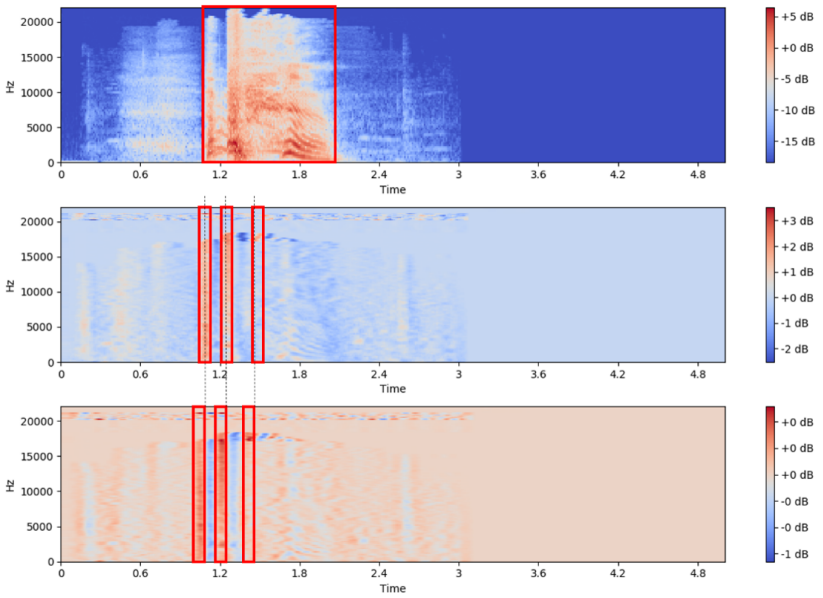

Figure 2.6: Three feature maps of a sneezing sound. From top to bottom: Log Mel-spectrogram, deltas, and delta-deltas.

may reside only in certain features. Li *et al.* introduced a multi-level attention architecture which combines convolutional layers with gated linear units (GLUs) and attaches one layer of bi-directional gated recurrent units at the end of the CNN [74]. Zhao *et al.* introduced a spatial attention pooling model for DCASE 2018 [75], and then further combined the idea with Atrous CNN [57].

2.3 Revisiting Audio Attention

One goal of the many proposed attention mechanisms is to discover the best approach for identifying the salient regions or features. As described in the previous section, because of the unique temporal structure of audio signals, temporal attention has been very widely used in audio classification models [24, 71, 72, 73]. As is shown in Figure 2.6, for different feature maps, the locations of key feature regions can be very different. Hypothetically, if temporal attention vectors are learned based on these three feature maps, their shapes should also be different, as can be seen in Figure 2.9. However, most audio classification models with temporal attention apply only one single temporal attention vector on feature

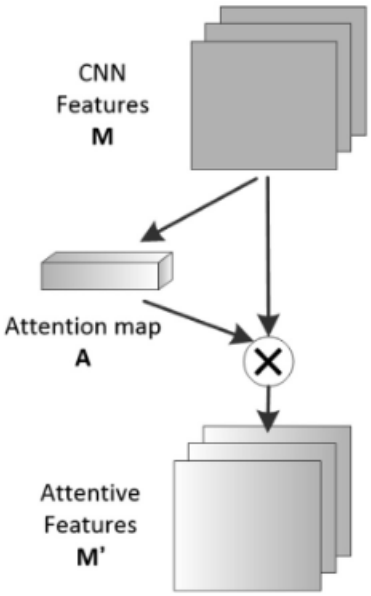

Figure 2.7: Temporal Attention for CNN Layers [72].

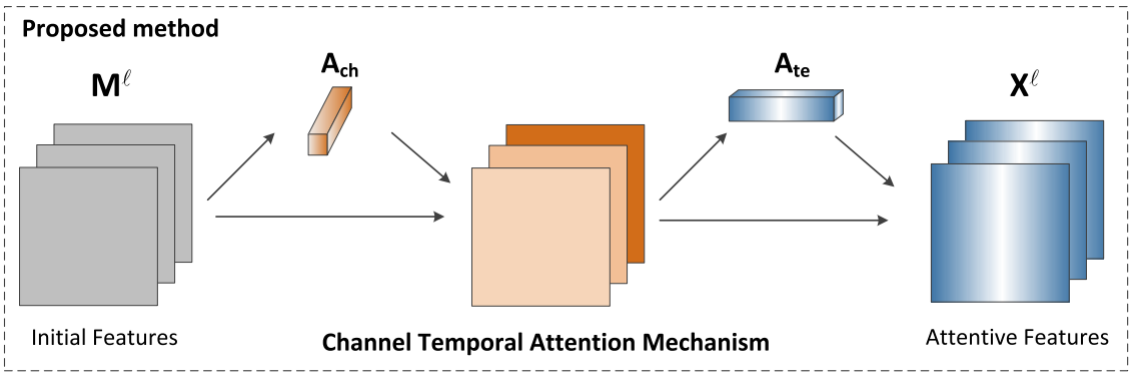

Figure 2.8: Channel Temporal Attention Mechanism [73].

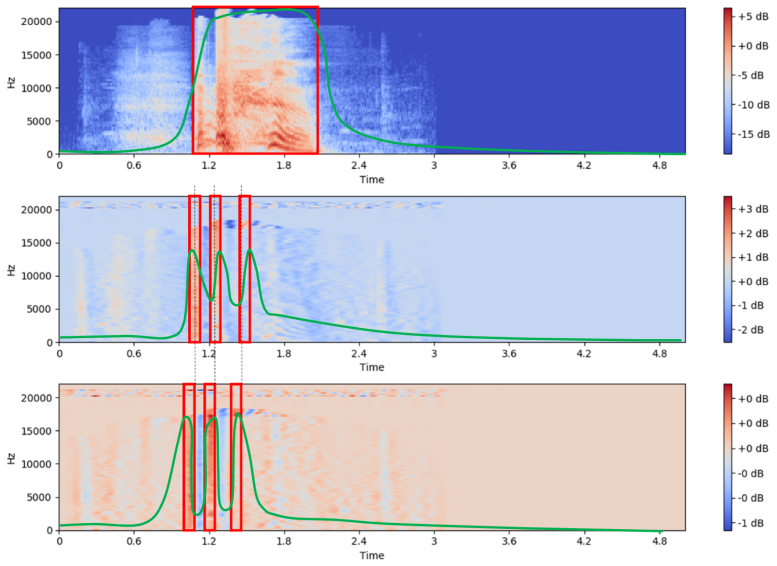

Figure 2.9: Three feature maps of a sneezing sound with respective attention vectors. From top to bottom: Log Mel-spectrogram, deltas, and delta-deltas.

maps of all channels. Therefore, these models fail to take advantage of the fact that feature maps in different channels actually have different temporal structures. To overcome this limitation, we propose a CNN model with a multi-channel temporal attention (MCTA) block that extracts different temporal attention vectors for different channels to more fully exploit channel-wise temporal information.

2.4 Multi-Channel Temporal Attention (MCTA) CNN Model

Typically, a CNN will apply convolution on the input with multiple filters to extract more high-level channel-wise features. These channels contain information from different aspects of the input, all of which will contribute to the final classification. Our proposed MCTA model provides each channel with a unique attention vector, better exploiting the different structures of channel-wise feature maps and different information associated with them.

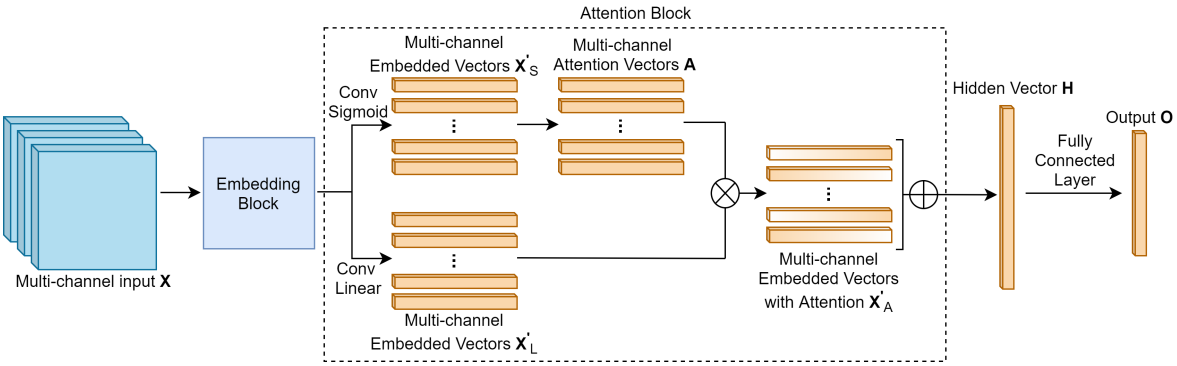

Figure 2.10: The block diagram of our proposed multi-channel temporal attention model. The $\otimes$ sign stands for Hadamard product and the $\oplus$ sign represents summing over the time dimension and squeezing the frequency dimension.

2.4.1 Model Architecture

The architecture of our proposed model is shown in Figure 2.10. The model takes multi-channel feature maps $\mathbf{X} \in \mathbb{R}^{C \times T \times F}$ as input where C is the number of channels, T is number of time frames and F is number of Mel-frequency bins, and passes them through an embedding block that consists of several convolutional and max-pooling layers. For example, the feature map extracted from ESC-50 dataset has a size of $3 \times 431 \times 128$. The embedding block will embed the input to a hidden number of channels and aggregate it along the frequency dimension, yielding a set of embedded feature vectors $\mathbf{X}' \in \mathbb{R}^{C' \times T' \times 1}$, where $C' = 512$ and $T' = 52$ in the ESC-50 case. These embedded vectors then enter the attention block where the network starts to bifurcate to generate the further embedded vectors $\mathbf{X}'_L$ and the attention vectors $\mathbf{A}$ respectively, and then the attended vectors $\mathbf{X}'_A$ are obtained by performing element-wise multiplication. Finally the time and frequency dimensions of $\mathbf{X}'_A$ are both squeezed to form a hidden vector $\mathbf{H} \in \mathbb{R}^{C' \times 1}$, which is then passed through a fully connected layer for the final classification.

2.4.2 The Embedding Block

Figure 2.11 presents the details of the embedding block. Because of the size of the datasets used in our work, a relatively shallow CNN structure is used, having a total of 5 convo-

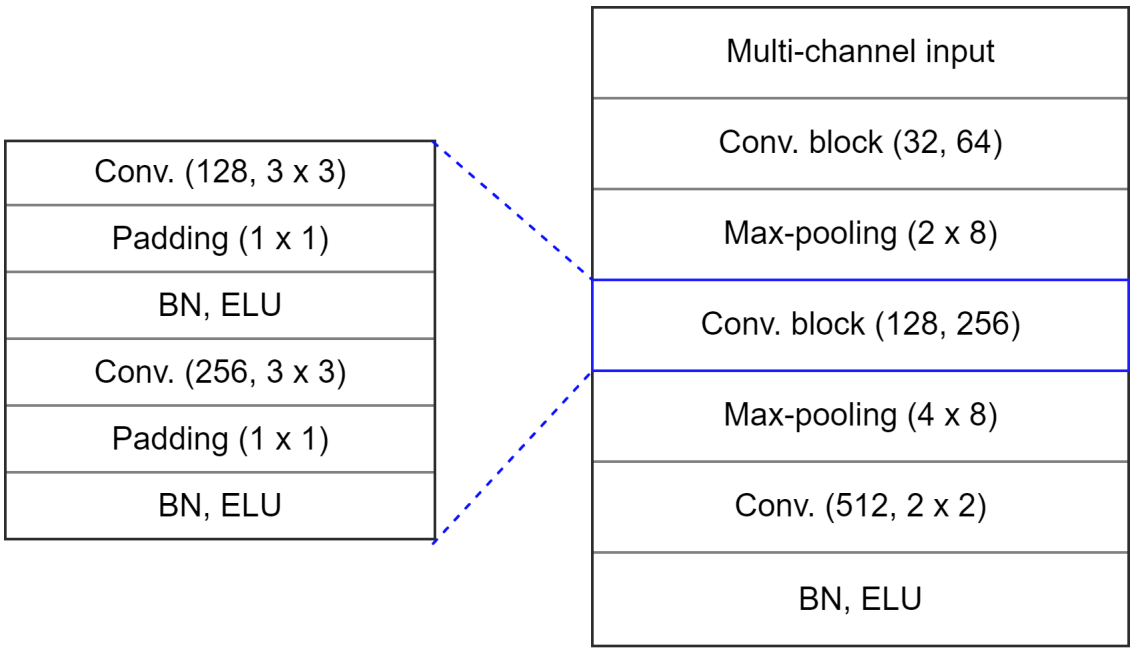

Figure 2.11: Details of the embedding block.

lutional layers and 2 max-pooling layers. Inspired by the work of Kumar [56], our embedding block is designed to eventually aggregate the 128 Mel bins to 1 and reduce the time dimension by max pooling to extract useful segment-level temporal information. After each convolutional layer, batch normalization (BN in the figure) is performed and ELU (Exponential Linear Unit) is used as activation. The numbers in the brackets after "Conv. block" indicate the numbers of filters of the two convolutional layers in them. The numbers after "Max-pooling" and "Padding" stand for kernel sizes. The numbers after "Conv." are the numbers of filters and the kernel size.

2.4.3 The Attention Block

There are two branches in the attention block and the details are described below. In the attention branch, a 1-by-1 convolutional layer and sigmoid activation is firstly applied to the output of the embedding block $\mathbf{X}' \in \mathbb{R}^{C' \times T' \times 1}$:

$$\mathbf{X}'_S = Sigmoid(Conv^{1\times1}(\mathbf{X}')) \tag{2.1}$$

where $\mathbf{X}'_S$ has the same dimensions as $\mathbf{X}'$ and $Conv$ stands for a convolutional layer. Then, the attention vectors $\mathbf{A} \in \mathbb{R}^{C' \times T' \times 1}$ are obtained by performing normalization along the T' dimension to make sure the weights sum up to 1, therefore acting like probabilities:

$$\mathbf{A}(c,t,1) = \frac{\mathbf{X}'_S(c,t,1)}{\sum_{t=1}^{T'} \mathbf{X}'_S(c,t,1)} \tag{2.2}$$

where $c \in [1, C']$ and $t \in [1, T']$ are indices of the channel and time dimension. This step guarantees that the attention is applied temporally and different across channels. In the other branch, $\mathbf{X}'$ goes through the same convolutional layer but without activation to become $\mathbf{X}'_L$ with the same size:

$$\mathbf{X}'_L = Conv^{1\times1}(\mathbf{X}') \tag{2.3}$$

Afterwards, the attention weighted feature vectors $\mathbf{X}'_A \in \mathbb{R}^{C' \times T' \times 1}$ are obtained by element-wise multiplication between $\mathbf{A}$ and $\mathbf{X}'_L$:

$$\mathbf{X}'_A = \mathbf{X}'_L \circ \mathbf{A} \tag{2.4}$$

where $\circ$ stands for Hadamard product. In the end, a single hidden vector $\mathbf{H} \in \mathbb{R}^{C' \times 1}$ is obtained by summing up $\mathbf{X}'_A$ along the time dimension and squeeze the frequency dimension before it is fed into a final fully connected layer for final classification:

$$\mathbf{H} = Squeeze_f(\sum_{t=1}^{T'} \mathbf{X}'_A(:,t,1)) \tag{2.5}$$

where $t \in [1, T']$ is again the time index and $Squeeze_f()$ stands for the dimension squeeze operation that removes the frequency dimension. Batch normalization and ReLU activation are applied at the end of the attention block. In addition, a dropout with rate 0.3 is applied before entering the final fully connected layer.

2.5 Experiments

2.5.1 Datasets

ESC-50 and ESC-10

ESC-50 [4] is a benchmark environmental sound dataset that has 2000 5-second-long audio clips with a sample rate of 44.1kHz, which are organized into 5 major categories: Animals; Natural soundscapes & water sounds; Human, non-speech sounds; Interior/domestic sounds; and Exterior/urban noises. These 5 categories are further divided into 50 balanced classes, with 10 classes coming from each category. ESC-10 serves as a small proof-of-concept subset of 10 classes selected from the main dataset [4]. Both datasets are pre-arranged into 5 folds and the final classification performance is measured by taking the average of all 5-fold cross-validation accuracies.

DCASE 2018 task1A and 2019 task1A

The development sets from the DCASE 2018 task1A [76] and the DCASE 2019 task1A [76] are used. Both of them contain 10-second-long audio clips with a 48kHz sampling rate from 10 classes: Airport, indoor shopping mall, metro station, pedestrian street, public square, street with medium level of traffic, travelling by a tram, travelling by a bus, travelling by an underground metro, and urban park. The DCASE 2018 task1A development set has 6122 clips for training and 2518 clips for validation, and the DCASE 2019 task1A development set has 9185 clips for training and 4185 for validation. No data augmentation was performed on these two datasets.

UrbanSound8K

This dataset contains 8732 labeled environmental urban sound recordings from 10 classes: Air conditioner, car horn, children playing, dog bark, drilling, engine idling, gun shot, jackhammer, siren, and street music [77]. These audio clips have various durations up to 4

seconds and different sample rates, and are pre-sorted into 10 folds for cross-validation.

2.5.2 Data Preparation

Feature Extraction

The log Mel-spectrograms of all audio samples from the above datasets were extracted with the number of Mel-bands is 128. For ESC and DCASE datasets, their original sample rates were kept, and for UrbanSound8K, all data samples were resampled at 44.1kHz. In addition, the deltas and delta-deltas of these log Mel-spectrograms were also computed to obtain time-dependent information. For ESC and DCASE datasets, the frame length and hop size for short-time-Fourier-transform are 1024 and 512, and for UrbanSound8K, they are 1764 and 882. Furthermore, for Urbansound8K, if an audio sample is shorter than 4 seconds, it was zero padded equally to the left and right until its duration became 4 seconds. All the features were extracted using the $librosa$ implementation. Instead of concatenating them together, these three feature maps were appended as three different channels to be the convolutional neural network inputs.

Data Augmentation

In many classification models, especially deep learning models, the amount of training data is usually required to be sufficiently large, but in many cases it is impractical to have large datasets. Additionally, a small number of training samples can also cause overfitting. In our case, the ESC datasets (ESC-50 and ESC-10) are relatively small since each class only consists of 40 samples, while the DCASE datasets are mid-sized datasets for audio classification. Data augmentation is one way that can address this issue. It is a strategy that effectively increases the amount of training data without actually collecting new data. The first kind of augmentation techniques involve applying various transformations on current available data samples. Some traditional transformations operate directly on the audio signals themselves, including time stretching and shifting, pitch changing, dynamic range

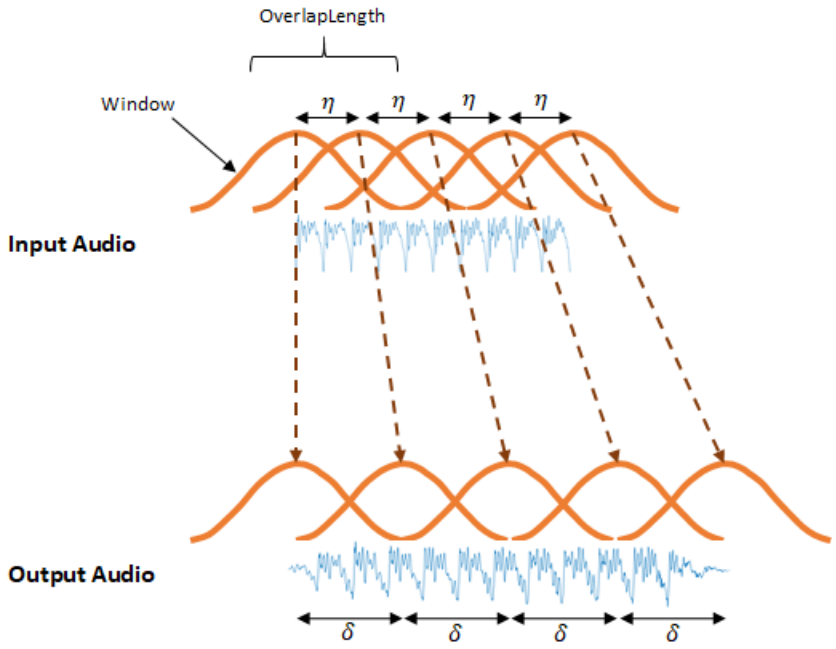

Figure 2.12: Time scale modification using the WSOLA algorithm [87]

compression, and adding random noise [78, 55, 79]. Figure 2.12 shows an illustration of time-stretching an audio signal using the WSOLA algorithm [80, 81]. Some other methods make modifications on feature maps such as spectrograms or log Mel-spectrograms. Koutini *et al.* used spectrogram rolling by shifting the spectrograms randomly over the time dimension [59]. Another approach is called SpecAugment, the idea of which is borrowed from computer vision, applies time-warping and random masking on time-frequency feature maps [82]. In addition, a method named Mixup augmentation mixes two random training samples and their corresponding labels with a given ratio sampled from beta distribution [83]. Some alternative kinds of augmentation techniques generate new data from scratch. Among them the most common methods are based on generative adversarial networks (GAN) [84]. Mun *et al.* synthesized intermediate embedding vectors instead of audio signals using GAN [85], and Kong *et al.* used SampleRNN to generate new acoustic scenes [86]. In this work, SpecAugment, Mix-up, and traditional augmentation methods such as time shifting, pitch changing, and adding background noise were explored as described later.

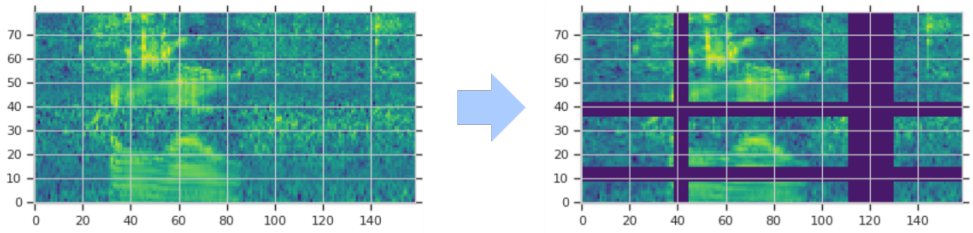

Figure 2.13: SpecAugment data augmentation by masking a spectrogram [88]

2.5.3 Single-channel Attention and Non-attention Models

To demonstrate the effectiveness of the proposed multi-channel temporal attention model, we also tested the performance of both the single-channel temporal attention model and the non-attention model using the same embedding block. Figure 2.14 shows the architectures of the upper branch in the attention block corresponding to these two models, respectively. In the single-channel case, an average pool is applied on the channel dimension of $\mathbf{X}'_S$ before the normalization. The final attention vector $\mathbf{A}_S \in \mathbb{R}^{T' \times 1}$ will be element-wisely multiplied with each $T' \times 1$ feature vector in $\mathbf{X}'_L \in \mathbb{R}^{C' \times T' \times 1}$. In the non-attention case, $\mathbf{X}'_S$ is simply divided by itself element-wisely to form a bunch of identity vectors and the effect of attention will be removed in this way. These two models also have the same number of parameters as the multi-channel model so that these three models are comparable.

2.5.4 Training the Network

We trained the network using the Adam optimizer with cross-entropy loss. The learning rate was chosen to begin with 0.001 and decayed by a factor of 0.5 if the training losses of two consecutive epochs did not decrease. The mini-batch size is 50 with the number of epochs as 50 for each of the 5-fold cross validation. The network was implemented with PyTorch and trained on NVIDIA RTX2080Ti. To get more reliable results, each model was trained 5 times and the average classification accuracies and their standard deviations were reported.

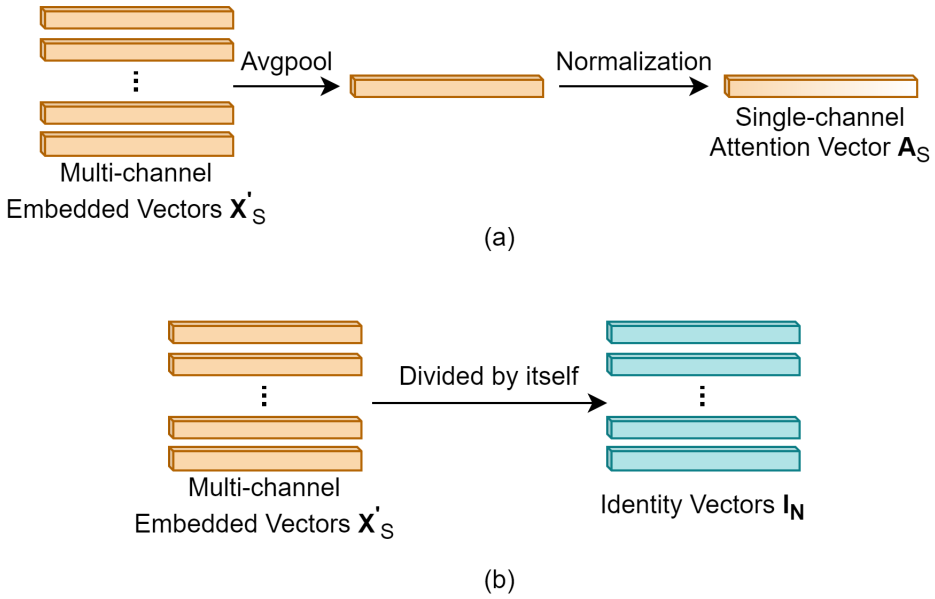

Figure 2.14: (a) The architecture of single-channel temporal attention. (b) Non-attention architecture by setting all attention weights to 1.

2.5.5 Comparison of Three Augmentation Methods

Since in the ESC datasets (ESC-50 and ESC-10), each class only has 40 samples, to reduce overfitting, data augmentation was performed on ESC datasets. To compare the performance of different augmentation methods, three popular techniques were evaluated: traditional, SpecAugment, and Mix-up. The details of these approaches are described below.

Traditional

For traditional methods, to each training sample we applied random time shifting, pitch shifting and adding Gaussian noise, resulting in 3 variants for each sample. For random time shifting, the maximum duration to be shifted was 2.5 seconds, which is half the length of an audio sample. The time shifts are only delays so that the information of some samples that only contain transient sound events at the beginning will not be eliminated. For pitch changing, the function $pitch\_shift$ of $librosa$ was used and the n_step parameter was set to be randomly chosen between -4 and 4. Finally, Gaussian noise values sampled from a standard normal distribution were added to clean audio samples after multiplying a noise

factor of 0.01. After augmentation, the number of samples in total was increased to 8000.

SpecAugment

For SpecAugment method, time warping, time masking, and frequency masking were used. For time warping, when given a feature map such as a log Mel-spectrogram with τ time steps, a random point along the time axis that goes through the center of the feature map within the time steps $(W, \tau - W)$ will be warped either to the left or to the right by a distance w, which is chosen from a uniform distribution from 0 to W [82]. For time masking, $t \in [t_0, t_0 + t)$ consecutive time steps are masked, and t is chosen from a uniform distribution from 0 to T, where T is a pre-defined time mask parameter. And t_0 is also randomly chosen from $[0, \tau - t)$ [82]. For frequency masking, $f \in [f_0, f_0 + f)$ consecutive Mel frequency bins are masked, and f is again chosen from a uniform distribution from 0 to the pre-defined frequency mask parameter F. And f_0 is randomly chosen from $[0, \mu - f)$, where μ is the number of Mel frequency bins [82]. In our experiments, the parameter W was chosen to be 8. The mask parameter T for time masking was 40, and the mask parameter F for frequency masking was 30, and 2 masks were applied for both masking types respectively. The masked parts of the feature maps were replaced by the mean of the whole masked areas.

Mixup

The mixup method is a data-agnostic augmentation technique, and according [83], it is implemented as follows:

$$\begin{aligned} \tilde{x} &= \lambda x_i + (1 - \lambda) x_j \\ \tilde{y} &= \lambda y_i + (1 - \lambda) y_j \end{aligned} \tag{2.6}$$

where x_i, x_j are input data samples randomly drawn from the training set, and y_i, y_j are their corresponding one-hot labels. The value of λ is within the range of $[0, 1]$, and it is sampled from a beta distribution with parameter α: $\lambda \sim Beta(\alpha, \alpha)$. The obtained $\tilde{x}$ and

Table 2.1: Performance of three augmentation methods: traditional, SpecAugment [82], and Mixup [83].

Augmentation Methods	ESC-50	ESC-10
Traditional	**87.05±0.18%**	**94.45±0.24%**
SpecAugment [82]	86.45±0.39%	**94.5±0.22%**
Mixup [83]	85.35±0.32%	92.65±0.30%

$\tilde{y}$ will be the new augmented data sample and label, and will be fed into the model and the loss function. In our experiments, α was chosen to be 0.2.

Results

Table 2.1 shows the performance of these three methods with 512 hidden channels. It can be observed that the traditional method has the best results as it performed better than the other two on ESC-50 and performed as well as SpecAugment on ESC-10. The Mixup method yields the lowest scores—the reason could be that, even though it has achieved some success in computer vision and natural language processing domains, environmental sound classification might not be a good fit.

2.5.6 Exploration of the Effects of Hidden Channel Number, Dropout, and Data Augmentation

To evaluate the performance of our proposed model, 5-fold cross validation was performed on the ESC-50 and ESC-10 datasets. Each model was trained 5 times and the average classification accuracies and their standard deviations were reported. Firstly, the effects of different combinations of some model design parameters, the number of hidden channels, dropout, and data augmentation, were explored. The results were shown in Table 2.2. It was observed that augmentation did help to improve the performance in most cases. Dropout also helped with the classification but only when combined with augmentation. In addition, 512 hidden channels outperformed 1024 hidden channels, indicating that more hidden channels is not necessarily beneficial. The best result was obtained with data augmentation, 512 hidden channels, and a dropout rate of 0.3.

Table 2.2: Classification accuracy and standard deviation of our model with different conditions. Aug means augmentation and hc stands for the number of hidden channels

Dropout	Aug	hc	ESC-50	ESC-10
0	No	512	84.83±0.15%	91.70±0.10%
0	Yes	512	86.31±0.31%	94.05±0.43%
0.3	No	512	84.84±0.29%	91.75±0.27%
0.3	**Yes**	**512**	**87.05±0.18%**	**94.45±0.24%**
0.3	Yes	1024	85.38±0.24%	94.05±0.43%
0.3	No	1024	84.76±0.18%	92.65±0.34%
0	Yes	1024	85.11±0.23%	93.70±0.29%
0	No	1024	84.33±0.23%	92.95±0.29%

2.5.7 Comparison of Classification Performance of Different Models

To illustrate the effect of multi-channel temporal attention, in this part, experiments were performed using MCTA, a single-channel attention model, and a non-attention model. When compared with other work, some results were reproduced by us by adapting the code available from the authors in our own experimental settings and the results were marked by "*". Table 2.3 shows the performance of our proposed model and other state-of-the-art methods including some recent attention models on the ESC datasets (ESC-50 and ESC-10) and the UrbanSound8K dataset. It was observed that for the ESC dataset, temporal attention did improve the classification results by a noticeable margin and multi-channel temporal attention performed better than single-channel attention only. For the UrbanSound8K dataset, the performance gain of the multi-channel model over the single-channel model is limited, which is probably due to less temporal feature variations across channels. When comparing with other work, our MCTA-CNN model outperformed the CNN baseline [53] by a very large margin, and outperformed the pretrained model on Audioset [50] by Kumar [56]. The FBE-ConvRBM [54] model uses filter-bank learning to extract features and applies system fusion to obtain the final results, and our model outperforms it with just a single model. Also, our model performed better than SENet [68] and CBAM [69], which have been successful in computer vision. More importantly, our model performs better on the ESC datasets than multi-stream temporal attention [24], ACRNN [72],

Table 2.3: Comparison of classification accuracy of ESC datasets (ESC-50 and ESC-10), and UrbanSound8K with other work. The $\oplus$ sign means system combination. The * denotes that the results were reproduced by us. The original results reported in [89] were 88.6%, 95.8%, and 88.5% for ESC-50, ESC-10, and UrbanSound8K respectively.

Methods/feature sets	ESC-50	ESC-10	UrbanSound8K
Piczak CNN [53]	64.50%	80.50%	73.00%
EnvNet-v2 [90]	84.90%	91.30%	78.30%
CNN pretrained on Audioset [56]	83.50%	-	-
FBEs$\oplus$ConvRBM-BANK [54]	86.50%	-	-
Multi-stream temporal attention [24]	84.00%	94.20%	-
ACRNN [72]	86.10%	93.70%	-
Channel-temporal attention [73]	85.80%	94.00%	-
SENet [68]*	81.68$\pm$0.35%	91.20$\pm$0.37%	78.80$\pm$0.28%
CBAM [69]*	85.10$\pm$0.46%	92.40$\pm$0.12%	79.63$\pm$0.32%
TS-CNN10 [89]*	85.78$\pm$0.40%	94.25$\pm$0.28%	**80.71$\pm$0.20**%
Non-attention (ours)	81.74$\pm$0.56%	90.95$\pm$0.40%	78.64$\pm$0.12%
Single-channel (ours)	85.24$\pm$0.12%	93.15$\pm$0.20%	79.44$\pm$0.24%
MCTA-CNN (ours)	**87.05$\pm$0.18**%	**94.45$\pm$0.24**%	**79.78$\pm$0.27**%

channel-temporal attention network [72], and TS-CNN10 [89], which all represent temporal attention using only one single vector. It is worth mentioning that for the UrbanSound8K dataset, our model was slightly outperformed by TS-CNN10 [89]. This model consists of 4 convolutional blocks with 4 separate parallel temporal-spectral attention modules, making the number of parameters significantly bigger (4.98M) than our model (1.47M). While our model still has a close performance.

Table 2.4: Comparison of classification accuracy of DCASE 2018 task1A and DCASE 2019 task1A datasets with other work. The * denotes that the results were reproduced by us. In [89] the original results were 68.7% and 70.6% for DCASE 2018 and 2019 respectively. In [57], the original result for DCASE 2018 was 72.7% and no DCASE 2019 result was given.

Methods	DCASE2018 1A	DCASE2019 1A
Self-attention [91]	70.81%	-
Atrous-CNN [57]*	69.07$\pm$0.59%	69.24$\pm$0.99%
TS-CNN10 [89]*	69.68$\pm$0.60%	69.59$\pm$0.68%
Non-attention(ours)	70.98$\pm$0.66%	74.60$\pm$0.32%
Single-channel(ours)	71.92$\pm$0.75%	74.88$\pm$0.40%
MCTA-CNN (ours)	**72.40$\pm$0.38**%	**75.71$\pm$0.28**%

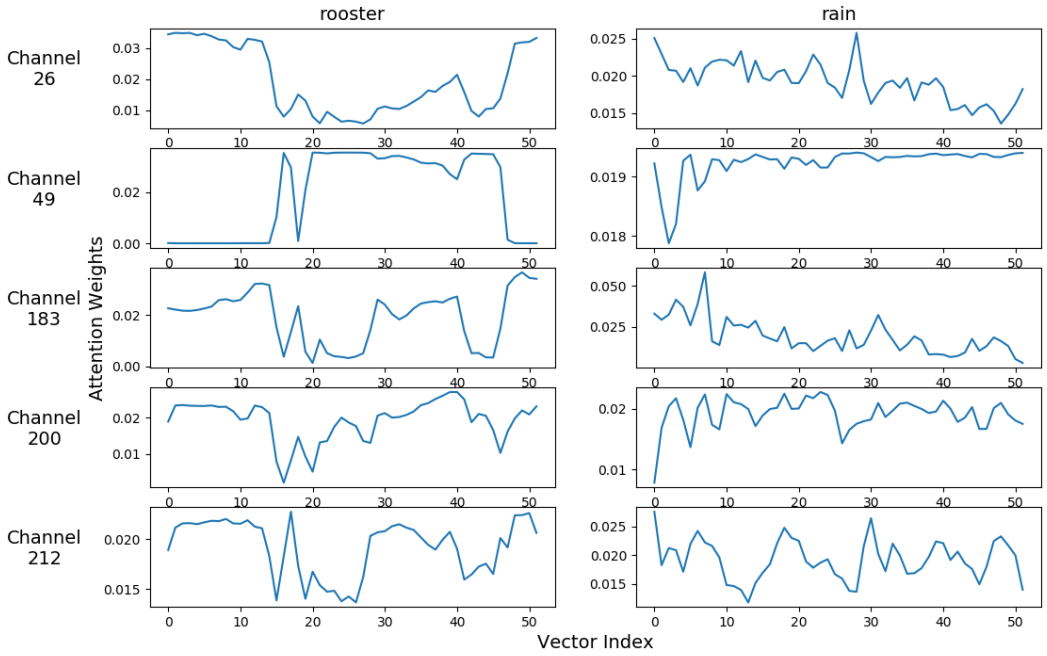

Figure 2.15: 5 random attention vectors for rooster and rain.

Our model was also tested on the two DCASE acoustic scene classification (ASC) datasets described above, and the results are shown in Table 2.4. It is shown that MCTA can also improve the performance of ASC as well. Our model outperformed several popular attention-based CNN models including the self-attention model [91] and the Atrous-CNN model [57], which were both initially designed for acoustic scene classification. Again, as mentioned above, the number of parameters of our model is 1.47M, which is much smaller than TS-CNN10 [89] (4.98M) and Atrous-CNN [57] (4.36M).

Figure 2.15 shows the attention vectors from 5 random channels (out of the 512 total in $\mathbf{A} \in \mathbb{R}^{512\times52\times1}$) for two classes in ESC-50, namely rooster and rain. It can be seen that within each class, the attention vectors from different channels can be very different, both in shape and magnitude, which further supports our proposition that a unique attention vector for each channel is beneficial. Furthermore, some confusion matrix examples for fold 3 of ESC-50, ESC-10, UrbanSound8K datasets, along with DCASE 2018 and DCASE 2019 datasets on our multi-channel temporal attention (MCTA) CNN model are shown in **APPENDIX A**. **This work was submitted to and published by ICASSP 2021 [92]**.

CHAPTER 3
PROTOTYPICAL NETWORKS WITH HYBRID ATTENTION FOR FEW-SHOT AUDIO CLASSIFICATION

This chapter introduces the details of the hybrid-attentional prototypical networks model. Firstly, some background and motivations of few-shot learning are described, including some related work and the learning strategy. Then, our proposed model is presented and its performance was evaluated on ESC-50, *noise*ESC-50, and a subset of FSD50K dataset containing environmental sound classes.

3.1 Few-Shot Learning

3.1.1 Introduction

Deep learning techniques are usually highly data dependent. When the data set is small or labeled training samples are lacking, it usually leads to overfitting and results in poor generalization performance. To tackle this problem, many new techniques have been proposed, such as weakly supervised learning, transfer learning, meta-learning, and few-shot learning (FSL). FSL is a type of machine learning scenario where the prediction is made based on a very small amount of data samples [93]. The motivations of the development of few-shot learning include: 1) In many cases, the amount of supervised training data is lacking, therefore causing many deep learning techniques to fail; 2) Sometimes a large but unlabelled dataset is available, but labelling these data can cost too much human effort and computational resource; 3) Human can learn with only a few examples, and some FSL algorithms are developed to imitate how human learns new things. Specifically, when there is only one supervised training sample within each class, few-shot learning then becomes one-shot learning [94, 95, 96]. Sometimes there is even no example with supervised

information, then FSL becomes zero-shot learning [97, 98].

According to [99], a few-shot learning problem can be approached from three different perspectives: *data*, *model*, and *algorithm*. *Data* methods increase the amount of training data through data augmentation or generative adversarial network (GAN) [84]. *Model* methods seek to constrain the hypothesis space using prior knowledge, including strategies such as multitask learning [100, 101], embedding learning [102, 103], learning with external memory [104, 105], and generative modeling [95, 106]. *Algorithm* methods [107, 108] make use of prior information to find better searching strategies for parameters by starting with a good initialization or guiding the search steps [99]. Among all kinds of methods mentioned above, embedding learning strategies from *Model* methods have become the most widely used in audio recognition and classification. Embedding learning algorithms [102, 109] embed data samples to a feature space equipped with a distance metric such as cosine or euclidean distance so that samples that belong to the same class are closer to each other, while samples from different classes can be more easily distinguished [99]. Some popular metric-based embedding learning include convolutional siamese neural network [110], relation network [111], matching nets [112], and prototypical networks [113]. The key idea of these methods is to learn a function or a network that can decently measure the similarity between samples. For example, matching nets (shown in Figure 3.1) embed support samples and query samples using different functions, and then compute the cosine similarities between the query embedding with each support embedding [114]. It also introduced a new type of training unit called "episode", within which training is performed in the same way as testing. Prototypical networks model also made use of episodic training and its performance has been superior over other related models so far. Therefore, in this work, the prototypical networks model is used and its details are described below.

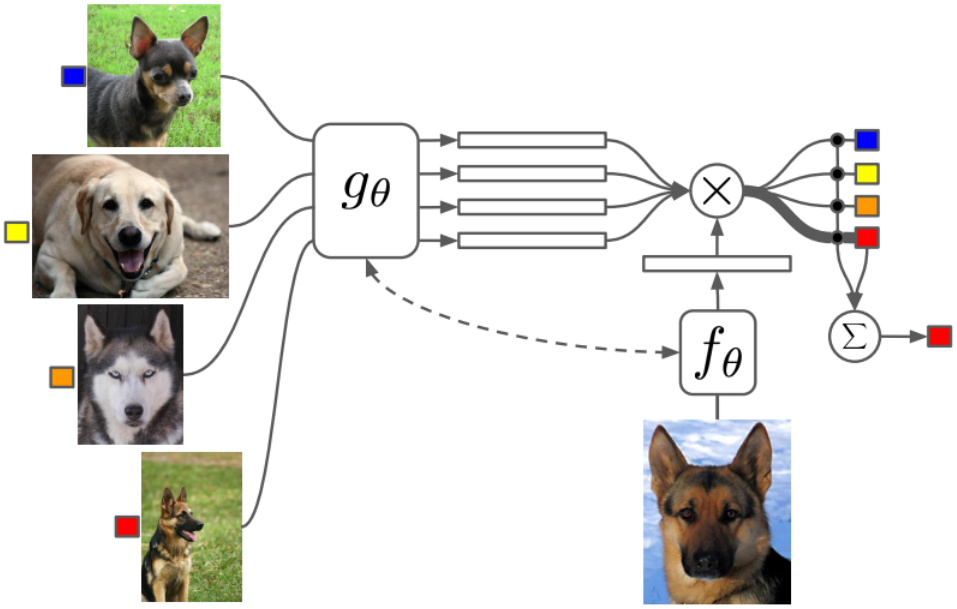

Figure 3.1: Matching Networks architecture [112].

3.1.2 Prototypical Networks

In prototypical networks, the training process is consisted of numerous episodes. We are given a training dataset $D = \{(\mathbf{x_i}, y_i)|i = 1, ..., N\}$ with N total samples, $\mathbf{x_i}$, each with an associated label, $y_i \in \{1, ..., C\}$, where C is the total number of classes in the set. In each episode, a subset $S = \{S_k|k = 1, ..., K\}$ that contains K randomly selected classes without replacement is formed, where S_k is consisted of samples that belong to class k. This subset is called a support set and for each chosen class k, N_k samples are picked randomly without replacement as well, making the total number of samples in the support set $K \times N_k$. After that, a disjoint set called query set Q with $K \times N_q$ samples is then selected, where each class contains N_q samples. Therefore, in each training episode, a $K\text{-}way\ N_k\text{-}shot$ problem is set up to be solved. The testing process acts in the same manner as training, but with a different dataset with unseen classes. The goal is to classify query samples to unknown labels given a small support set within the testing dataset.

The key idea of prototypical networks is the computation of an M-dimensional proto-

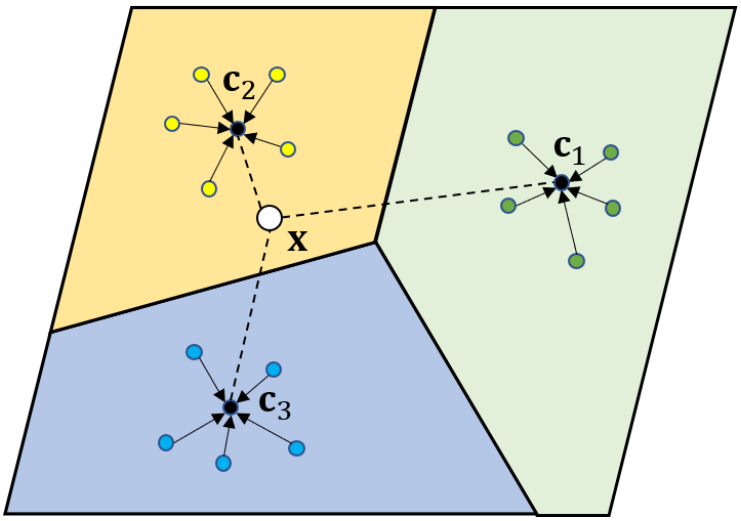

Figure 3.2: Prototype Networks for few-shot learning.

type feature vector $\mathbf{c}_k \in \mathbb{R}^M$ as the mean of embedded feature vectors of support samples within a class.

$$\mathbf{c}_k = \frac{1}{N_k} \sum_{i=1}^{N_k} f_\theta(\mathbf{x}_i^k), \quad k \in \{1, ..., K\} \tag{3.1}$$

where f_θ denotes the embedding function with learnable parameters θ which is typically a neural network. Given a distance metric $d : \mathrm{R}^M \times \mathrm{R}^M \rightarrow [0, +\infty)$, the posterior probability that a query sample $\mathbf{x} \in Q$ belongs to a class k is based on a softmax over the inverse of distances between the query sample embedding and the prototype vector of class k.

$$P_\theta(y = k|\mathbf{x}) = \frac{\exp(-d(f_\theta(\mathbf{x}), \mathbf{c}_k))}{\sum_{k'} \exp(-d(f_\theta(\mathbf{x}), \mathbf{c}_{k'}))} \tag{3.2}$$

In the original paper [113], the squared euclidean distance was used and the loss function is the negative log-likelihood: $L(\theta) = -logP_\theta(y = k|\mathbf{x})$. Figure 3.2 shows the illustration of the prototypical networks for a 3-way 5-shot few-shot learning scenario.

3.1.3 Learning Strategy

Few-shot learning methods such as prototypical networks belong to a broader learning concept called meta-learning. Instead of training on the training set and generalizing on

the test set, which is the main objective in traditional supervised learning scenarios, the core idea is ”Learning to learn”. Figure 3.3 illustrates how a 2-way 4-shot few-shot image classification model is learned. Just like normal supervised learning process, there is also a training phase and testing phase. However, the main difference is that in normal supervised learning, the training set and the testing set usually contain the same list of classes, while in few-shot learning the classes in the testing set typically have never been seen during the training phase. In this example, for each training episode or minibatch, 2 classes, namely cats and birds are randomly chosen (2-way) and 4 samples within each class are randomly picked (4-shot), which forms a small sample set called support set. In addition, one or several query samples (the query set) are drawn from the remaining samples in these 2 classes. In this example, one query sample, a bird image, is drawn in the first episode, and during the training the model learns to assign the correct label ”bird” to this query sample image. Then in the next episode, another 2 classes, flowers and bikes are chosen (choosing overlapping classes with the previous episode is allowed), and another support set is formed. Similarly, a query sample with the ground truth label ”flowers” is picked, and the models again learns to assign the correct label to it.

After numerous episodes of training, the trained model is the evaluated on a separate testing dataset. As mentioned previously, the testing set contains a completely different set of classes. As is shown in Figure 3.3, during testing, the exact same process happens as in the training, where a support set (containing 4 dog samples and 4 otter samples) is formed and a query sample with an unknown label is fed into the model to compute the prediction. This learning strategy is called ”training in the same way as testing” [115], one of the basic ideas of meta-learning, which few-shot classification belongs to.

3.2 Few-Shot Learning with Hybrid Attention

As described in **CHAPTER 2**, few-shot learning models are used to classify unseen labels with few training samples. And metric-learning or meta-learning approaches are becom-

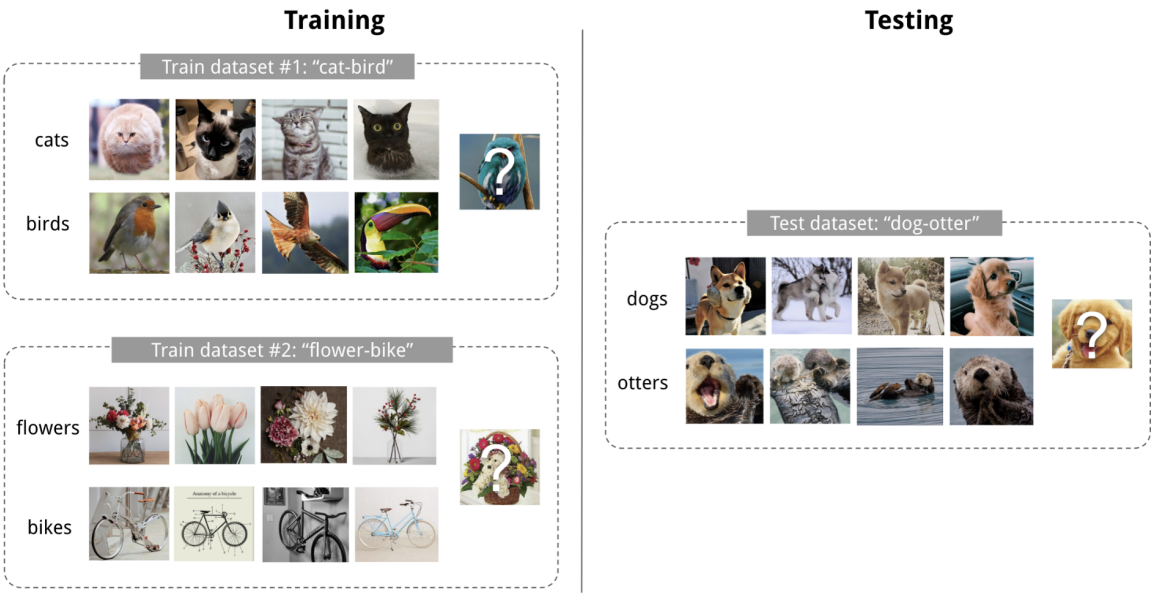

Figure 3.3: An example of 2-way 4-shot classification [115]

ing increasingly popular in recent years. To our knowledge, most of the research done using few-shot learning models focuses mainly on computer vision and natural language processing, whereas little work has been done for audio classification. In this work, the prototypical networks model is chosen among all metric-learning models. This model is simple yet effective, and has achieved the state-of-art results on several few-shot learning benchmarks [116]. However, as a metric-based method, the performance often degrades in the presence of bad or noisy embedded features, and outliers in support instances. To solve this problem, a hybrid attention module is introduced in this work and Figure 3.4 shows the overall architecture of the whole model. The architecture consists of three major parts: a backbone CNN network as a feature encoder, a feature-level attention block to emphasize key feature regions, and an instance-level attention block to focus on crucial support instances and diminish the effect of outliers. These three parts are combined together and integrated into the prototypical networks few-shot learning framework.

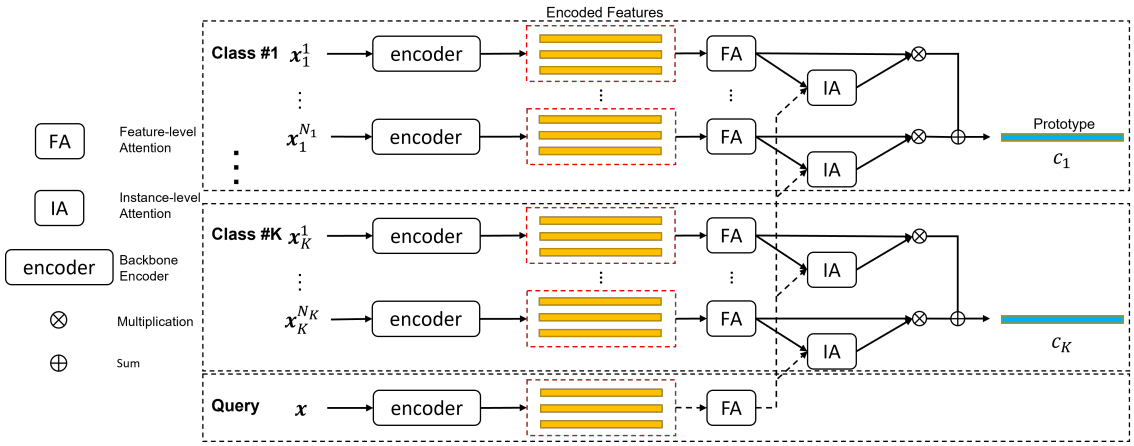

Figure 3.4: The architecture of our proposed model.

3.3 Backbone Network

Our backbone CNN network has a similar structure as the embedding block in our MCTA model. It contains 3 basic blocks and each block consists of a 3×3 convolutional layer, batch normalization, ReLU activation, and a max pooling layer consecutively. The kernel sizes for these 3 max pooling layers are 8×2, 8×2, and 2×1, respectively. The numbers of channels for these 3 convolutional layers are 128, 256, and 384. Figure 3.5 shows the details of the backbone network structure. Similarly as in subsection 2.4.2, the numbers in the brackets after "Conv. block" indicate the input and output numbers of filters of the convolutional layer. And the numbers after "Max-pooling" and "Padding" stand for kernel sizes. The numbers after "Conv." are the numbers of filters and the kernel size. Again BN stands for batch normalization and ReLU stands for Rectified Linear Units as the activation function.

3.4 Feature-level Attention

Since each class in the support set only has a few samples and the prototypes are solely determined by the features extracted from these samples, the performance of prototypical

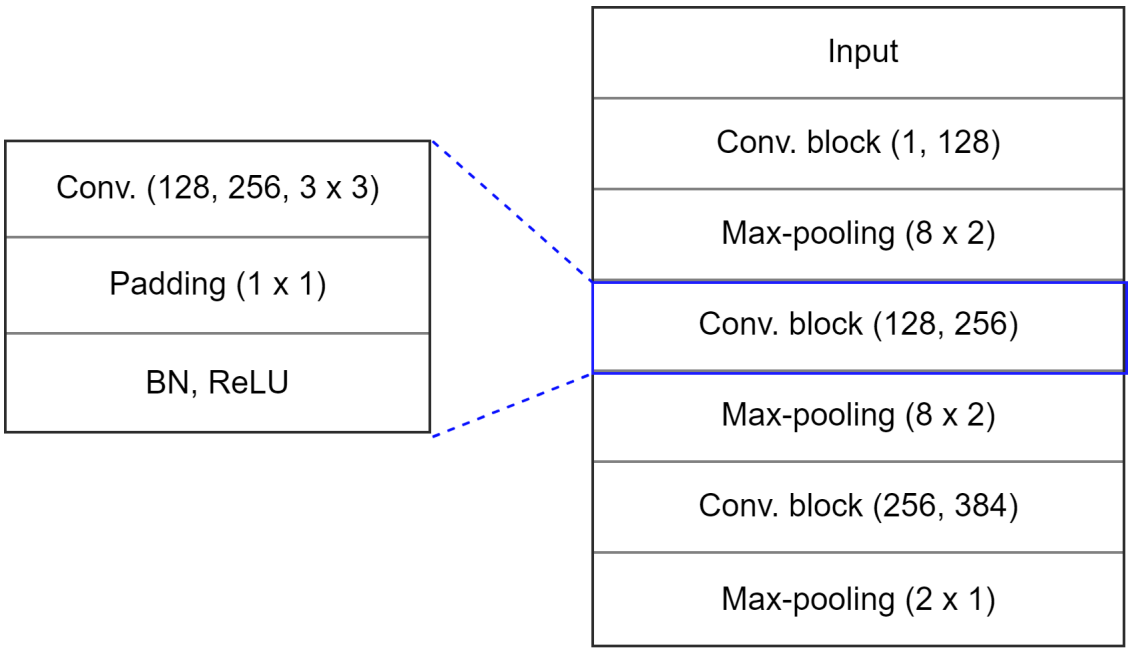

Figure 3.5: Details of the backbone network.

networks is highly dependent on the discriminative power of the encoded feature vectors. There are certain parts of the features that contain more relevant information than others, and the feature-level attention block is introduced to focus on those parts. As described in the previous chapter, the attention block from the proposed MCTA model is introduced into the embedding function f_θ with a small modification by changing the standard normalization in the attention branch to *softmax* normalization (Figure 3.6). By focusing on more informative features using attention, this feature-level attention block can help the model to better determine prototype vectors.

3.5 Instance-level Attention

In the original prototypical networks, the prototypes are computed by averaging all support instance embeddings within each class, which means it is assumed that the contribution of each support instance is equal. However, as shown in Figure 3.8(a), if one of the support instances is an outlier or is much farther away from the query sample than the other instances, it can cause a deviation on the resultant prototype and thereby causing incorrect

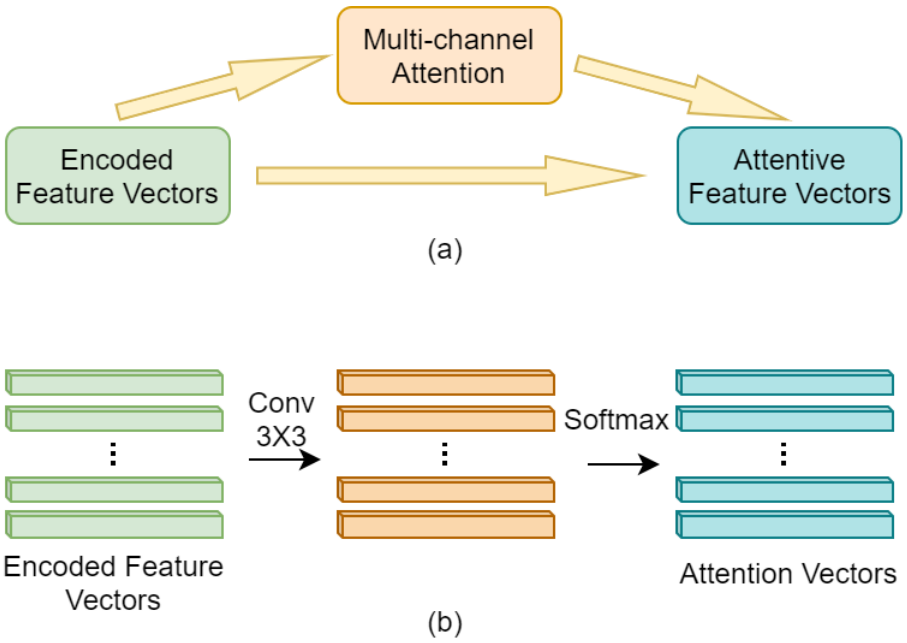

Figure 3.6: (a) Block diagram the feature-level attention block. (b) Structure of the multi-channel attention.

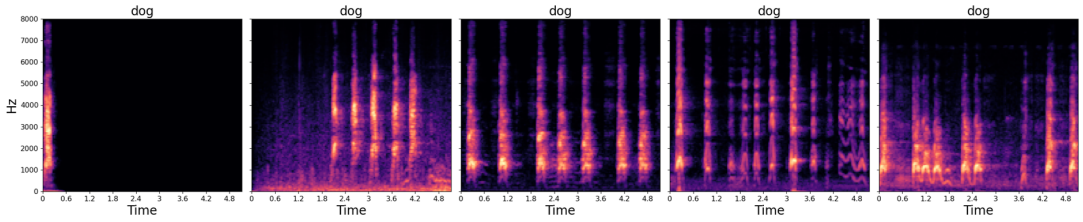

Figure 3.7: A support set of dog sound log Mel-spectrograms with 5 samples.

classification. For example, Figure 3.7 shows a potential support set that contains 5 log Mel-spectrograms of dog sound. It can be clearly seen that the first sample is very different from the other four, and it can then be considered as an outlier. By simply averaging embeddings of these 5 samples, we might end up getting a deviated prototype for class "dog".

Therefore, inspired by [116], we propose to introduce an instance-level attention block to alleviate this problem. The core idea is to generate an attention score for each support instance based on its relationship with the query sample and replace Equation 3.1 with a

weighted average:

$$\mathbf{c}_k = \sum_{i=1}^{N_k} \beta_i^k f_\theta(\mathbf{x}_i^k) \tag{3.3}$$

where β_i^k is the score assigned to a support instance in class k, and it is modelled as follows:

$$\beta_i^k = \frac{e_i^k}{\sum_{n=1}^{N_k} e_n^k} \tag{3.4}$$

$$e_i^k = sum\{\sigma(f_\phi(f_\theta(\mathbf{x}_i^k)) \circ f_\phi(f_\theta(\mathbf{x})))\} \tag{3.5}$$

where $f_\phi(\cdot)$ is a fully connected layer, $\sigma(\cdot)$ is $sigmoid$ function, and $sum\{\cdot\}$ is the summation over all elements of a vector. The purpose of Equation 3.5 is to compute the similarity between a support sample and the query sample, therefore it can be seen from Equation 3.4 and Equation 3.5 that the weight of a support sample is query-dependent, meaning that the contribution of each support sample depends on the incoming query sample instead of having an independent "quality." Moreover, the reason why we choose standard normalization over $softmax$ in this case is that $softmax$ will dramatically enlarge the difference between the weights of the support samples and may result in overly biased prototypes. As an illustration shown in Figure 3.8 (b), by assigning lower scores to bad instances, the model can become more capable of assigning the correct label to the query sample.

3.6 Experiments

3.6.1 Datasets

Our model was evaluated on two datasets: ESC-50 dataset and *noise*ESC-50 dataset. The details of ESC-50 dataset were described in **CHAPTER 2** and the *noise*ESC-50 dataset was created in [117] by mixing the clean ESC-50 samples with random recordings from 15 acoustic scenes of DCASE2016 dataset [118] as additive background noise. The mixture generation procedure followed the one described in [52], where the event-to-background

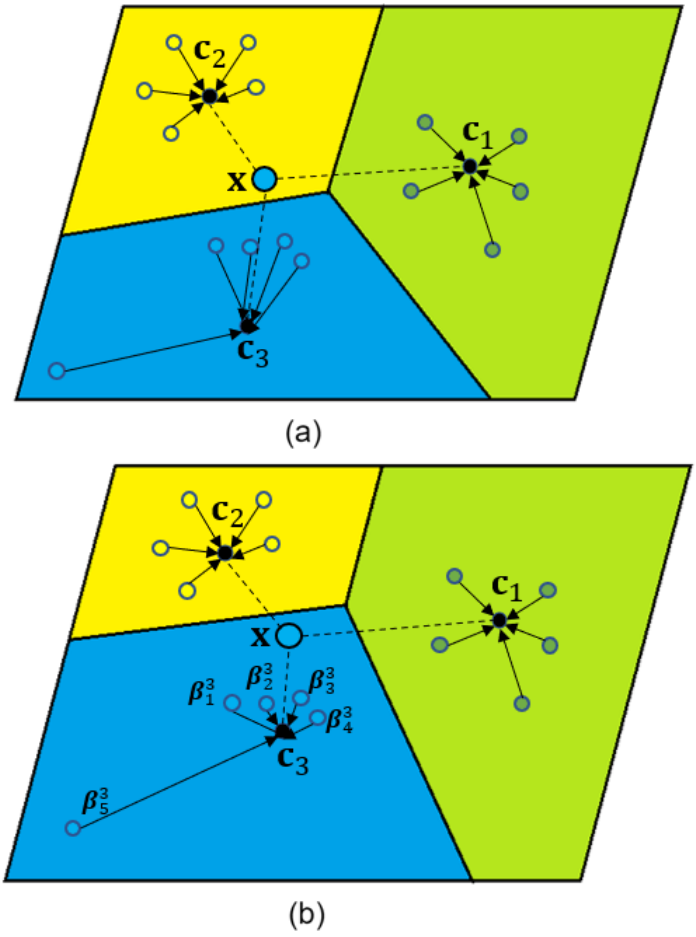

Figure 3.8: (a) Prototypical networks without instance-level attention. (b) Prototypical networks with instance-level attention.

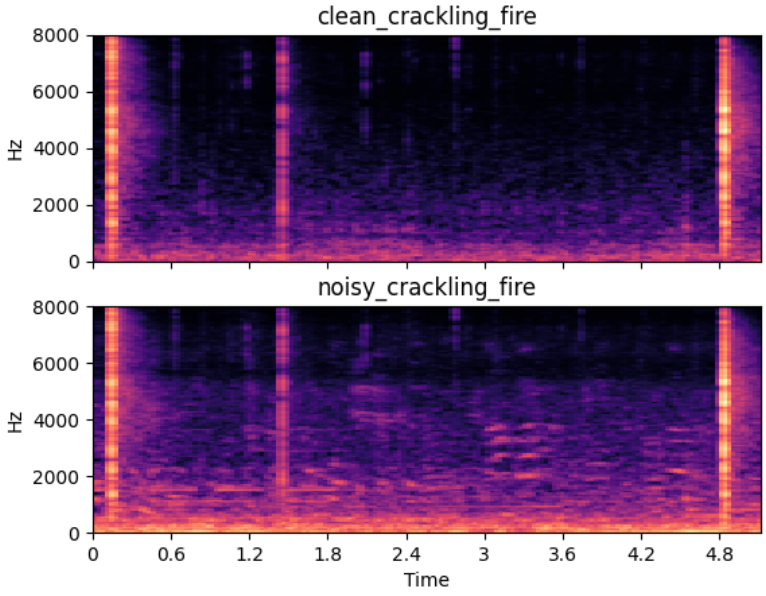

Figure 3.9: Log Mel-spectrograms of a "crackling_fire" sound and its corresponding noisy counterpart.

ratios (EBR) were randomly selected from -6, 0, and 6 dB. The EBR was defined as a ratio of average RMSE values between a foreground clean audio signal and a background segment with the same duration [52]. As a result, each clean data sample in ESC-50 has a corresponding noisy counterpart in *noise*ESC-50, and the noisy signals can better represent everyday sound environment [117]. Figure 3.9, Figure 3.10, and Figure 3.11 show examples of clean and noisy "crackling_fire", "pouring_water", and "crickets" sounds. The relatively small size of ESC-50 makes it a good candidate for few-shot sound classification.

3.6.2 Initial Experimental Setup

By using the same splitting strategy as in [117], we randomly selected 35 classes for 10-way 5-shot training, 5 classes for 5-way 5-shot validation, and the remaining 10 classes for testing. During testing, the results for 5-way 1-shot, 5-way 5-shot, 10-way 1-shot, and 10-way 5-shot scenarios were obtained. Also according to [117], for both ESC-50 and *noise*ESC-50, to speed up the training process and save memory, the audio clips were firstly downsampled from 44.1kHz to 16kHz, and then log Mel-spectrograms with 128 Mel bins

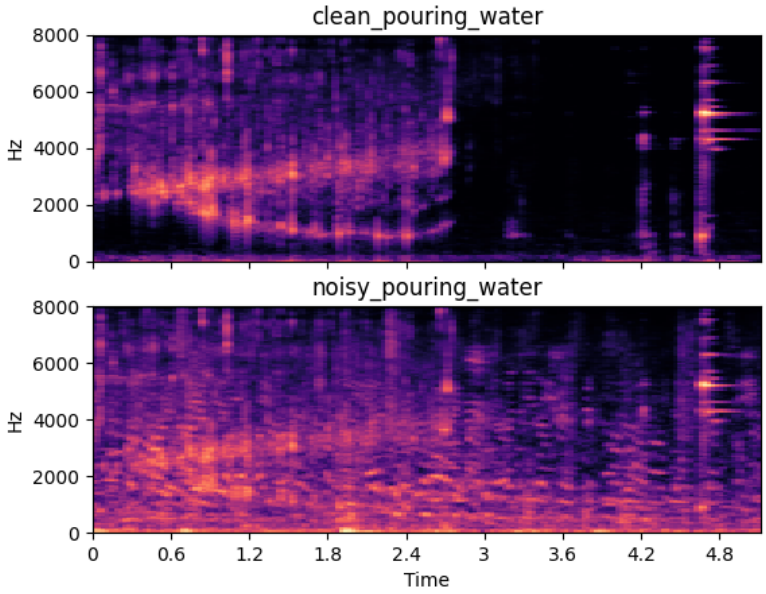

Figure 3.10: Log Mel-spectrograms of a "pouring_water" sound and its corresponding noisy counterpart.

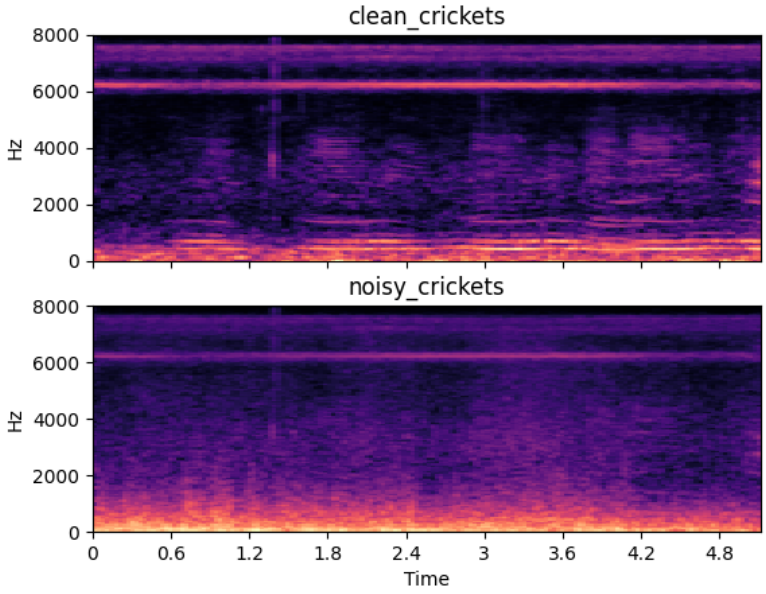

Figure 3.11: Log Mel-spectrograms of a "crickets" sound and its corresponding noisy counterpart.

were extracted. The window size is 2048 and the hopsize is 497, resulting in a 128×160 feature map for each data sample. The input features were then z-score normalized using the mean and standard deviation of the training set before being fed into the model. We trained the network using Adam with cross-entropy loss and a starting learning rate as 0.001. The total number of epoch is 60, and the learning rate is decayed by a factor of 10 after every 20 epochs. We also used a 1e-4 weight decay as in [117]. All experiments were run on an NVIDIA RTX 2080Ti GPU.

3.6.3 Results

The few-shot audio classification accuracies for the validation and test sets of ESC-50 and *noise*ESC-50 are shown in Table 3.1 and Table 3.2 respectively. For each dataset, our model was run 10 times and the average accuracy is reported along with the 95% confidence intervals. For each run, the best model was selected based on the 5-way 5-shot accuracies of the 5 validation classes, and it was then applied to the test set. Proto-Net means the original model of prototypical networks. Proto-IA denotes prototypical networks with the instance-level block only. Proto-FA stands for prototypical networks with the feature-level attention block only, and Proto-HA stands for prototypical networks with the hybrid attention module including both the feature-level and the instance-level blocks. Since the instance-level attention assigns weights to multiple support samples, we only focus on the 5-shot cases when comparing the performances involving instance-level and hybrid attention.

Table 3.1 compares the performance of prototypical networks with or without attention on ESC-50. It can be seen that by adding the feature-level attention block, there is a significant improvement in all scenarios for both the validation set and the test set. Especially for 10-way 1-shot and 10-way 5-shot cases in the test set, where the performance gain is 9.25% and 7.48% respectively, which indicates that our multi-channel temporal attention block is capable of highlighting key features to make data samples more distinguishable. However, it is worth noticing that with only the instance-level attention block, the validation

performance is similar to the original model while the test results are even a little worse. This could be due to the fact that without any feature-level attention, the embedded feature vectors are not very food representations of the data samples. Therefore, only applying instance-level attention won't be advantageous as expected. Moreover, the hybrid attentional prototypical networks perform similarly as Proto-FA in validation, and perform best in 5-shot test scenarios, showing that when applied to the embedded representations with feature-level attention, the instance-level attention block is able to focus on crucial support samples when computing the prototypes. Note, that in the 1-shot case, the instance-level attention block performs no function since there is only a single support sample and its weight will always be 1. Therefore 1-shot results for Proto-HA are not reported.

The results for *noise*ESC-50 are shown in Table 3.2. For *noise*ESC-50, the Proto-FA model enhances the performance and the improvement is even greater than the performance gain of Proto-FA with ESC-50. This suggests that emphasizing relevant features can largely diminish the effect of noise. Also, as discussed above, for the same reason, worse test results are observed with instance-level attention only compared to Proto-Net. Similarly, when using the Proto-HA model, the performance is further enhanced for the 5-shot cases, showing that in the noisy setting, the instance-level attention module is still able to lessen the contribution of those bad support instances. It should be noted that the improvement provided by the Proto-HA model for the test 10-way 5-shot case on *noise*ESC-50 is less than the the improvement seen on ESC-50. The reason for this might be that when there are good and bad support instances, the instance-level attention can assign higher weights to those good ones, but when all the support and query samples are degraded, the advantage might not be as big. **This work was submitted and published by ICASSP 2022 [119].**

3.7 Modify the Instance-level Attention

3.7.1 Model Modification and Experimental Setup

In the previous experiments, it was observed that by adding only the instance-level attention, there was no classification performance improvement comparing to the original prototypical networks model. Therefore, we tried to make a slight modification on how e_i^k, the similarity between a support sample and the query smaple, is calculated (Equation 3.5). This time, the $sigmoid$ activation is applied to the embeddings $f_\phi(f_\theta(\mathbf{x}_i^k))$ and $f_\phi(f_\theta(\mathbf{x}))$ respectively instead of just applying it to their product. We felt that in this way the similarities could be more properly obtained. The modified Equation 3.5 is shown below:

$$e_i^k = sum\{\sigma(f_\phi(f_\theta(\mathbf{x}_i^k))) \circ \sigma(f_\phi(f_\theta(\mathbf{x})))\} \tag{3.6}$$

In addition, to acquire more reliable results, 10 classes were chosen as validation set instead of 5 in order to match the 10-way 5-shot training, resulting in the number of training classes down to 30 from 35. Moreover, each model was run 20 times instead of 10 to get more stable results, and the experiments were run on the PACE cluster of Georgia Tech using a Quadro RTX 6000 GPU with other hyperparameters the same as previous experiments.

3.7.2 Results

The results for ESC-50 and *noise*ESC-50 with the modified model are shown in Table 3.3 and Table 3.4, respectively. It can be observed that with the modified model and the new experimental setups, the Proto-HA model still had the highest overall classification accuracies. Despite the similar 5-way 5-shot performance between Proto-HA and Proto-FA, noticeable improvement can be seen in all 10-way 5-shot cases. Most notably, the modified Proto-IA model outperformed the original prototypical networks this time, especially on the validation set. This demonstrates that this modification (Equation 3.6) can better cal-

Table 3.1: Few-shot sound classification accuracies for prototypical networks with or without attention module on ESC-50.

Model	Validation	Test			
	5-way 5-shot	5-way 1-shot	5-way 5-shot	10-way 1-shot	10-way 5-shot
Proto-Net	86.65±0.88%	64.40±1.38%	83.83±0.92%	47.83±1.10%	71.00±1.04%
Proto-IA (ours)	86.95±0.82%	–	82.68±0.28%	–	69.33±0.58%
Proto-FA (ours)	**91.35±0.57%**	**71.18±1.23%**	89.60±1.08%	**57.08±1.43%**	78.48±1.51%
Proto-HA (ours)	**91.20±0.62%**	–	**90.35±0.83%**	–	**80.08±1.31%**

Table 3.2: Few-shot sound classification accuracies for prototypical networks with or without attention module on *noise*ESC-50.

Model	Validation	Test			
	5-way 5-shot	5-way 1-shot	5-way 5-shot	10-way 1-shot	10-way 5-shot
Proto-Net	82.7±0.48%	61.53±0.40%	81.03±0.57%	45.90±0.28%	64.98±0.52%
Proto-IA (ours)	82.75±0.75%	–	80.48±0.66%	–	64.35±1.08%
Proto-FA (ours)	90.35±0.72%	**71.35±0.90%**	88.00±0.63%	**56.55±1.22%**	78.55±0.75%
Proto-HA (ours)	**90.65±1.27%**	–	**88.78±0.45%**	–	**79.08±1.12%**

culate the similarity between support and query samples, and therefore resulting in better instance weights.

Furthermore, as described in **Section 3.5**, similar as outliers, noisy samples among clean ones can also affect the computation of prototypes. Therefore, to further demonstrate that the instance-level attention module is capable of diminishing the effect of bad or noisy support samples, it is tempting to try our models on mixed support samples where a few of them are noisy and the others are clean.

3.8 Training with Mixed Support Samples

3.8.1 Experimental Setup

A new set of experiments was run on the clean ESC-50 dataset, but within every 5 clean support samples of a class, 2 of them were randomly picked and replaced by their noisy counterparts from *noise*ESC-50. In this setting, unlike using all noisy data samples, the 2 noisy support samples are clearly considered bad ones or outliers against the other 3. The trained models were then tested on the validation and test sets from *noise*ESC-50 dataset. It was expected that with this training strategy, the performance gain by adding the instance-level attention module would increase comparing to previous experiments.

3.8.2 Results

The results of this mixed training experiment are shown in Table 3.5. Again, each model was run 20 times and the average classification accuracy and the corresponding 95% confidence intervals are reported. It can be clearly observed that when comparing Proto-IA with Proto-Net, and Proto-HA with Proto-FA, the performance gain on all 5-shot cases are much bigger than Table 3.4. And all 10-way 5-shot results for the validation and test sets increased by over 1%, and the test 5-way 5-shot results increased by over 0.5%. These results further prove that the instance-level attention is able to assign higher weights to good support samples and lower weights to bad ones.

Table 3.3: Few-shot sound classification accuracies for prototypical networks with or without attention module on ESC-50.

Model	Validation	Test			
	10-way 5-shot	5-way 1-shot	5-way 5-shot	10-way 1-shot	10-way 5-shot
Proto-Net	72.99±0.38%	63.65±0.88%	82.86±0.66%	46.43±0.85%	69.59±0.95%
Proto-IA (ours)	73.96±0.41%	–	83.14±0.53%	–	70.44±0.82%
Proto-FA (ours)	80.66±0.52%	**71.95±0.91%**	**89.28±0.50%**	**56.98±0.97%**	78.28±1.03%
Proto-HA (ours)	**81.30±0.39%**	–	**89.25±0.49%**	–	**79.26±0.80%**

Table 3.4: Few-shot sound classification accuracies for prototypical networks with or without attention module on *noise*ESC-50.

Model	Validation	Test			
	10-way 5-shot	5-way 1-shot	5-way 5-shot	10-way 1-shot	10-way 5-shot
Proto-Net	69.08±0.50%	61.06±0.68%	81.21±0.64%	45.24±0.62%	65.74±0.61%
Proto-IA (ours)	69.93±0.53%	–	81.26±0.48%	–	66.05±0.56%
Proto-FA (ours)	78.16±0.50%	**71.40±0.73%**	**87.53±0.47%**	**56.48±0.80%**	77.51±0.65%
Proto-HA (ours)	**78.93±0.58%**	–	**87.65±0.43%**	–	**78.10±0.66%**

Table 3.5: Few-shot sound classification accuracies for prototypical networks with or without attention module with mixed training support samples on *noise*ESC-50.

Model	Validation	Test			
	10-way 5-shot	5-way 1-shot	5-way 5-shot	10-way 1-shot	10-way 5-shot
Proto-Net	68.20±0.47%	59.51±0.66%	80.48±0.72%	43.20±0.70%	65.50±0.82%
Proto-IA (ours)	69.63±0.63%	–	81.01±0.57%	–	66.80±0.72%
Proto-FA (ours)	77.18±0.61%	**69.39±0.91%**	87.26±0.55%	**55.55±1.06%**	76.15±0.87%
Proto-HA (ours)	**78.21±0.72%**	–	**87.78±0.52%**	–	**77.29±0.67%**

3.9 Testing the model on a subset of FSD50K

In addition to the ESC-50 and *noise*ESC-50 datasets, in order to evaluate how our models would perform on other datasets. We created a larger environmental sound dataset drawn from the FSD50K dataset, and the details are described in this section.

3.9.1 Dataset

FSD50K [120] is an open human-labeled sound event dataset containing 51,197 Freesound clips drawn from the Audioset Ontology [50]. These mono audio clips have a total duration of 108.3 hours and are unequally distributed in 200 hierarchically organized classes, which consist of 144 leaf nodes and 56 intermediate nodes. The sound events are primarily produced by physical sound sources and production mechanisms [120], and the main categories include human sounds, sounds of things, animals, natural sounds, musical instruments and so on. Clips have various length between 0.3s and 30s, and all of them have a sample rate of 44.1kHz.

The dataset we used is a subset of FSD50K, where we extracted all environmental sound classes which are not human or musical instrument sounds from the 144 leaf nodes, resulting in a total of 108 classes. Afterwards, we further removed all clips that contain multiple leaf node labels, and ended up with totally 16,923 single-label recordings, and we call this new subset Env-FSD50K. Figure 3.12 to Figure 3.15 show some statistics and distributions of the created dataset.

Then, to generate training samples, we further downsampled the data to 16kHz and processed them into 5-second segments. If the original duration of a clip is shorter than 5 seconds, it was zero padded to 5 seconds long. And if originally an audio is longer than 5 seconds, it was split into 5-second segments and if the length of the remaining segment is shorter than 2 seconds, it was discarded, otherwise it was zero padded to become another 5-second segment. For example, if an audio clip is 21 seconds long, it will be split into 4

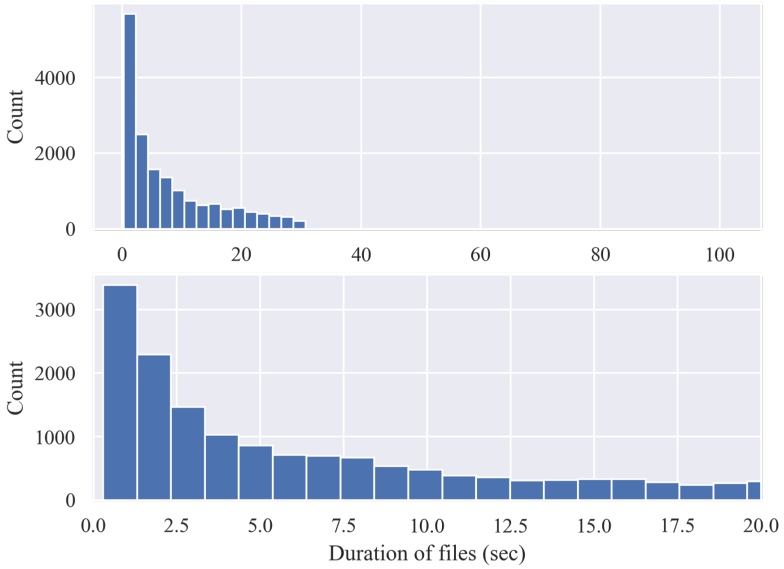

Figure 3.12: The number of files for different durations in seconds. The second row is a zoom-in version of the duration range from 0s to 20s.

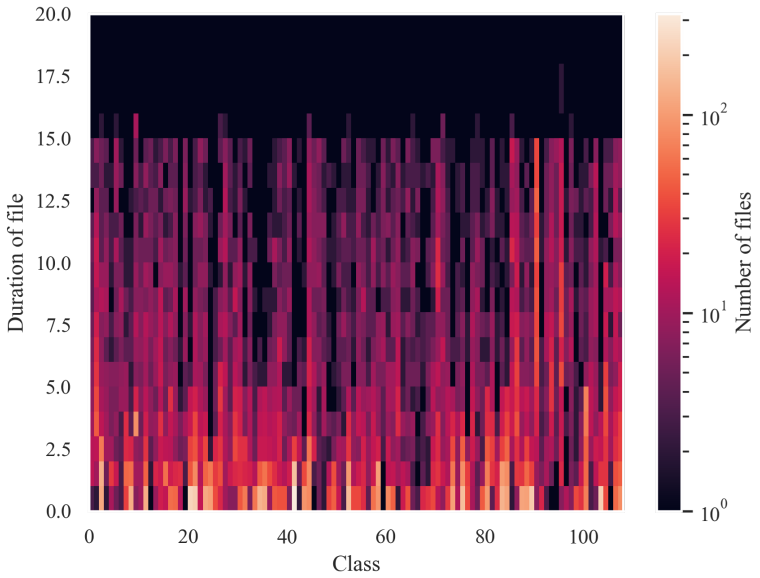

Figure 3.13: The distribution of durations of files per class.

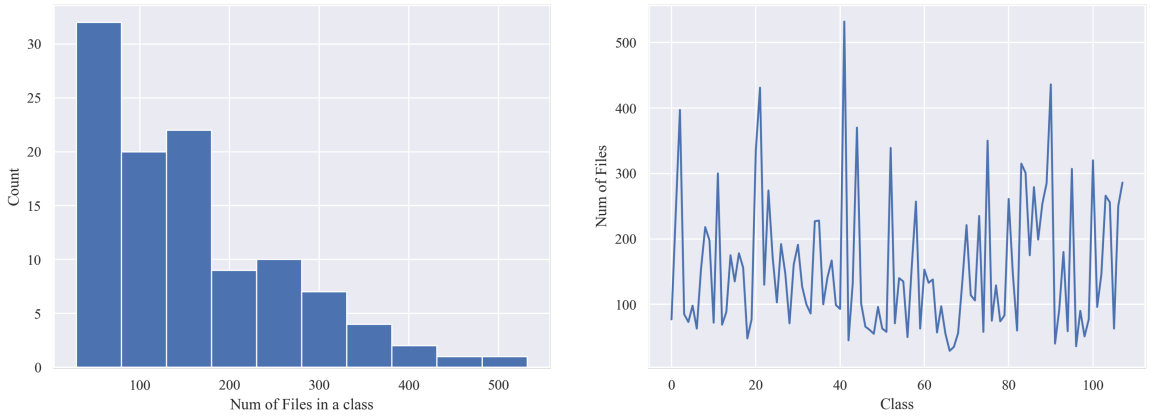

Figure 3.14: Statistics of the number of files within each class.

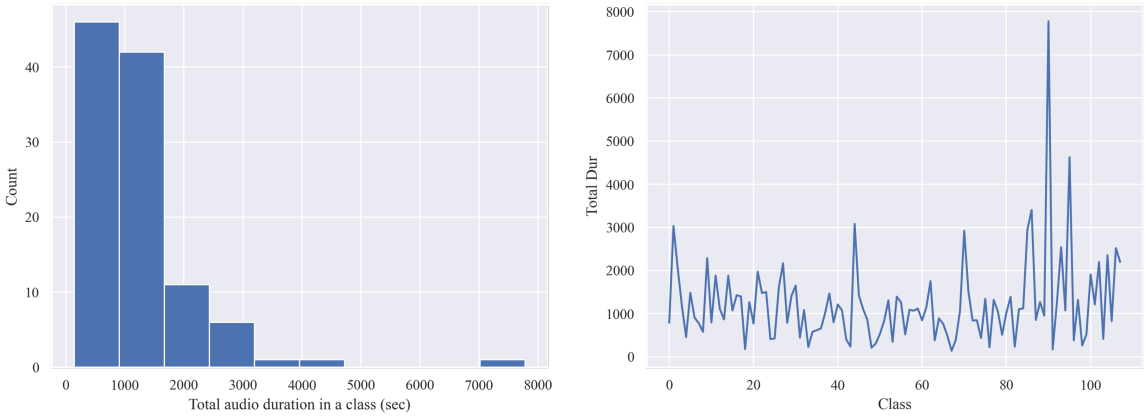

Figure 3.15: Statistics of the total duration for each class.

5-second segments and the remaining 1 second will be discarded. On the contrary, if an audio clip is 23 seconds long, it will firstly be zero padded to 25 seconds, and then split into five 5-second segments. With this strategy, we ended up obtaining 32,522 segments, and 78 classes were randomly selected as the training set, 10 classes as the validation set and the remaining 20 classes as the test set.

3.9.2 Results

The results for the Env-FSD50K dataset are shown in Table 3.6. Similarly, the table shows significant improvement from the Proto-FA model over the Proto-Net model. And it is worth noting that the performance of Proto-HA and Proto-FA is generally similar for this dataset, which could be due to the fact that with the feature refinement of the feature-level attention module, the similarity between a query sample and support samples in the same class is already relatively high. In the meantime, we can still see a noticeable performance gain from Proto-Net to Proto-IA, suggesting that without the feature-level attention, the instance-level attention can still emphasize relevant support samples in this dataset.

Table 3.6: Few-shot sound classification accuracies for prototypical networks with or without attention module on the subset of Env-FSD50K.

Model	Validation	Test			
	10-way 5-shot	5-way 1-shot	5-way 5-shot	10-way 1-shot	10-way 5-shot
Proto-Net	52.78±0.15%	44.94±0.29%	66.75±0.16%	31.85±0.26%	52.98±0.13%
Proto-IA (ours)	53.67±0.18%	–	67.29±0.26%	–	53.44±0.42%
Proto-FA (ours)	**59.70±0.40%**	**53.50±0.29%**	**73.32±0.27%**	**38.88±0.22%**	**59.66±0.48%**
Proto-HA (ours)	**59.59±0.44%**	–	**73.34±0.20%**	–	**59.89±0.26%**

CHAPTER 4

HYBRID ATTENTIONAL PROTOTYPICAL NETWORKS WITH Π-MODEL

From previous results it can be seen that, as expected, the classification accuracy on *noise*ESC-50 dataset is in general lower than that on the clean ESC-50 dataset. To improve the overall performance on noisy data, we thought that it would be beneficial to train a model that is robust to noise. We found that it could be helpful if we made use of the idea of ensemble learning and tried to bring the noisy data closer to their clean counterparts during training. Therefore, we propose to introduce Π-model into the training phase of our framework, which was firstly proposed by Laine *et al.* [121] as one of the self-ensembling methods.

4.1 Ensemble Learning

Ensemble learning is a machine learning strategy that combines two or more models in order to obtain the optimal predictions. It is usually able to improve the predictive performance because: 1) By averaging or combining different hypothesis or models, the risk of overfitting can be reduced; 2) With only a single model, the training process may get stuck in local optima, whereas combining models can help jump out of them; 3) It may not be possible to reach the best model within the space of any single model, while the combination of different models is more likely to get closer to the optimal hypothesis [122]. With the success of deep neural networks (DNNs) in various machine learning tasks, there have been a considerable number of attempts to combine ensemble learning with DNNs [122]. Deep neural decision forests [123] unifies classification trees with representation learning from deep convolutional networks by introducing a stochastic, differentiable, and backpropagation compatible decision tree model. Wen *et al.* [124] proposed an ensemble of CNN models for facial expression recognition using probability-based fusion. For deep-learning based audio classification problems, the usage of ensemble learning has also been

frequently explored. For example, Huang *et al.* [30] created an ensemble of 4 pre-trained neural networks which outperformed individual models on an acoustic scene classification task. And Nanni *et al.* [125] used an ensemble of multiple CNNs along with handcrafted texture descriptors to improve the performance of animal audio classification.

4.2 Π-model

The concept behind Π-model is also ensemble learning, and it is called self-ensembling by the authors [121], since only one classifier is involved and the consensus prediction is obtained from the outputs of the same classifier but under different regularization and input augmentation conditions. The original structure of Π-model is shown in Figure 4.1 and the pseudocode in Algorithm 1. The model was initially designed for semi-supervised learning problems, where the majority of dataset is unlabeled. In Algorithm 1, the overall setting is that the training dataset is consisted of totally N samples, of which M are labeled. The input data sample is denoted as x_i, where $i \in \{1...N\}$. L is the set of labeled input indices and $|L| = M$. The ground truth label for x_i is denoted as y_i if $i \in L$, and $y_i \in \{1...C\}$, where C is the number of classes [121]. From Algorithm 1 it can be seen that every input sample x_i is evaluated twice by the network after some stochastic augmentation. Afterwards, two prediction vectors z_i and $\tilde{z}_i$ are obtained as the outputs. The loss function is consisted of two parts, one is the cross-entropy loss for all the labeled data samples between z_i and their corresponding ground truth labels y_i, and the second part is a mean squared error for all input samples which penalizes the difference between z_i and $\tilde{z}_i$. These two components are then combined by a time-dependent function $w(t)$, which is described as a Gaussian ramp-up curve $exp[-5(1-T)^2]$, where T ranges from 0 to 1 linearly within the ramp-up period (80 epochs in the original paper) [121]. The core purpose of this particular loss function is to try to bring these predictions z_i and $\tilde{z}_i$ close together. It is pointed out in the original paper that the dropout regularization and Gaussian noise along with augmentations are the reasons causing z_i and $\tilde{z}_i$ to be different. Therefore minimizing the

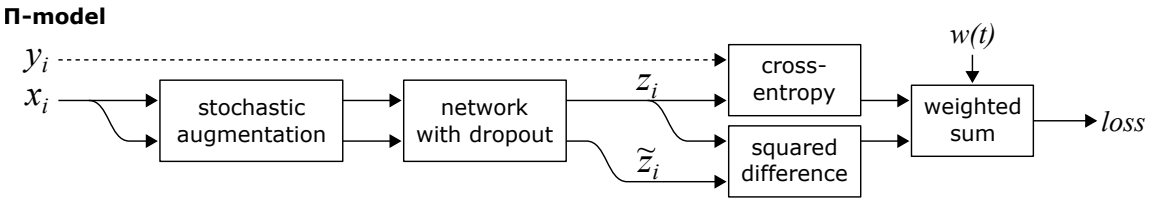

Figure 4.1: The original structure of Π-model [121].

difference between them can be considered as encouraging consistent output between two realizations of the same input sample [121].

4.3 Introducing Π-model into Our Framework

To improve the performance on noisy data, we introduced the idea of Π-model into our hybrid attentional prototypical networks framework, and the system overview is shown in Figure 4.2. We name the model **HAPPi**, which stands for **H**ybrid **A**ttentional **P**rototypical Networks with **Pi**-model. We made several modifications to the original Π-model. First, instead of applying stochastic augmentation on the input stimuli, we used the noisy input sample $\tilde{x}_i$ and its corresponding clean counterpart x_i as the inputs. Second, since dropout regularization is typically used with larger datasets and deeper networks, and the few-shot learning setting might not be a good fit for it. Therefore we did not use any dropout in our network layers. In addition, according to our experiments, we set the ramp-up period to 90 instead of 80 for the weighting function $w(t)$, and the total number of epochs is 100. After the training, the validation and test sets used to evaluate the model were all from *noise*ESC-50 to assess its capability to handle noisy data. Again as previous experiments, each model was run 20 times on a Quadro RTX 6000 GPU. Figure 4.3 shows the training curves of the cross-entropy loss (upper row) and the mean squared loss (lower row). It can be clearly observed that training with Π-model can successfully minimize the classification error and encourage consistent predictions from clean and noisy data.

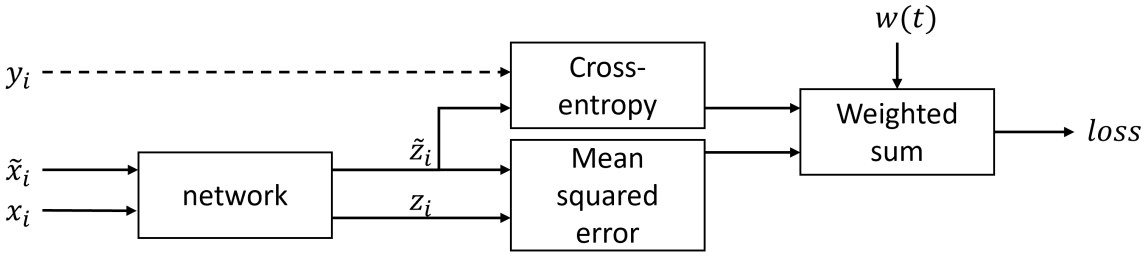

Figure 4.2: The Π-model structure in our setting.

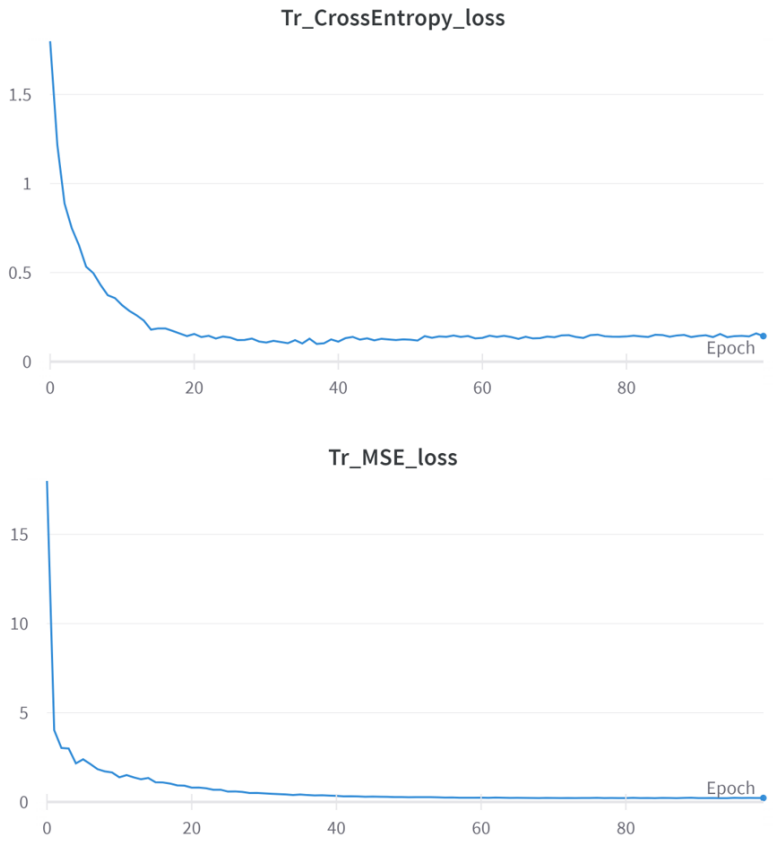

Figure 4.3: The Π-model training curves of the cross-entropy loss (upper row) and the mean squared error loss (lower row).

4.4 Results

Table 4.1 shows the comparison of all models trained with or without Π-model on *noise*ESC-50. The suffix Pi attached to each model means that this model was trained with Π-model. Again the average accuracy and 95% confidence intervals are reported. It is shown that by adding Π-model and training with both ESC-50 and *noise*ESC-50, the validation and test results are boosted in all scenarios, indicating that the whole model becomes more noise robust with the addition of Π-model. And our final proposed model, **HAPPi**, performed best among all models, proving that the combination of hybrid attention and Π-model is indeed beneficial to deal with bad support samples and noisy data for few-shot environmental sound classification.

Algorithm 1 **Π-model pseudocode** [121]

Require: x_i = training stimuli
Require: L = set of training input indices with known labels
Require: y_i = labels for labeled inputs $i \in L$
Require: $w(t)$ = unsupervised weight ramp-up function
Require: $f_\theta(x)$ = stochastic neural network with trainable parameters θ
Require: $g(x)$ = stochastic input augmentation function
for t in [1, num_epochs] **do**
  for each minibatch B **do**
    $z_{i \in B} \leftarrow f_\theta(g(x_{i \in B}))$ ▷ Evaluate network outputs for augmented inputs
    $\tilde{z}_{i \in B} \leftarrow f_\theta(g(x_{i \in B}))$ ▷ Again, with different dropout and augmentation
    $loss \leftarrow -\frac{1}{|B|} \sum_{i \in (B \bigcap L)} log z_i[y_i]$ ▷ Supervised loss component
      $+ w(t) \frac{1}{C|B|} \sum_{i \in B} \left\| z_i - \tilde{z}_i \right\|^2$ ▷ Unsupervised loss component
    update θ using, e.g., ADAM ▷ Update network parameters
  end for
end for
return θ

Table 4.1: Comparison between models trained with or without Π-model on *noise*ESC-50

Model	Validation	Test			
	10-way 5-shot	5-way 1-shot	5-way 5-shot	10-way 1-shot	10-way 5-shot
Proto-Net	69.08±0.50%	61.06±0.68%	81.21±0.64%	45.24±0.62%	65.74±0.61%
Proto-IA (ours)	69.93±0.53%	–	81.26±0.48%	–	66.05±0.56%
Proto-FA (ours)	78.16±0.50%	**71.40±0.73%**	**87.53±0.47%**	**56.48±0.80%**	77.51±0.65%
Proto-HA (ours)	**78.93±0.58%**	–	**87.65±0.43%**	–	**78.10±0.66%**
Proto-Net-Pi	70.61±0.38%	62.23±1.00%	82.15±0.47%	46.43±0.83%	66.65±0.47%
Proto-IA-Pi (ours)	71.96±0.58%	–	82.03±0.54%	–	66.65±0.61%
Proto-FA-Pi (ours)	78.53±0.54%	**72.30±0.69%**	88.09±0.51%	**58.91±0.64%**	78.50±0.72%
HAPPi (ours)	**79.54±0.69%**	–	**88.34±0.44%**	–	**78.71±0.72%**

CHAPTER 5
EXPLORING FEATURE EXTRACTION USING AN INNOVATIVE SIGNAL PROCESSING TECHNIQUE

As previously mentioned, feature extraction is an essential step for audio classification tasks, and for years cepstral features have been superior over other types of features. Fundamentally, cepstral features such as MFCCs and Log Mel-spectrograms are both fourier based features, where the short-time-fourier-transform is always computed in the first step. In this chapter, we explore an innovative signal processing based temporal-frequency representation named IMFogram. It is based on a signal decomposition method called iterative filtering (IF), which is an alternative version of empirical mode decomposition (EMD), one of the first ever introduced iterative methods in the literature of signal processing. The motivation to explore IMFogram comes from the fact that it more-compactly models non-stationary signals. Furthermore, we found iterative filtering features to be very useful (outperforming Fourier methods) in the analysis on machinery signals in an early project. Given the recent development of IMFogram as a time-frequency representation method, which relies on the signal decomposition produced by the iterative filtering method, we tested how this new feature perform with our multi-channel-temporal-attention (MCTA) CNN model described in **CHAPTER 2**.

5.1 Empirical Mode Decomposition

The empirical mode decomposition (EMD) algorithm [19] was initially designed to solve the problem of decomposing non-stationary signals into simpler components called Intrinsic Mode Functions (IMFs). Each IMF has a unique instantaneous frequency at every instant of time, and it is defined to satisfy two requirements: 1) The number of extrema (local maxima and minima) and the number of zero-crossing must be equal or differ at most by

one; 2) The mean value of the envelope defined by local maxima and the envelope defined by local minima is zero at any point [21]. This method is capable of decomposing a signal without any a priori information about it or a priori selection of a certain basis [126]. The pseudocode of the EMD algorithm is shown in Algorithm 2 [126]. The core idea behind it is called the sifting process, where within each iteration, the moving average $M(s)$ is subtracted from the signal itself. To compute $M(s)$, it was proposed in [19] that the upper and lower envelopes which connect the signal maxima and minima respectively to be computed first through cubic splines, followed by computing the point by point mean of these two curves [126]. The same process is applied to the remainder signal after subtraction, and it is repeated until it converges to an IMF or the stopping criterion is satisfied. One of the major disadvantages is that the EMD method is very vulnerable to noise, and the decomposition results can be dramatically different between clean and noisy signals. Therefore, to solve this issue, the same authors proposed the Ensemble Empirical Mode Decomposition (EEMD) [20], of which the basic idea is to firstly add hundreds of white noise realizations to the target signal, and then the average of all different decompositions is computed as the final outcome [126]. However, for EMD and EEMD, there are still some fundamental mathematical problems unresolved, such as the convergence of the sifting problem, which was then addressed in the original work of iterative filtering (IF) method described below [21].

5.2 Iterative Filtering

Iterative filtering (IF) is an alternative version of Empirical Mode Decomposition (EMD) based algorithm. This algorithm was first proposed by Lin *et al.* [21] and it has been proved that it has guaranteed convergence and stability [127]. The main difference between IF and EMD based methods is that instead of computing the moving average of the upper and lower envelopes, the local average values of the signal are computed by integrating the signal itself weighted with an a priori chosen mask function [126]. And to guarantee the

Algorithm 2 Empirical Mode Decomposition IMF = EMD(f)

IMF = $\{\}$
while the number of extrema of $f \geq 2$ **do**
  $s_1 = s$
  while the stopping criterion is not satisfied **do**
    compute the moving average $M(s_m(x))$
    $s_{m+1}(x) = s_m(x) - M(s_m(x))$
    $m = m + 1$
  end while
  IMF = IMF $\bigcup \{s_m\}$
  $s = s - s_m$
end while
IMF = IMF $\bigcup \{s\}$

convergence and stability, the mask function is chosen to be a result of convolution with itself of a function fulfilling properties such as compactly supported, non-negative, even, and with integral equal to 1 [126]. Algorithm 3 shows the pseudo code of this algorithm , where $w_m(f)$ is the chosen mask function with a support in $[-l_m, l_m]$, and l_m stands for mask length which represents half of the support length [126]. As for the implementation, Cicone *et al.* [127] proposed a fast fourier transform (FFT) based method called Fast Iterative Filtering (FIF), which was used in this work. The details of this implementation method are left out since they are out of the scope of this thesis.

Algorithm 3 Iterative Filtering IMF = IF(s)

IMF = $\{\}$
while the number of extrema of $s \geq 2$ **do**
  $s_1 = s$
  while the stopping criterion is not satisfied **do**
    compute the filter length l_m for $s_m(x)$
    $s_{m+1}(x) = s_m(x) - \int_{-l_m}^{l_m} s_m(x+t) w_m(t) dt$
    $m = m + 1$
  end while
  IMF = IMF $\bigcup \{s_m\}$
  $s = s - s_m$
end while
IMF = IMF $\bigcup \{s\}$

5.3 IMFogram

With the resultant IMFs from iterative filtering decomposition, an innovative time-frequency representation called IMFogram was proposed in [128]. It is a matrix that contains local amplitudes of various IMFs which spread across rows and columns based on the corresponding local frequencies and time windows respectively [129]. It is noteworthy that it has been proved in [129] that IMFogram can converge to spectrogram in certain conditions, and it is actually some sort of generalization of spectrogram with higher accuracy in time and frequency representation, suggesting that it could be a good alternative to spectrogram and other fourier based features. The pseudocode of IMFogram calculation is shown in Algorithm 4 [129] and we leave out further details as well. Figure 5.1 shows the comparison between an IMFogram and a log Mel-spectrogram of the a "hen" sound from ESC-50.

Algorithm 4 IMFogram A = IMFogram(IMFs)

M : number of IMFs
N : signal length
R : number of overlapping time windows $I_j = [a_j,\ b_j]$
for $k = 1$ to M **do**
  Compute the local amplitudes $LA_{I_j}^{(k)}$
  Compute the local frequencies $LF_{I_j}^{(k)}$
  for $j = 1$ to R **do**

$$A(LF_{I_j}^{(k)}, j) = A(LF_{I_j}^{(k)}, j) + LA_{I_j}^{(k)}$$

  end for
end for
return A

5.4 Comparing IMFograms with Log Mel-spectrograms

To evaluate how IMFograms perform as features for environmental sound classification, we compared the results with IMFograms and log Mel-spectrograms as inputs using our proposed MCTA-CNN model. No delta features were extracted and no data augmentation was applied. The log Mel-spectrograms were created with a window length of 4096 and

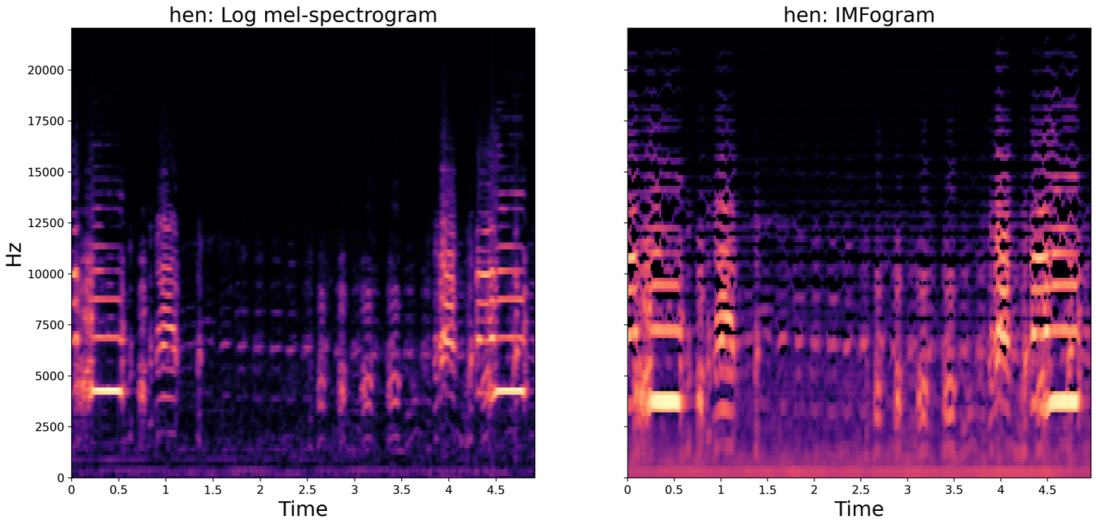

Figure 5.1: Log Mel-spectrogram vs. IMFogram for a "hen" sound.

Table 5.1: Classification results of log Mel-spectrogram and IMFogram on MCTA-CNN model.

Model	Log Mel-spectrogram	IMFogram
MCTA-CNN (ours)	79.22±0.45%	79.26±0.30%

a hop size of 3700 with 128 Mel bins, as described in **CHAPTER 2**, and the IMFograms were created with a window length of 2000 and a hop size of 1000 with 128 frequency bins as well. Both features were extracted using MATLAB. The results are shown in Table 5.1 and the average accuracy of 5 runs and its standard deviation are reported. It can be seen that with regard to overall classification performance, these two features have very similar performance. After breaking down the average accuracy for each class, as shown in Figure 5.2, we can observe that IMFogram actually outperformed log Mel-spectrogram in many classes, such as "dog", "frog", "cat", "footsteps", "helicopter", "fireworks", and so on. This means that at least for certain classes, IMFogram has the potential to perform better than log Mel-spectrogram. Since IMFogram was not a clear winner in all cases or, at least, most cases, we used the Fourier-based features in this work. However, with further finetuning, IMFogram may even beat log Mel-spectrograms in many other ways, which makes it an interesting future research direction.

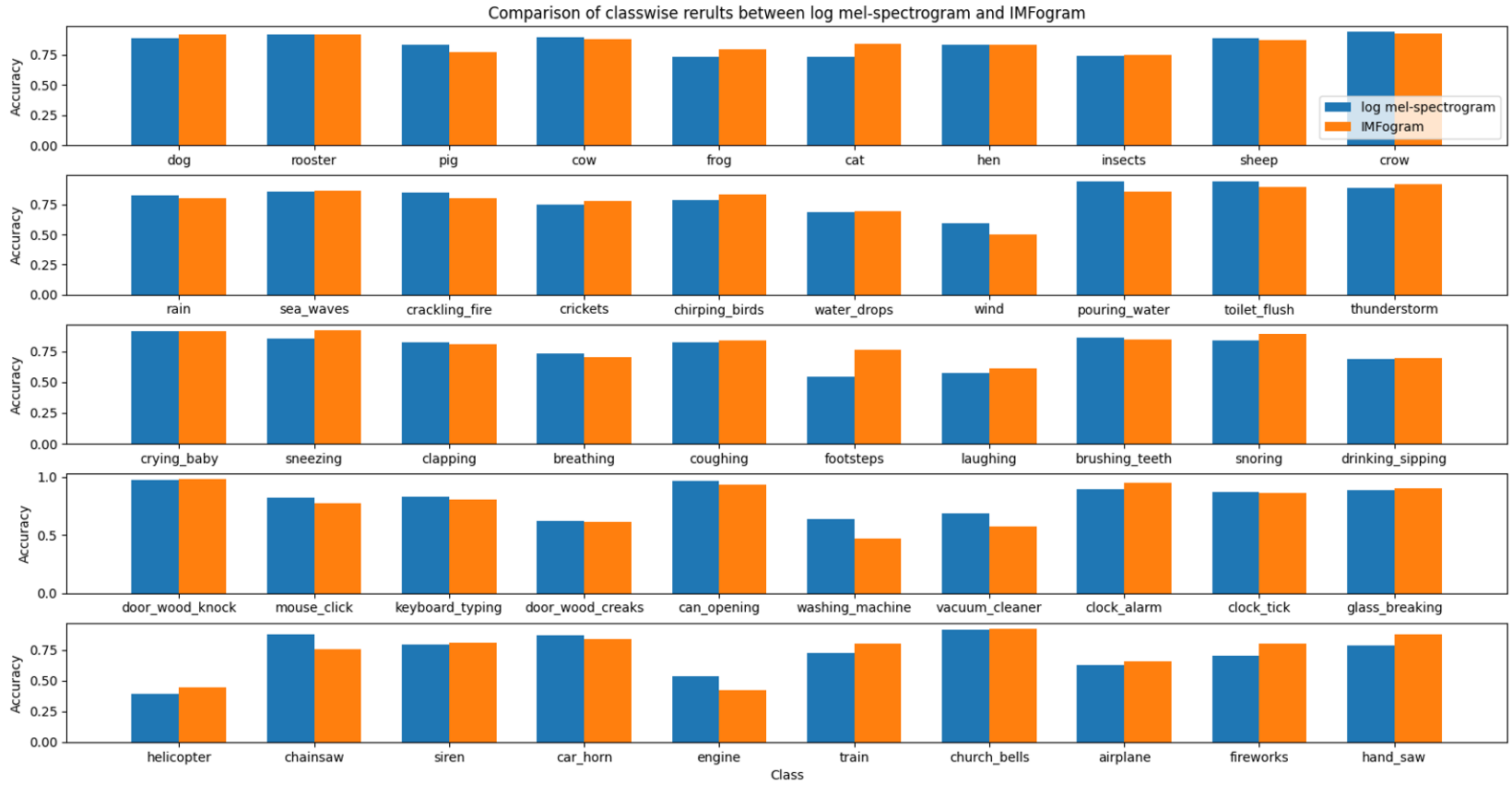

Figure 5.2: Classwise comparison between log Mel-spectrogram and IMFogram.

CHAPTER 6
CONCLUSION AND FUTURE WORK

The proposed research focuses on deep learning based environmental sound classification with attention and its combination with few-shot learning. This chapter concludes our work and discusses potential future work plans.

6.1 Conclusion

In **CHAPTER 2**, the proposed multi-channel temporal attention (MCTA) CNN model is presented. This model makes use of the unique temporal structure of audio signals and therefore introduces temporal attention to focus certain parts of the extracted features that contribute more to the final classification. Unlike other previous temporal attention deep learning models which only generate a single attention vector, the multi-channel structure of our model is capable of fully exploiting the temporal information from different embedded channels. Experiments on various environmental sound classification datasets such as ESC-50, ESC-10, UrbanSound8K, and acoustic scene classification datasets such as DCASE 2018 and 2019 development sets show that our MCTA model can indeed improve classification results.

One of the big challenges of environmental sound classification is that in many cases the amount of training data is limited, especially when there are not enough samples in certain classes. To overcome this issue, few-shot learning techniques are used. **CHAPTER 3** introduces a prototypical network framework integrated with a hybrid attention module. This module is consisted of two separate attention blocks, one is feature-level attention and the other is instance-level attention. The feature-level attention block is adapted from our MCTA model, and the instance-level block assigns different weights to samples in the support set when computing the prototypes in order to diminish the effect of bad support

samples. When tested on ESC-50, *noise*ESC-50, and Env-FSD50K datasets with different settings, our model provides promising results on 5-way 1-shot, 5-way 5-shot, 10-way 1-shot, and 10-way 5-shot scenarios.

To further improve few-shot performance on *noise*ESC-50, in **CHAPTER 4** the idea of Π-model is introduced and combined with our hybrid attention prototypical network framework. Π-model is a type of self-ensembling method in which a mean squared error term is added to the loss function. This helps to encourage the same predictions for clean and noisy inputs. Experiments on *noise*ESC-50 show that adding the Π-model can enhance the few-shot performance in all models and scenarios.

6.2 Future Work

Based on the work that has been done, there are many directions in which it can be extended, which are listed below.

6.2.1 Other Datasets and Model Improvement

Our proposed model can be further tested on other datasets such as the dataset from DCASE 2021 task 5. This task aims to apply few-shot learning settings to sound event detection for animal voclalizations [130]. The sound classes are mainly from mammals and birds, and both model development and evaluation are in 5-shot scenarios. Since this task focuses sound event detection, it is a more complex problem than just the classification since the onset/offset time steps are also involved. Our model can be modified to fit this challenge and it would be interesting to see where it stands in the rankings. Audioset is another much larger and more challenging dataset which our model can be tested on. Additionally, the architecture of our model can be further improved, such as finetuning the architectures of the backbone CNN encoders, and investigating better ways of computing instance-level attention weights.

6.2.2 Exploration of Transformers

The transformer architecture has been one of the most influential models in the past few years. It was originally proposed by Vaswani *et al.* in 2017 [131] and has since achieved state-of-the-art performance in Natural Language Processing [132, 133] and other domains including audio classification [134, 135]. The essential idea of transformers is to obtain a weighted average of a sequence of vectors called values, with the weights computed based on three other core parts: query, key, and a score function [136]. Query is a feature vector that describes the target characteristic in the sequence, i.e. what we potentially need to pay attention to. Keys are also feature vectors, each of which contains information provided by each input sequence element. A score function then takes the query and a key as input and compute the score of this query-key pair as the attention weight for a corresponding value vector [136]. Since transformers deal with sequential input elements, the whole process is actually to compute a representation of an input sequence by evaluating the relationship between each pair of elements within the same sequence, and hence this type of attention is called self-attention. Inspired by the recent success of Transformers and other related transformer-based audio classification models, we think that it is desirable to replace our current attention implementation with transformers to further boost the performance.

6.2.3 Continual Learning

Our few-shot learning model can be combined with the continual learning framework. The key idea of continual learning is that the class vocabulary is not fixed and can be expanded every time a new class is seen. With a trained base classifier, the model can keep learning to recognize new classes after being exposed to just a few instances [137]. This can largely enhance the model's ability to adapt to new tasks without forgetting the existing knowledge it already possesses.

6.2.4 IMFogram Finetuning

As introduced in **CHAPTER 5**, we saw potentials in IMFogram as a new type of feature for audio classification. The parameters used to generate it can be finetuned and its mathematical foundation can be further investigated. It is possible that it can replace cepstral features in some applications.

Appendices

APPENDIX A

CONFUSION MATRIX EXAMPLES FOR MCTA CNN MODEL

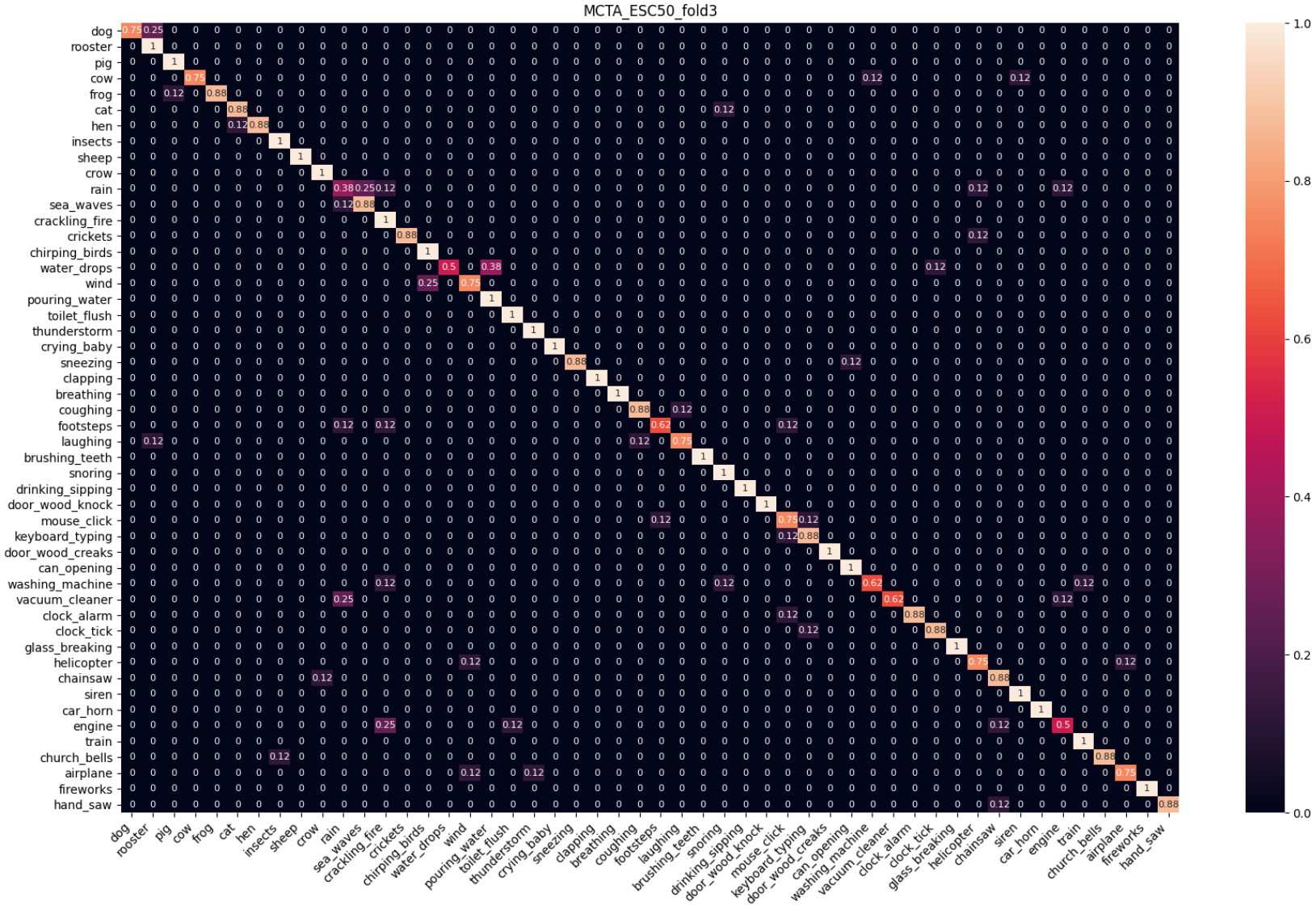

Figure A.1: Confusion matrix of ESC-50 dataset fold 3 with our multi-channel temporal attention (MCTA) CNN model

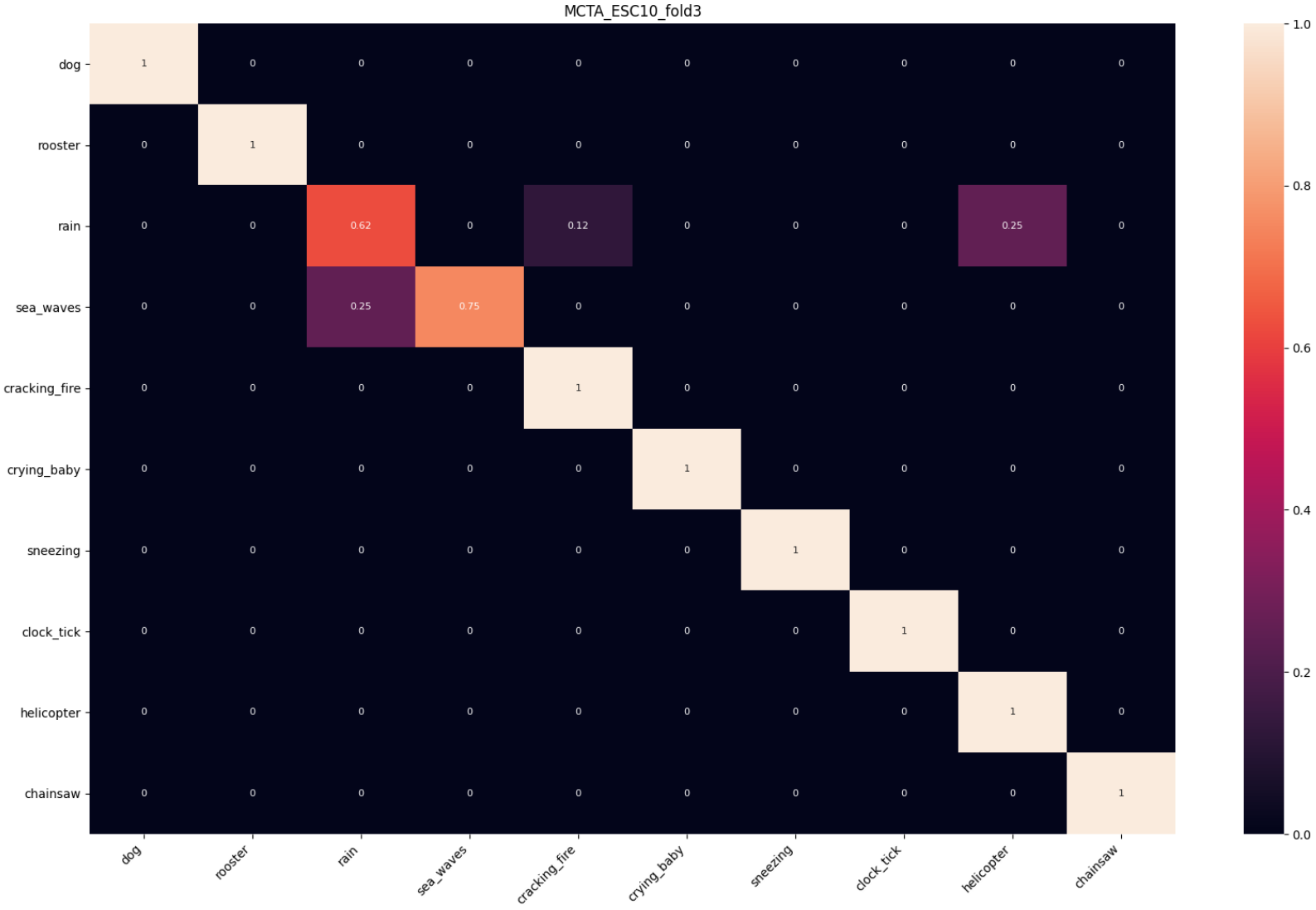

Figure A.2: Confusion matrix of ESC-10 dataset fold 3 with our multi-channel temporal attention (MCTA) CNN model

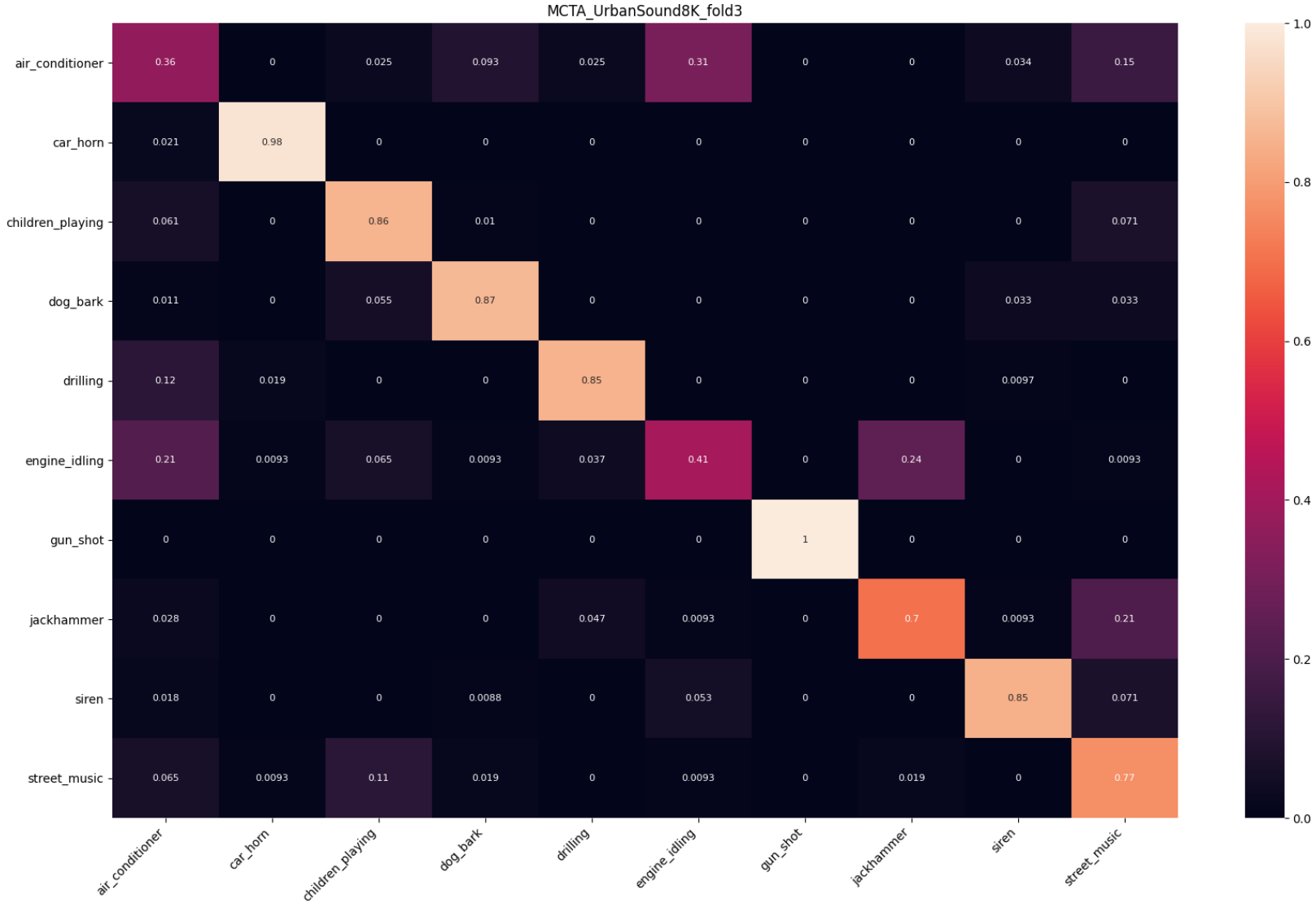

Figure A.3: Confusion matrix of UrbanSound8K dataset fold 3 with our multi-channel temporal attention (MCTA) CNN model

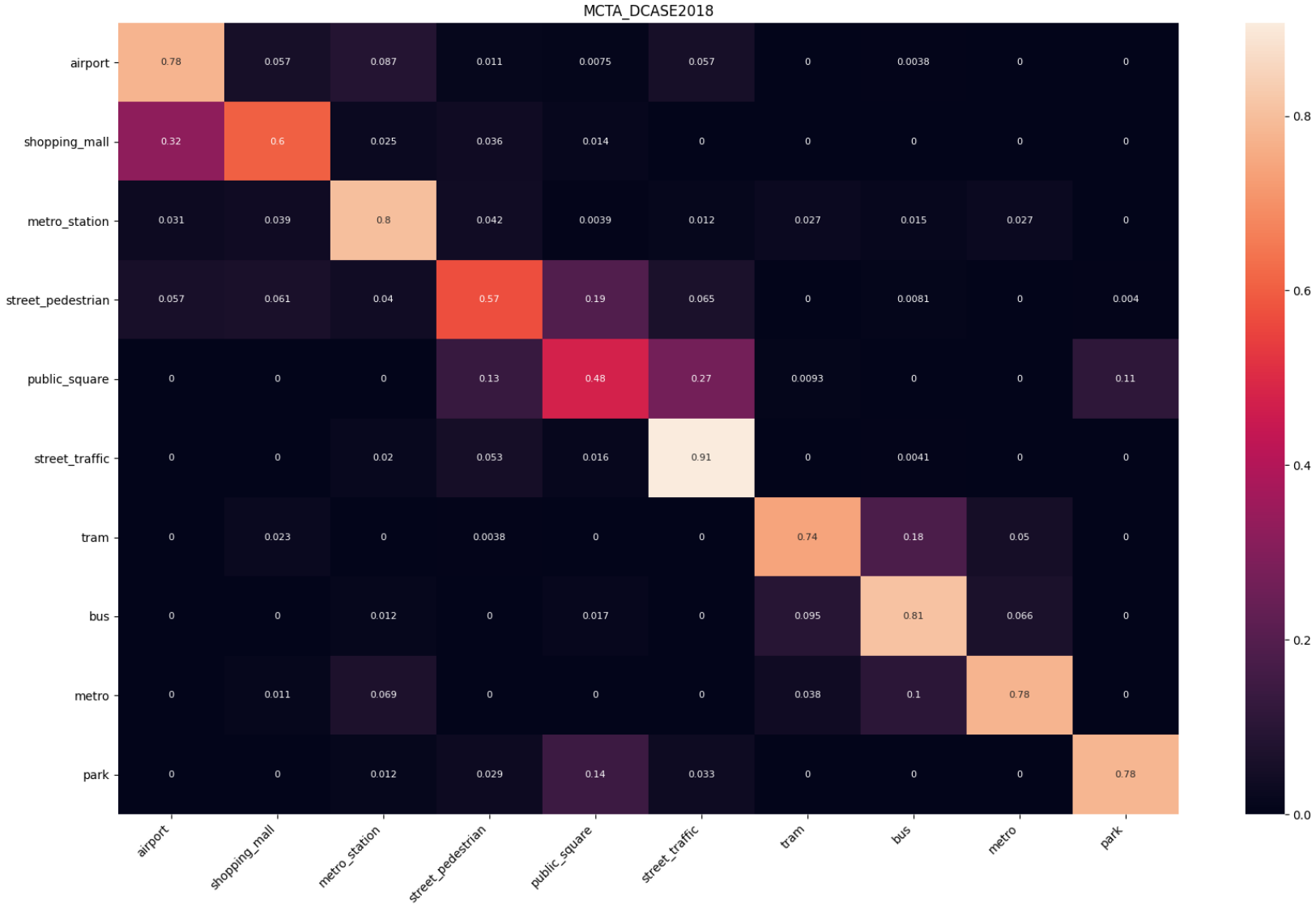

Figure A.4: Confusion matrix of DCASE2018 dataset fold 3 with our multi-channel temporal attention (MCTA) CNN model

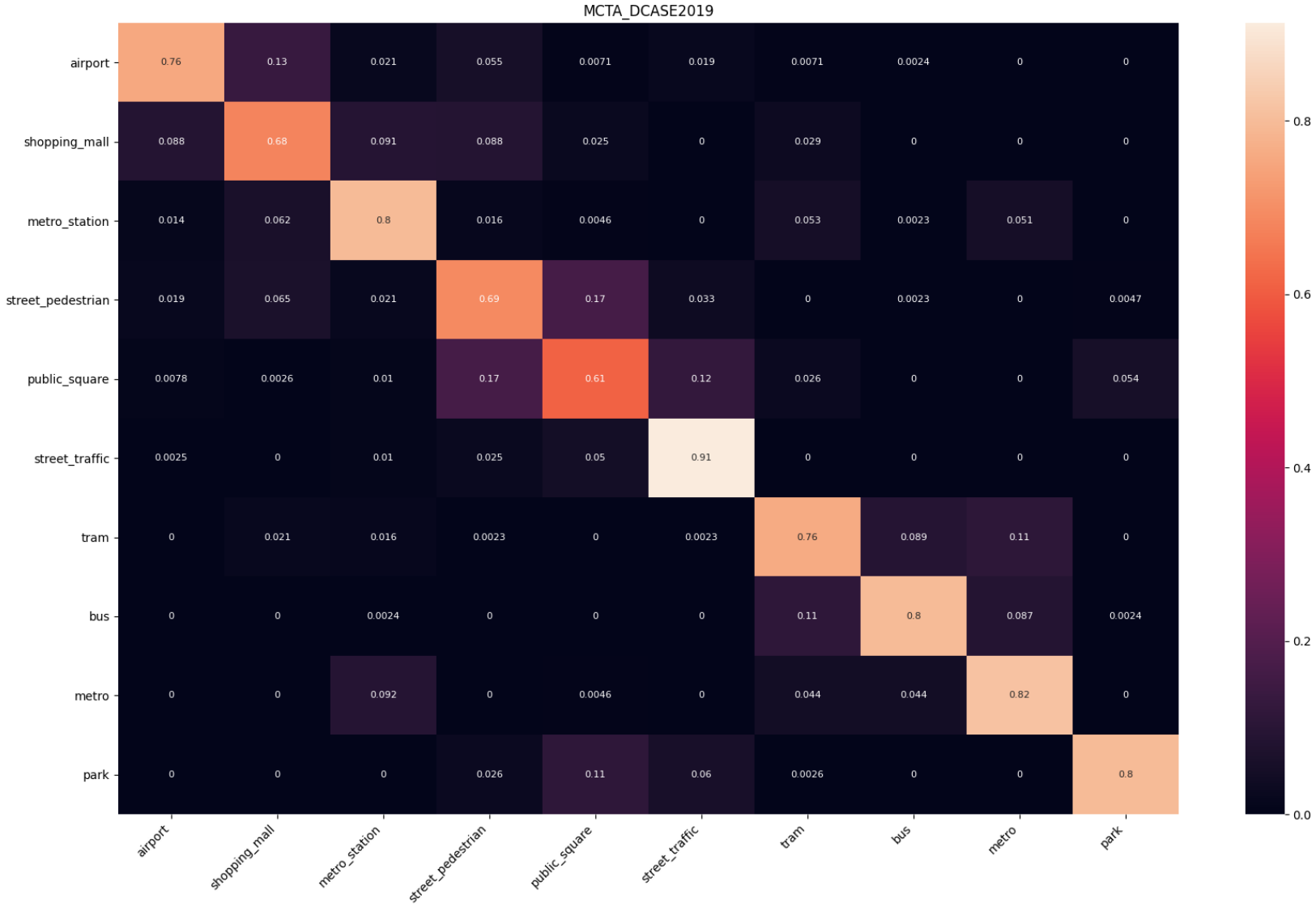

Figure A.5: Confusion matrix of DCASE2019 dataset fold 3 with our multi-channel temporal attention (MCTA) CNN model

REFERENCES

[1] P. Foggia, N. Petkov, A. Saggese, N. Strisciuglio, and M. Vento, "Audio surveillance of roads: A system for detecting anomalous sounds," *IEEE transactions on intelligent transportation systems*, vol. 17, no. 1, pp. 279–288, 2015.

[2] E. Alexandre, L. Cuadra, M. Rosa, and F. Lopez-Ferreras, "Feature selection for sound classification in hearing aids through restricted search driven by genetic algorithms," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 8, pp. 2249–2256, 2007.

[3] M. Vacher, J.-F. Serignat, and S. Chaillol, "Sound classification in a smart room environment: An approach using gmm and hmm methods," 2007.

[4] K. J. Piczak, "Esc: Dataset for environmental sound classification," in *Proceedings of the 23rd ACM international conference on Multimedia*, ACM, 2015, pp. 1015–1018.

[5] E. Fix and J. L. Hodges, "Discriminatory analysis. nonparametric discrimination: Consistency properties," *International Statistical Review/Revue Internationale de Statistique*, vol. 57, no. 3, pp. 238–247, 1989.

[6] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[7] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern classification and scene analysis*, vol. 3.

[8] L. R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[9] J. Abeßer, "A review of deep learning based methods for acoustic scene classification," *Applied Sciences*, vol. 10, no. 6, 2020.

[10] *Detection and Classification of Acoustic Scenes and Events*, https://dcase.community/.

[11] *Overview of an acoustic scene classification system.* https://dcase.community/challenge2016/task-acoustic-scene-classification.

[12] S. Chachada and C.-C. J. Kuo, "Environmental sound recognition: A survey," *APSIPA Transactions on Signal and Information Processing*, vol. 3, 2014.

[13] S. S. Stevens, J. Volkmann, and E. B. Newman, "A scale for the measurement of the psychological magnitude pitch," *The journal of the acoustical society of america*, vol. 8, no. 3, pp. 185–190, 1937.

[14] S. Davis and P. Mermelstein, "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences," *IEEE transactions on acoustics, speech, and signal processing*, vol. 28, no. 4, pp. 357–366, 1980.

[15] A. Eronen, "Comparison of features for musical instrument recognition," in *Proceedings of the 2001 IEEE Workshop on the Applications of Signal Processing to Audio and Acoustics (Cat. No. 01TH8575)*, IEEE, 2001, pp. 19–22.

[16] *Speech Processing for Machine Learning: Filter banks, Mel-Frequency Cepstral Coefficients (MFCCs) and What's In-Between*, https://haythamfayek.com/2016/04/21/speech-processing-for-machine-learning.html.

[17] J. Makhoul, "Linear prediction: A tutorial review," *Proceedings of the IEEE*, vol. 63, no. 4, pp. 561–580, 1975.

[18] E. Wong and S. Sridharan, "Comparison of linear prediction cepstrum coefficients and mel-frequency cepstrum coefficients for language identification," in *Proceedings of 2001 International Symposium on Intelligent Multimedia, Video and Speech Processing. ISIMP 2001 (IEEE Cat. No. 01EX489)*, IEEE, 2001, pp. 95–98.

[19] N. E. Huang *et al.*, "The empirical mode decomposition and the hilbert spectrum for nonlinear and non-stationary time series analysis," *Proceedings of the Royal Society of London. Series A: mathematical, physical and engineering sciences*, vol. 454, no. 1971, pp. 903–995, 1998.

[20] Z. Wu and N. E. Huang, "Ensemble empirical mode decomposition: A noise-assisted data analysis method," *Advances in adaptive data analysis*, vol. 1, no. 01, pp. 1–41, 2009.

[21] L. Lin, Y. Wang, and H. Zhou, "Iterative filtering as an alternative algorithm for empirical mode decomposition," *Advances in Adaptive Data Analysis*, vol. 1, no. 04, pp. 543–560, 2009.

[22] K. Qian, Z. Ren, V. Pandit, Z. Yang, Z. Zhang, and B. Schuller, "Wavelets revisited for the classification of acoustic scenes," in *Proc. DCASE Workshop, Munich, Germany*, 2017, pp. 108–112.

[23] Z. Ren, V. Pandit, K. Qian, Z. Yang, Z. Zhang, and B. Schuller, "Deep sequential image features on acoustic scene classification," in *Proc. DCASE Workshop, Munich, Germany*, 2017, pp. 113–117.

[24] X. Li, V. Chebiyyam, and K. Kirchhoff, "Multi-stream network with temporal attention for environmental sound classification," *arXiv preprint arXiv:1901.08608*, 2019.

[25] V. Bisot, S. Essid, and G. Richard, "Hog and subband power distribution image features for acoustic scene classification," in *2015 23rd European Signal Processing Conference (EUSIPCO)*, IEEE, 2015, pp. 719–723.

[26] J. Sharma, O.-C. Granmo, and M. Goodwin, "Environment sound classification using multiple feature channels and deep convolutional neural networks," *arXiv preprint arXiv:1908.11219*, 2019.

[27] N. Jaitly and G. Hinton, "Learning a better representation of speech soundwaves using restricted boltzmann machines," in *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2011, pp. 5884–5887.

[28] R. Salakhutdinov, A. Mnih, and G. Hinton, "Restricted boltzmann machines for collaborative filtering," in *Proceedings of the 24th international conference on Machine learning*, 2007, pp. 791–798.

[29] J. J. Huang and J. J. A. Leanos, "Aclnet: Efficient end-to-end audio classification cnn," *arXiv preprint arXiv:1811.06669*, 2018.

[30] J. Huang, H. Lu, P. Lopez Meyer, H. Cordourier, and J. Del Hoyo Ontiveros, "Acoustic scene classification using deep learning-based ensemble averaging," 2019.

[31] Y. Tokozume and T. Harada, "Learning environmental sounds with end-to-end convolutional neural network," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2017, pp. 2721–2725.

[32] A. Singh, P. Rajan, and A. Bhavsar, "Deep multi-view features from raw audio for acoustic scene classification," 2019.

[33] S. Boll, "Suppression of acoustic noise in speech using spectral subtraction," *IEEE Transactions on acoustics, speech, and signal processing*, vol. 27, no. 2, pp. 113–120, 1979.

[34] Y. Wang, P. Getreuer, T. Hughes, R. F. Lyon, and R. A. Saurous, "Trainable frontend for robust and far-field keyword spotting," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2017, pp. 5670–5674.

[35] V. Lostanlen *et al.*, "Per-channel energy normalization: Why and how," *IEEE Signal Processing Letters*, vol. 26, no. 1, pp. 39–43, 2018.

[36] Y. Wu and T. Lee, "Enhancing sound texture in cnn-based acoustic scene classification," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2019, pp. 815–819.

[37] Y. Han, J. Park, and K. Lee, "Convolutional neural networks with binaural representations and background subtraction for acoustic scene classification," *the Detection and Classification of Acoustic Scenes and Events (DCASE)*, pp. 1–5, 2017.

[38] Z. Rafii and B. Pardo, "Music/voice separation using the similarity matrix.," in *ISMIR*, 2012, pp. 583–588.

[39] H. Seo, J. Park, and Y. Park, "Acoustic scene classification using various preprocessed features and convolutional neural networks," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events Workshop (DCASE), New York, NY, USA*, 2019, pp. 25–26.

[40] O. Mariotti, M. Cord, and O. Schwander, "Exploring deep vision models for acoustic scene classification," *Proc. DCASE*, pp. 103–107, 2018.

[41] A. Briliani, B. Irawan, and C. Setianingsih, "Hate speech detection in indonesian language on instagram comment section using k-nearest neighbor classification method," in *2019 IEEE International Conference on Internet of Things and Intelligence System (IoTaIS)*, IEEE, 2019, pp. 98–104.

[42] J. K. Shah, B. Y. Smolenski, R. E. Yantorno, and A. N. Iyer, "Sequential k-nearest neighbor pattern recognition for usable speech classification," in *2004 12th European Signal Processing Conference*, IEEE, 2004, pp. 741–744.

[43] A. Ganapathiraju, J. E. Hamaker, and J. Picone, "Applications of support vector machines to speech recognition," *IEEE transactions on signal processing*, vol. 52, no. 8, pp. 2348–2355, 2004.

[44] M. Jain, S. Narayan, P. Balaji, A. Bhowmick, R. K. Muthu, *et al.*, "Speech emotion recognition using support vector machine," *arXiv preprint arXiv:2002.07590*, 2020.

[45] D. Povey *et al.*, "Subspace gaussian mixture models for speech recognition," in *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*, IEEE, 2010, pp. 4330–4333.

[46] T. Chen, C. Huang, E. Chang, and J. Wang, "Automatic accent identification using gaussian mixture models," in *IEEE Workshop on Automatic Speech Recognition and Understanding, 2001. ASRU'01.*, IEEE, 2001, pp. 343–346.

[47] H.-G. Kim, N. Moreau, and T. Sikora, "Audio classification based on mpeg-7 spectral basis representations," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 14, no. 5, pp. 716–725, 2004.

[48] F. Su, L. Yang, T. Lu, and G. Wang, "Environmental sound classification for scene recognition using local discriminant bases and hmm," in *Proceedings of the 19th ACM international conference on Multimedia*, 2011, pp. 1389–1392.

[49] J. Salamon, C. Jacoby, and J. P. Bello, "A dataset and taxonomy for urban sound research," in *22nd ACM International Conference on Multimedia (ACM-MM'14)*, Orlando, FL, USA, Nov. 2014, pp. 1041–1044.

[50] J. F. Gemmeke *et al.*, "Audio set: An ontology and human-labeled dataset for audio events," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2017, pp. 776–780.

[51] A. Mesaros *et al.*, "Detection and classification of acoustic scenes and events: Outcome of the dcase 2016 challenge," *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, vol. 26, no. 2, pp. 379–393, 2018.

[52] A. Mesaros *et al.*, "Dcase 2017 challenge setup: Tasks, datasets and baseline system," 2017.

[53] K. J. Piczak, "Environmental sound classification with convolutional neural networks," in *2015 IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP)*, IEEE, 2015, pp. 1–6.

[54] H. B. Sailor, D. M. Agrawal, and H. A. Patil, "Unsupervised filterbank learning using convolutional restricted boltzmann machine for environmental sound classification.," in *INTERSPEECH*, 2017, pp. 3107–3111.

[55] J. Salamon and J. P. Bello, "Deep convolutional neural networks and data augmentation for environmental sound classification," *IEEE Signal Processing Letters*, vol. 24, no. 3, pp. 279–283, 2017.

[56] A. Kumar, M. Khadkevich, and C. Fügen, "Knowledge transfer from weakly labeled audio using convolutional neural network for sound events and scenes," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2018, pp. 326–330.

[57] Z. Ren, Q. Kong, J. Han, M. D. Plumbley, and B. W. Schuller, "Attention-based atrous convolutional neural networks: Visualisation and understanding perspectives of acoustic scenes," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2019, pp. 56–60.

[58] K. Koutini, H. Eghbal-zadeh, G. Widmer, and J. Kepler, "Cp-jku submissions to dcase'19: Acoustic scene classification and audio tagging with receptive-field-regularized cnns," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events Workshop (DCASE), New York, NY, USA*, 2019, pp. 25–26.

[59] K. Koutini, H. Eghbal-zadeh, and G. Widmer, "Receptive-field-regularized cnn variants for acoustic scene classification," *arXiv preprint arXiv:1909.02859*, 2019.

[60] A. M. Basbug and M. Sert, "Acoustic scene classification using spatial pyramid pooling with convolutional neural networks," in *2019 IEEE 13th International Conference on Semantic Computing (ICSC)*, IEEE, 2019, pp. 128–131.

[61] Q. Kong, C. Yu, Y. Xu, T. Iqbal, W. Wang, and M. D. Plumbley, "Weakly labelled audioset tagging with attention neural networks," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 27, no. 11, pp. 1791–1802, 2019.

[62] Q. Kong, I. Sobieraj, W. Wang, and M. Plumbley, "Deep neural network baseline for dcase challenge 2016," *Proceedings of DCASE 2016*, 2016.

[63] *Attention? Attention!* https://lilianweng.github.io/posts/2018-06-24-attention/.

[64] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[65] K. Xu *et al.*, "Show, attend and tell: Neural image caption generation with visual attention," in *International conference on machine learning*, 2015, pp. 2048–2057.

[66] Y. Wang, M. Huang, L. Zhao, *et al.*, "Attention-based lstm for aspect-level sentiment classification," in *Proceedings of the 2016 conference on empirical methods in natural language processing*, 2016, pp. 606–615.

[67] J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," in *Advances in neural information processing systems*, 2015, pp. 577–585.

[68] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7132–7141.

[69] S. Woo, J. Park, J.-Y. Lee, and I. So Kweon, "Cbam: Convolutional block attention module," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 3–19.

[70] J. Fu *et al.*, "Dual attention network for scene segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3146–3154.

[71] C. Yu, K. S. Barsim, Q. Kong, and B. Yang, "Multi-level attention model for weakly supervised audio classification," *arXiv preprint arXiv:1803.02353*, 2018.

[72] Z. Zhang, S. Xu, T. Qiao, S. Zhang, and S. Cao, "Attention based convolutional recurrent neural network for environmental sound classification," in *Chinese Conference on Pattern Recognition and Computer Vision (PRCV)*, Springer, 2019, pp. 261–271.

[73] Z. Zhang, S. Xu, S. Zhang, T. Qiao, and S. Cao, "Learning attentive representations for environmental sound classification," *IEEE Access*, vol. 7, pp. 130 327–130 339, 2019.

[74] Z. Li *et al.*, "Multi-level attention model with deep scattering spectrum for acoustic scene classification," in *2019 IEEE International Conference on Multimedia & Expo Workshops (ICMEW)*, IEEE, 2019, pp. 396–401.

[75] Z. Ren, Q. Kong, K. Qian, M. D. Plumbley, B. Schuller, *et al.*, "Attention-based convolutional neural networks for acoustic scene classification," in *DCASE 2018 Workshop Proceedings*, 2018.

[76] A. Mesaros, T. Heittola, and T. Virtanen, "A multi-device dataset for urban acoustic scene classification," *arXiv preprint arXiv:1807.09840*, 2018.

[77] J. Salamon, C. Jacoby, and J. P. Bello, "A dataset and taxonomy for urban sound research," in *Proceedings of the 22nd ACM international conference on Multimedia*, 2014, pp. 1041–1044.

[78] J. Abeßer, S. I. Mimilakis, R. Gräfe, H. Lukashevich, and I. Fraunhofer, "Acoustic scene classification by combining autoencoder-based dimensionality reduction and convolutional neural networks," in *Proc. of the Detection and Classification of Acoustic Scenes and Events 2017 Workshop (DCASE2017)*, 2017, pp. 7–11.

[79] J.-X. Xu, T.-C. Lin, T.-C. Yu, T.-C. Tai, and P.-C. Chang, "Acoustic scene classification using reduced mobilenet architecture," in *2018 IEEE International Symposium on Multimedia (ISM)*, IEEE, 2018, pp. 267–270.

[80] J. Driedger and M. Müller, "A review of time-scale modification of music signals," *Applied Sciences*, vol. 6, no. 2, p. 57, 2016.

[81] J. Driedger, "Time-scale modification algorithms for music audio signals," *Master's thesis, Saarland University*, 2011.

[82] D. S. Park *et al.*, "Specaugment: A simple data augmentation method for automatic speech recognition," *arXiv preprint arXiv:1904.08779*, 2019.

[83] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "Mixup: Beyond empirical risk minimization," *arXiv preprint arXiv:1710.09412*, 2017.

[84] I. Goodfellow *et al.*, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[85] S. Mun, S. Park, D. K. Han, and H. Ko, "Generative adversarial network based acoustic scene training set augmentation and selection using svm hyper-plane," *Proc. DCASE*, pp. 93–97, 2017.

[86] Q. Kong, Y. Xu, T. Iqbal, Y. Cao, W. Wang, and M. D. Plumbley, "Acoustic scene generation with conditional samplernn," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2019, pp. 925–929.

[87] *Time-stretch audio - MATLAB strechAudio*, https://www.mathworks.com/help/audio/ref/stretchaudio.html.

[88] *SpecAugment: A New Data Augmentation Method for Automatic Speech Recognition*, https://ai.googleblog.com/2019/04/specaugment-new-data-augmentation.html.

[89] H. Wang, Y. Zou, D. Chong, and W. Wang, "Environmental sound classification with parallel temporal-spectral attention," *Proceedings of INTERSPEECH 2020*, 2020.

[90] Y. Tokozume, Y. Ushiku, and T. Harada, "Learning from between-class examples for deep sound recognition," *arXiv preprint arXiv:1711.10282*, 2017.

[91] J. Wang and S. Li, "Self-attention mechanism based system for dcase2018 challenge task1 and task4," in *IEEE AASP Challenge on DCASE 2018 technical reports*, 2018.

[92] Y. Wang, C. Feng, and D. V. Anderson, "A multi-channel temporal attention convolutional neural network model for environmental sound classification," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2021, pp. 930–934.

[93] *An Introduction to Few-Shot Learning*, https://www.analyticsvidhya.com/blog/2021/05/an-introduction-to-few-shot-learning/.

[94] L. Bertinetto, J. F. Henriques, J. Valmadre, P. Torr, and A. Vedaldi, "Learning feed-forward one-shot learners," in *Advances in neural information processing systems*, 2016, pp. 523–531.

[95] L. Fei-Fei, R. Fergus, and P. Perona, "One-shot learning of object categories," *IEEE transactions on pattern analysis and machine intelligence*, vol. 28, no. 4, pp. 594–611, 2006.

[96] A. Tjandra, S. Sakti, and S. Nakamura, "Machine speech chain with one-shot speaker adaptation," *arXiv preprint arXiv:1803.10525*, 2018.

[97] C. H. Lampert, H. Nickisch, and S. Harmeling, "Learning to detect unseen object classes by between-class attribute transfer," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, 2009, pp. 951–958.

[98] H. Xie and T. Virtanen, "Zero-shot audio classification via semantic embeddings," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 1233–1242, 2021.

[99] Y. Wang, Q. Yao, J. T. Kwok, and L. M. Ni, "Generalizing from a few examples: A survey on few-shot learning," *ACM Computing Surveys (CSUR)*, vol. 53, no. 3, pp. 1–34, 2020.

[100] R. Caruana, "Multitask learning," *Machine learning*, vol. 28, no. 1, pp. 41–75, 1997.

[101] Z. Hu, X. Li, C. Tu, Z. Liu, and M. Sun, "Few-shot charge prediction with discriminative legal attributes," in *Proceedings of the 27th International Conference on Computational Linguistics*, 2018, pp. 487–498.

[102] Y. Jia *et al.*, "Caffe: Convolutional architecture for fast feature embedding," in *Proceedings of the 22nd ACM international conference on Multimedia*, 2014, pp. 675–678.

[103] E. Triantafillou, R. Zemel, and R. Urtasun, "Few-shot learning through an information retrieval lens," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[104] A. Graves, G. Wayne, and I. Danihelka, "Neural turing machines," *arXiv preprint arXiv:1410.5401*, 2014.

[105] T. Ramalho and M. Garnelo, "Adaptive posterior learning: Few-shot learning with a surprise-based memory module," *arXiv preprint arXiv:1902.02527*, 2019.

[106] R. Salakhutdinov, J. Tenenbaum, and A. Torralba, "One-shot learning with a hierarchical nonparametric bayesian model," in *Proceedings of ICML Workshop on Unsupervised and Transfer Learning*, JMLR Workshop and Conference Proceedings, 2012, pp. 195–206.

[107] S. Caelles, K.-K. Maninis, J. Pont-Tuset, L. Leal-Taixé, D. Cremers, and L. Van Gool, "One-shot video object segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 221–230.

[108] D. Yoo, H. Fan, V. Boddeti, and K. Kitani, "Efficient k-shot learning with regularized deep networks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, 2018.

[109] M. D. Spivak, *A comprehensive introduction to differential geometry*. Publish or perish, 1970.

[110] G. Koch, R. Zemel, and R. Salakhutdinov, "Siamese neural networks for one-shot image recognition," in *ICML deep learning workshop*, Lille, vol. 2, 2015.

[111] F. Sung, Y. Yang, L. Zhang, T. Xiang, P. H. Torr, and T. M. Hospedales, "Learning to compare: Relation network for few-shot learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1199–1208.

[112] O. Vinyals, C. Blundell, T. Lillicrap, D. Wierstra, *et al.*, "Matching networks for one shot learning," in *Advances in neural information processing systems*, 2016, pp. 3630–3638.

[113] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," in *Advances in neural information processing systems*, 2017, pp. 4077–4087.

[114] *Tutorial #2: few-shot learning and meta-learning*, https://www.borealisai.com/en/blog/tutorial-2-few-shot-learning-and-meta-learning-i/.

[115] *Meta-Learning: Learning to Learn Fast*, https://lilianweng.github.io/posts/2018-11-30-meta-learning/.

[116] T. Gao, X. Han, Z. Liu, and M. Sun, "Hybrid attention-based prototypical networks for noisy few-shot relation classification," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 6407–6414.

[117] S.-Y. Chou, K.-H. Cheng, J.-S. R. Jang, and Y.-H. Yang, "Learning to match transient sound events using attentional similarity for few-shot sound recognition," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2019, pp. 26–30.

[118] A. Mesaros, T. Heittola, and T. Virtanen, "Tut database for acoustic scene classification and sound event detection," in *2016 24th European Signal Processing Conference (EUSIPCO)*, IEEE, 2016, pp. 1128–1132.

[119] Y. Wang and D. V. Anderson, "Hybrid attention-based prototypical networks for few-shot sound classification," in *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2022, pp. 651–655.

[120] E. Fonseca, X. Favory, J. Pons, F. Font, and X. Serra, "FSD50K: An open dataset of human-labeled sound events," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 30, pp. 829–852, 2022.

[121] S. Laine and T. Aila, "Temporal ensembling for semi-supervised learning," *arXiv preprint arXiv:1610.02242*, 2016.

[122] O. Sagi and L. Rokach, "Ensemble learning: A survey," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 8, no. 4, e1249, 2018.

[123] P. Kontschieder, M. Fiterau, A. Criminisi, and S. R. Bulo, "Deep neural decision forests," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1467–1475.

[124] G. Wen, Z. Hou, H. Li, D. Li, L. Jiang, and E. Xun, "Ensemble of deep neural networks with probability-based fusion for facial expression recognition," *Cognitive Computation*, vol. 9, no. 5, pp. 597–610, 2017.

[125] L. Nanni, Y. M. Costa, R. L. Aguiar, R. B. Mangolin, S. Brahnam, and C. N. Silla, "Ensemble of convolutional neural networks to improve animal audio classification," *EURASIP Journal on Audio, Speech, and Music Processing*, vol. 2020, no. 1, pp. 1–14, 2020.

[126] A. Cicone, "Nonstationary signal decomposition for dummies," in *Advances in Mathematical Methods and High Performance Computing*, Springer, 2019, pp. 69–82.

[127] A. Cicone and H. Zhou, "Numerical analysis for iterative filtering with new efficient implementations based on fft," *Numerische Mathematik*, vol. 147, no. 1, pp. 1–28, 2021.

[128] P. Barbe, A. Cicone, W. S. Li, and H. Zhou, "Time-frequency representation of nonstationary signals: The imfogram," *arXiv preprint arXiv:2011.14209*, 2020.

[129] A. Cicone, W. S. Li, and H. Zhou, "New theoretical insights in the decomposition and time-frequency representation of nonstationary signals: The imfogram algorithm," *arXiv preprint arXiv:2205.15702*, 2022.

[130] *Few-shot Bioacoustic Event Detection*, https://dcase.community/challenge2021/task-few-shot-bioacoustic-event-detection.

[131] A. Vaswani *et al.*, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[132] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[133] T. Brown *et al.*, "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.

[134] K. Koutini, J. Schlüter, H. Eghbal-zadeh, and G. Widmer, "Efficient training of audio transformers with patchout," *arXiv preprint arXiv:2110.05069*, 2021.

[135] S. Wyatt *et al.*, "Environmental sound classification with tiny transformers in noisy edge environments," in *2021 IEEE 7th World Forum on Internet of Things (WF-IoT)*, IEEE, 2021, pp. 309–314.

[136] *Tutorial 6: Transformers and Multi-Head Attention*, https://uvadlc-notebooks.readthedocs.io/en/latest/tutorial_notebooks/tutorial6/Transformers_and_MHAttention.html.

[137] Y. Wang, N. J. Bryan, M. Cartwright, J. P. Bello, and J. Salamon, "Few-shot continual learning for audio classification," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2021, pp. 321–325.