MEDICAL SOUND EVENT DETECTION USING AUDIO SPECTROGRAM FOURIER NETWORK

by

K. M. Naimul Hassan
0421062556

MASTER OF SCIENCE
IN
ELECTRICAL AND ELECTRONIC ENGINEERING

Department of Electrical and Electronic Engineering
Bangladesh University of Engineering and Technology

Dhaka, Bangladesh

July 2023

The thesis titled, **"MEDICAL SOUND EVENT DETECTION USING AUDIO SPECTROGRAM FOURIER NETWORK"**, submitted by **K. M. Naimul Hassan**, Roll No.: 0421062556, Session: April 2021, has been accepted as satisfactory in partial fulfillment of the requirement for the degree of Master of Science in Electrical and Electronic Engineering on 25 July 2023.

BOARD OF EXAMINERS

Dr. Mohammad Ariful Haque Professor Dept. of EEE, BUET, Dhaka	Chairman (Supervisor)
Dr. Md. Aynal Haque Professor and Head Dept. of EEE, BUET, Dhaka	Member (Ex-Officio)
Celia Shahnaz Dr. Celia Shahnaz Professor Dept. of EEE, BUET, Dhaka	Member
Dr. Hafiz Imtiaz Associate Professor Dept. of EEE, BUET, Dhaka	Member
Dr. Taufiq Hasan Al Banna Associate Professor Dept. of BME, BUET, Dhaka	Member (External)

Candidate's Declaration

This is to certify that the work presented in this thesis entitled, "MEDICAL SOUND EVENT DETECTION USING AUDIO SPECTROGRAM FOURIER NETWORK", is the outcome of the research carried out by K. M. Naimul Hassan under the supervision of Dr. Mohammad Ariful Haque, Professor, Department of Electrical and Electronic Engineering, Bangladesh University of Engineering and Technology (BUET), Dhaka-1205, Bangladesh.

It is also declared that neither this thesis nor any part thereof has been submitted anywhere else for the award of any degree, diploma, or other qualifications.

Signature of the Candidate

K. M. Naimul Hassan
0421062556

Dedication

To my family

”The measure of intelligence is the ability to change.”

— Albert Einstein

Contents

List of Figures

List of Tables

List of Important Abbreviations

ASFNet	Audio Spectrogram Fourier Network
AST	Audio Spectrogram Transformer
CNN	Convolutional Neural Network
CV	Computer Vision
DFT	Discrete Fourier Transform
FFT	Fast Fourier Transform
mAP	Mean Average Precision
NLP	Natural Language Processing
PANNs	Pretrained Audio Neural Networks
RNN	Recurrent Neural Network
SDR	Source-to-Distortion Ratio
SED	Sound Event Detection
ViT	Vision Transformer

Acknowledgement

I would like to take this opportunity to express my sincere gratitude to all those who have supported me throughout the completion of my thesis.

First and foremost, I would like to express my deepest appreciation to my supervisor, Dr. Mohammad Ariful Haque. His guidance, expertise, and unwavering support have been invaluable to the successful completion of this research. Dr. Haque's profound knowledge in the field, his constructive feedback, and his constant encouragement have played a significant role in shaping the direction and quality of my work. I am truly grateful for his mentorship and the opportunity to learn from him.

I would also like to extend my thanks to the faculty members of the department for their valuable insights, suggestions, and encouragement. Their expertise and dedication to academic excellence have greatly enriched my research experience.

I am indebted to my family, friends, seniors, and juniors for their continuous love, support, and understanding throughout this journey. Their encouragement and belief in my abilities have been a constant source of motivation.

To everyone who has supported me along this journey, thank you from the bottom of my heart. Your contributions and encouragement have made this thesis possible, and I am truly grateful for your presence in my life.

Abstract

Sound event detection (SED) in medical environments is crucial for extracting valuable information from diverse sound events such as coughing, sneezing, sniffling, speech, gasping, and snoring. These events carry vital information for diagnosis, monitoring, and prevention. By utilizing sound events, healthcare professionals can make informed decisions and provide optimal care. Due to the success of Transformer encoder architectures for sound event detection, they seem to be a prudent choice for detecting audio events in hospital settings. However, applying Transformers to medical audio event detection faces two significant challenges. Firstly, there is a severe scarcity of medical audio data, making it difficult to train Transformer models effectively. Secondly, SED models must be computationally efficient to be deployable in resource-limited medical environments. Unfortunately, Transformers have high computational complexity due to the attention mechanism they employ. To tackle these obstacles, this thesis introduces Audio Spectrogram Fourier Network (ASFNet), a novel attention-free Transformer encoder specifically designed for sound event detection in medical environments. ASFNet replaces the attention operation with a simplified Fast Fourier Transform. By employing this technique, ASFNet surpasses other methods, achieving an impressive average mean average precision (mAP) of 0.474 with a 16.76% relative improvement. ASFNet achieves this performance with fewer model parameters and smaller model size, making it a highly efficient and effective solution for detecting medical audio events.

Furthermore, speech-privacy is a critical consideration in medical audio event detection. It is important to separate speech data from audio recordings to protect privacy of the patients when collecting the dataset. While audio source separation techniques can separate speech signals of different speakers, we need to differentiate speech and other medical audio events of the same speaker. Therefore, a custom dataset was prepared and a Wave-U-Net model was trained for separating speech data from medical audio events during data acquisition. Wave-U-Net demonstrates an overall source-to-distortion ratio (SDR) of 11.829 indicating a near-perfect source separation task.

Therefore, the combination of ASFNet and Wave-U-Net has the potential to play a significant role in developing speech-privacy conscious and resource-efficient medical sound event detection or monitoring systems.

Chapter 1

Introduction

1.1 Motivation

Sounds contain valuable information about our surroundings and the physical events occurring in them. Our ability to perceive sound scenes and recognize individual sound sources opens up possibilities for developing signal processing methods to automatically extract this information, which can be applied in various applications. However, while there has been research focused on sound event detection (SED) in urban or environmental contexts [1, 2], there is a noticeable lack of research specifically targeting sound event detection in medical environments.

In medical settings, diverse sound events such as coughing, sneezing, sniffling, speech, gasping, snoring, and others carry vital information that is crucial for diagnosis, monitoring, and prevention. One notable application is in the field of influenza-like-illness (ILI) surveillance, where the detection and quantification of cough events have proven to be valuable [3]. By extracting cough counts, healthcare professionals can track and analyze the prevalence of coughing, which is a significant symptom in ILI cases.

Furthermore, extracting important features from sound events can aid in the identification of patients with chronic cough. Features such as cough frequency, cough count, and the ratio of coughs to speech provide valuable insights into the severity and persistence of a chronic cough condition. These quantitative measures assist in the diagnostic process and inform appropriate treatment strategies.

Sleep apnea encompasses various types, with Obstructive Sleep Apnea (OSA) being the most common and a primary sleep-related breathing disorder. OSA is characterized by recurrent interruptions in breathing during sleep, caused by periodic relaxation of the throat muscles leading to airway obstruction. One prominent indicator of obstructive sleep apnea is the presence of loud snoring. Therefore, the detection of snoring sounds offers a potential means to identify the occurrence of OSA.

Another significant application of sound event detection in medical environments is the monitoring of patients' progress over time. For instance, in the case of a patient with chronic cough prescribed medication, tracking the cough frequency over a specific period allows for the assessment of the drug's effectiveness. A decrease in cough frequency indicates a positive response to the medication, confirming its efficacy. Conversely, if the cough frequency remains persistent or increases, healthcare professionals may consider alternative treatment options or adjustments to the current medication regimen.

Overall, sound event detection in medical environments has the potential to greatly enhance treatment decision-making for healthcare professionals. By harnessing the information embedded in sound events, including coughing and other relevant sounds, medical practitioners can gain valuable insights into patients' conditions, monitor treatment effectiveness, and make informed decisions to provide optimal care.

1.2 Challenges

The detection of medical audio events presents significant challenges for two main reasons. Firstly, the availability of medical audio data for training sound event detection models is extremely limited. Secondly, it is necessary for the sound event detection model to be computationally efficient for deployment in resource-limited medical settings and edge devices.

Previous research has explored audio classification models based on manually designed features [4, 5]. However, with advancements in deep neural networks, it has become possible to develop end-to-end architectures that directly convert spectrograms into event labels [6–8]. Convolutional Neural Networks (CNNs) have gained popularity in this domain. Another approach involves incorporating a self-attention mechanism alongside CNNs to capture extensive global context. Hybrid models combining CNNs and attention have demonstrated state-of-the-art performance in various audio classification tasks [9–12].

While attention-based models have shown success in vision tasks [13–15], the question arises as to whether audio classification still requires CNNs. In response to this, the Audio Spectrogram Transformer (AST), an attention-based model, was introduced and demonstrated superior performance compared to other state-of-the-art models [16].

Given the circumstances, employing AST for sound event detection in medical environments would be a prudent choice. However, Transformers have a higher data requirement for training [13], which poses a challenge as there is typically a lack of audio data in medical environments to meet these demands. Additionally, the attention mechanism in Transformers makes them computationally inefficient, especially for inputs with large context sizes, as the attention operation exhibits quadratic time and space complexity.

Recent works in Natural Language Processing (NLP) and Computer Vision (CV) have addressed this scalability issue by approximating or replacing the attention process [17–23]. Approaches such as kernel approximation, locality-sensitive hashing, and sparsity, low-rank decomposition aim to approximate or replace attention [24]. However, in the context of audio spectrogram transformers, no such approximation or alternative to the attention mechanism has been proposed so far.

Furthermore, ensuring speech-privacy is a crucial aspect to consider in medical audio event detection. When deploying SED algorithms in a medical device, it becomes necessary to separate speech data to protect privacy and prevent potential breaches. Although audio source separation models have been extensively used in general environmental settings and for music source separation tasks, there is a notable scarcity of research focused on utilizing these models in hospital environments.

So, the main challenge of this thesis is to address the following inquiries-

- How can we train a robust SED model using a restricted amount of medical audio data?
- What strategies can be employed to enhance the efficiency of the SED model while maintaining its classification performance?
- How can we guarantee speech privacy in the context of medical audio event detection?

1.3 Objectives of the Thesis

The objectives with specific aims are-

- To detect audio events such as breath, cough, gasp, hiccup, sneeze, sniffling, speech, silence, and throat-clearing in the medical environment.

- To build a data acquisition pipeline for separating speech data from medical audio events in order to ensure speech-privacy.
- To improve the accuracy of medical audio event detection models compared to the state-of-the-art.
- To develop a data-efficient model that can be trained properly with a limited amount of medical audio data.

1.4 Contribution

The contributions of this thesis are as follows-

- We have introduced the Audio Spectrogram Fourier Network (ASFNet), a Transformer encoder architecture specifically designed for sound event detection in medical environments. In the domains of NLP and CV, the Fourier transform is a commonly used operation to approximate or speed up CNNs [25–31], RNNs [32–34], and even transformers [35–37]. Taking inspiration from these ideas, ASFNet deviates from traditional self-attention sublayers and instead incorporates Fourier sublayers, eliminating the need for attention mechanisms.
- We have compared ASFNet with the other state-of-the-art models in terms of both performance and efficiency. ASFNet outperforms the other methods with an average mAP of 0.474 and it is achieved with fewer model parameters and model size.
- We have adapted the Wave-U-Net [38] architecture, an audio source separation model for the data pre-processing pipeline of our medical audio event detection. Wave-U-Net shows a near-perfect speech source separation task under the hospital settings ensuring the speech-privacy.

1.5 Thesis Outline

The rest of this thesis is organized as follows-

Chapter 2 presents the conceptual and theoretical background of deep neural networks including Transformers. It provides a comprehensive guide to the conceptual and theoretical foundations of deep neural networks, with a particular emphasis on Transformers.

Chapter 3 provides a comprehensive survey of existing literature and research efforts that are relevant to medical sound event detection. It aims to present a holistic overview of the advancements, methodologies, and findings in the field, showcasing the existing body of knowledge and identifying the gaps and opportunities for further research.

Chapter 4 presents the speech source separation model, Wave-U-Net. It delves into the details of Wave-U-Net, explaining its underlying architecture and the rationale behind its design choices. The chapter covers the training methodology employed for Wave-U-Net and assesses the model's performance, discussing metrics and evaluations that demonstrate its effectiveness in separating speech sources and maintaining the confidentiality of speech data.

Chapter 5 is dedicated to the exploration of the Audio Spectrogram Fourier Network (ASFNet), a proposed model for medical sound event detection. The chapter covers important aspects such as the model's architecture, the methodology employed for training, and its performance evaluation. It begins by providing a detailed overview of the ASFNet model architecture, highlighting its design principles and key components. The chapter then proceeds to discuss the experimental setup used during the training phase of ASFNet. Following that, the chapter presents an evaluation of the model's performance and efficiency. Additionally, the chapter includes an ablation study, which investigates the impact of incorporating Fourier sublayers instead of self-attention sublayers in ASFNet.

Chapter 6 provides the conclusive remarks and highlights the potential directions for future research. This chapter serves as a summary of the key findings, insights, and contributions discussed throughout the book. It also reflects on the main outcomes of the research, emphasizing the significance and implications. Additionally, it outlines the prospects of future work, unresolved questions, and potential areas of exploration that could advance the field.

Chapter 2

Deep Learning and Transformers

In this chapter, we will provide an overview of the fundamental concepts and theoretical foundation behind deep neural networks, specifically focusing on Transformers. Initially, brief details of deep neural networks and convolutional neural networks (CNN) are given. Then we describe the attention mechanism- how it works and its applications. Next, we discuss hybrid models comprising CNNs and attention mechanisms. Finally, the architectures and applications of Transformers including Vision Transformer are described in detail.

2.1 Deep Neural Network

Deep neural networks, often referred to as deep learning models, are a class of artificial neural networks that are capable of learning and extracting complex representations from data. They have gained significant attention and popularity in the field of machine learning due to their remarkable ability to solve a wide range of challenging tasks, including image recognition, natural language processing, speech recognition, and more.

The term “deep” in deep neural networks refers to the presence of multiple layers of interconnected nodes, also known as artificial neurons or units. These layers form a hierarchical architecture that enables the network to learn and model intricate patterns and relationships in the input data. Each layer in the network performs a set of computations on the data and passes the transformed information to the next layer, progressively building a higher level of abstraction.

The fundamental building block of a deep neural network is the artificial neuron, also called a perceptron. A neuron takes a set of inputs, applies weights to them, sums them up, and passes the result through an activation function. The activation function introduces non-linearity into the network, enabling it to capture complex relationships between inputs and outputs.

Deep neural networks typically consist of an input layer, one or more hidden layers, and an output layer. The input layer receives the raw data, such as pixel values of an image or word embeddings of a sentence. Each neuron in the hidden layers receives inputs from the previous layer and computes a weighted sum, followed by an activation function. This process continues until the output layer, which produces the final result based on the learned representations.

Training a deep neural network involves an optimization process called backpropagation. During training, the network learns to adjust its internal parameters, including the weights and biases associated with each neuron, in order to minimize a predefined loss function. This is done by iteratively propagating the error signal from the output layer back to the earlier layers and adjusting the weights along the way using gradient descent or related optimization algorithms. This process allows the network to fine-tune its parameters and improve its ability to make accurate predictions or classifications.

One of the key advantages of deep neural networks is their ability to automatically learn feature representations from raw data. In traditional machine learning approaches, engineers and researchers often had to manually engineer relevant features from the data, which can be a time-consuming and challenging task. Deep learning models, on the other hand, can automatically learn hierarchical representations from the data, relieving the need for explicit feature engineering.

2.2 Convolutional Neural Network

Convolutional Neural Network (CNN) [39] is a specialized type of deep neural network designed specifically for processing grid-like data, such as images or videos. CNNs have been instrumental in achieving groundbreaking results in computer vision tasks, such as image classification, object detection, and image segmentation.

The architecture of a CNN is inspired by the organization of the visual cortex in the human brain. It leverages the concept of local receptive fields, shared weights, and hierarchical representations to effectively extract features and capture spatial dependencies in an input image.

The key components of a CNN are convolutional layers, pooling layers, and fully connected layers. Let's explore each of these components in more detail:

- **Convolutional Layers:** These layers perform convolution operations on the input image using a set of learnable filters or kernels. Each filter slides over the image, computing dot products between its weights and the corresponding local regions of the image. This process generates a feature map that represents the presence of specific features or patterns in the input. Multiple filters are typically applied simultaneously to capture different features at different spatial locations.

- **Pooling Layers:** Pooling layers are used to downsample the spatial dimensions of the feature maps, reducing the computational complexity and providing a degree of translation invariance. The most common pooling operation is max pooling [40], which selects the maximum value within a local region of the feature map. Pooling helps to retain the most salient features while reducing the sensitivity to small spatial shifts or distortions in the input.

- **Fully Connected Layers:** These layers are typically placed at the end of the CNN and are responsible for making final predictions based on the extracted features. Fully connected layers connect every neuron in one layer to every neuron in the next layer, similar to traditional neural networks. They capture high-level representations by learning complex combinations of features from the previous layers and provide the network with discriminative power for classification or regression tasks.

During the training process, CNNs employ backpropagation [41] and gradient descent [42] to optimize their weights and learn meaningful feature representations. The network is trained by comparing its predicted outputs with the ground truth labels, using a loss function such as cross-entropy. The gradients of the loss function are propagated backward through the network, allowing the weights to be updated based on the error signal.

One of the significant advantages of CNNs is their ability to automatically learn hierarchical representations from raw image data. The early layers of a CNN capture low-level features, such as edges and textures, while deeper layers learn more abstract and high-level representations, such as shapes, objects, or semantic concepts. This hierarchical learning process enables CNNs to effectively model complex visual patterns and achieve superior performance in various computer vision tasks.

Furthermore, CNNs have been augmented with additional architectural components to enhance their performance. For instance, architectures like ResNet [43], DenseNet [44], and Inception [45] incorporate skip connections, bottleneck layers, and parallel convolutions to alleviate the vanishing gradient problem and improve gradient flow, enabling the training of even deeper networks.

The success of CNNs can be attributed to their ability to exploit local spatial correlations, share weights to reduce the number of parameters, and hierarchically learn representations. These models have been applied not only in image analysis but also in domains like natural language processing and audio processing.

2.3 Attention Mechanism

An attention function [46] is a mapping that takes a query and a collection of key-value pairs as inputs and produces an output. In this mapping, the query, keys, values, and output are represented as vectors. The output is calculated by taking a weighted sum of the values, where the weight assigned to each value is determined by a compatibility function that compares the query with its corresponding key.

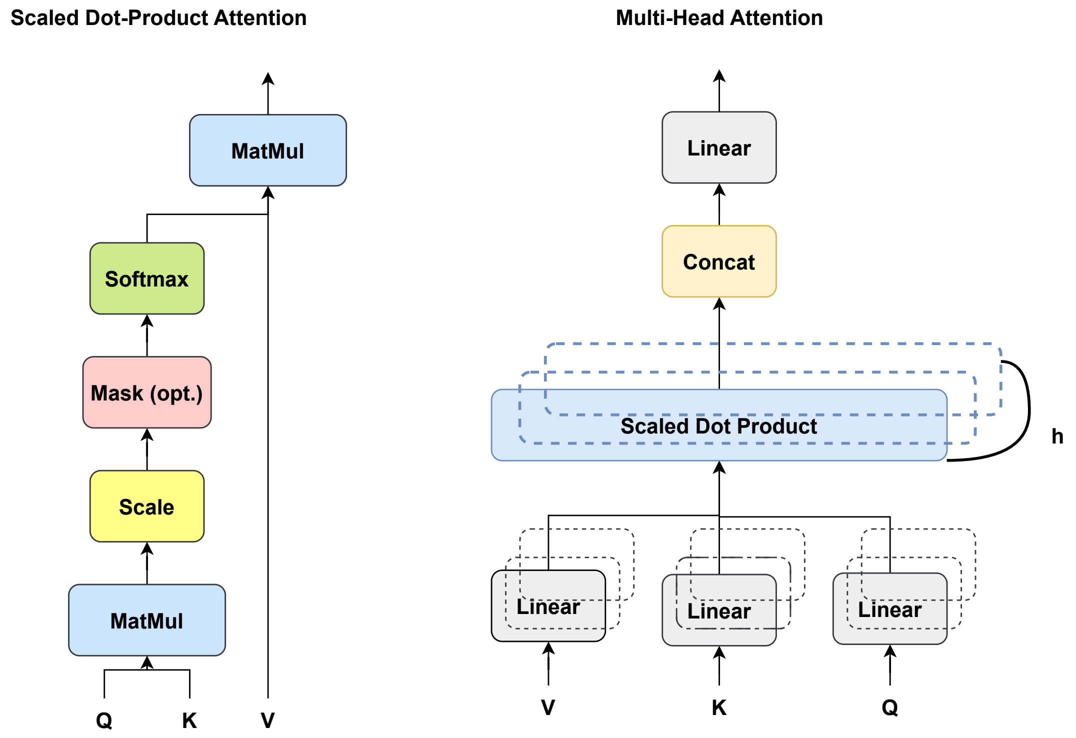

Figure 2.1: (left) Scaled Dot-Product Attention. (right) Multi-Head Attention consists of several attention layers running in parallel.

2.3.1 Scaled Dot-Product Attention

The attention mechanism, known as "Scaled Dot-Product Attention" (Fig. 2.1), operates on queries and keys of size d_k and values of size d_v. It calculates the dot product between each query and key, divides the result by $\sqrt{d_k}$, and applies a softmax function to obtain weights for the values.

In practical applications, we perform the attention function on a batch of queries, which are organized into a matrix Q. The keys and values are also organized into matrices K and V. The output matrix is computed as follows:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{2.1}$$

There are two commonly used attention functions: additive attention and dot-product attention. Dot-product attention lacks the scaling factor of $\frac{1}{\sqrt{d_k}}$. Additive attention, on the other hand, uses a feed-forward network with a single hidden layer to compute the compatibility function. While both mechanisms have similar theoretical complexity, dot-product attention is faster and more memory-efficient due to optimized matrix multiplication implementations.

For small values of d_k, the performance of the two mechanisms is comparable. However, as d_k increases, additive attention outperforms dot-product attention without scaling. This discrepancy may be attributed to the dot products growing significantly in magnitude for larger d_k values, resulting in the softmax function entering regions with extremely small gradients. To address this issue, the dot products are scaled by $\frac{1}{\sqrt{d_k}}$.

2.3.2 Multi-Head Attention

Instead of utilizing a single attention function with d_{model}-dimensional keys, values, and queries, the advantages of employing h linear projections on the queries, keys, and values have been discovered. These projections, learned individually, result in dimensions d_k, d_k, and d_v, respectively. Each of the projected query, key, and value versions undergoes the attention function in parallel, producing output values of dimension d_v. These outputs are then concatenated and projected once more, culminating in the final values depicted in Fig. 2.1. Multi-head attention enables the model to simultaneously attend to information from diverse representation subspaces and various positions. Averaging, which occurs with a single attention head, hinders this capability.

The formulation for multi-head attention is as follows:

$$MultiHead(Q, K, V) = Concat(head_1, ..., head_h)W^O \tag{2.2}$$

$$where\ head_i = Attention(QW_i{}^Q, KW_i{}^K, VW_i{}^V) \tag{2.3}$$

The projections are parameter matrices: $W_i{}^Q \in \mathbb{R}^{d_{model} \times d_k}$, $W_i{}^K \in \mathbb{R}^{d_{model} \times d_k}$, $W_i{}^V \in \mathbb{R}^{d_{model} \times d_v}$, and $W_i{}^O \in \mathbb{R}^{hd_v \times d_{model}}$.

2.3.3 Self-Attention

Self-attention, also known as intra-attention, is an attention mechanism that establishes connections between different positions within a single sequence to generate a representation of the sequence. In self-attention, each element in the sequence attends to other elements within the same sequence. It captures dependencies between different positions or elements within the input sequence. Self-attention allows for capturing long-range dependencies and modeling interactions between different parts of the sequence. It has proven to be effective in various tasks such as reading comprehension, abstractive summarization, textual entailment, and learning task-independent sentence representations [47–50].

2.3.4 Advantages and Applications of Attention

The attention mechanism is a key component in modern deep learning models that has greatly improved the performance of various tasks, particularly in the fields of natural language processing (NLP) and computer vision. It enables models to focus on specific parts of the input data that are deemed more relevant or informative for the task at hand.

At its core, the attention mechanism allows the model to assign different weights or importance values to different elements in the input. These weights are dynamically computed based on the context and content of the data, allowing the model to selectively attend to relevant information. The attention mechanism has its roots in neuroscience, inspired by the human cognitive process of selectively focusing on particular aspects of the environment.

In the context of NLP, attention mechanisms have gained significant prominence in tasks such as machine translation, text summarization, question answering, and sentiment analysis. They have also been successfully applied to computer vision tasks like image captioning, image generation, and object recognition.

The benefits of attention mechanisms include improved model interpretability, as the attention weights indicate which elements the model is focusing on, and enhanced performance in tasks that involve long or complex sequences, where the model needs to selectively attend to relevant parts.

Attention mechanisms have become a fundamental component in state-of-the-art models, such as Transformers, which have achieved remarkable results in various NLP tasks. They have also contributed to significant advancements in computer vision tasks, where attention mechanisms can attend to different spatial regions of an image and generate descriptive captions.

In summary, attention mechanisms have revolutionized deep learning models by enabling selective focus on relevant parts of the input data. By assigning attention weights and attending to important elements, models can better capture dependencies, improve performance, and gain interpretability in complex tasks across NLP and computer vision domains.

2.4 CNN+Attention Hybrid Network

CNN+Attention hybrid models combine the strengths of Convolutional Neural Networks (CNNs) and attention mechanisms to improve the performance of various tasks, particularly in the field of computer vision and natural language processing.

Convolutional Neural Networks are powerful models for extracting spatial features from grid-like data such as images. They excel at capturing local patterns and hierarchically learning representations through convolutional and pooling layers. However, CNNs treat all regions of the input equally, which may not be ideal when certain regions or elements are more important than others.

On the other hand, attention mechanisms provide a way to selectively focus on relevant parts of the input, allowing the model to allocate more attention to specific regions or elements that are crucial for the task at hand. Attention mechanisms have been successful in tasks involving sequential data, such as machine translation and text summarization, by attending to different words or phrases based on their relevance.

In CNN+Attention hybrid models, attention mechanisms are integrated into CNN architectures to enhance their performance. This combination allows the model to benefit from both the spatial feature extraction capability of CNNs and the selective attention mechanism. Here's a high-level overview of how CNN+Attention hybrid models work:

- **Convolutional Layers:** The initial layers of the model typically consist of convolutional layers to extract spatial features from the input data. These layers apply filters to the input, capturing local patterns and generating feature maps that represent different aspects of the data.

- **Attention Mechanism:** The attention mechanism is introduced to the model to selectively focus on relevant regions or elements within the extracted features. Attention weights are calculated based on learned parameters and applied to the feature maps, assigning higher weights to more important regions and lower weights to less relevant ones.

- **Aggregation:** The attended feature maps are then aggregated to produce a compact representation that captures the most salient information. This aggregation can be performed through pooling operations or by using attention-based weighted sums.

- **Fully Connected Layers:** Following the aggregation step, fully connected layers are often employed to further process the attended features and make predictions based on the task requirements. These layers can perform classification, regression, or any other relevant operations.

The integration of attention mechanisms into CNNs enables the model to selectively attend to relevant parts of the input, giving more emphasis to important features or regions. This improves the model's ability to focus on the most informative aspects of the data, leading to enhanced performance in tasks such as image captioning, image classification, visual question answering, and text-to-image synthesis.

Overall, CNN+Attention hybrid models leverage the strengths of both CNNs and attention mechanisms to improve performance in various computer vision and natural language processing tasks. By integrating attention mechanisms, these models can selectively attend to relevant regions or elements, leading to more accurate predictions and a better understanding of the input data.

2.5 Transformer

Transformer [46] is based on the concept of self-attention mechanisms, which allow it to capture relationships between different words or tokens in a sequence. Unlike traditional recurrent neural networks (RNNs) or convolutional neural networks (CNNs),

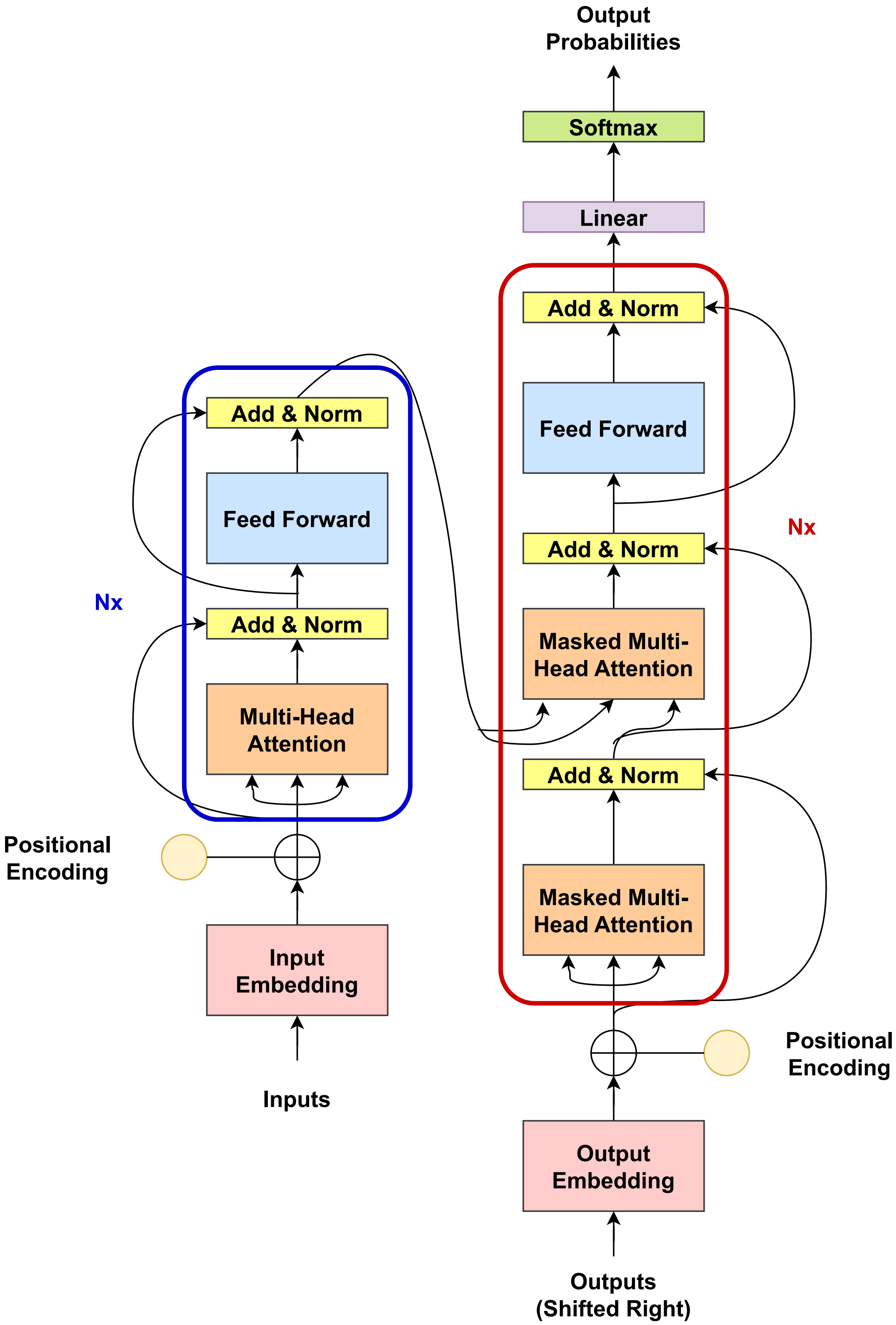

Figure 2.2: Transformer Architecture.

the Transformer does not rely on sequential processing or fixed-size convolutional filters. Instead, it processes the entire sequence of input tokens in parallel, making it highly parallelizable and efficient for both training and inference. The architecture of the Transformer is shown in Fig. 2.2.

2.5.1 Encoder and Decoder Stacks

The encoder (left half of Fig. 2.2) consists of a series of N identical layers, each containing two sub-layers. The first sub-layer is a multi-head self-attention mechanism, while the second sub-layer is a simple, fully connected feed-forward network that operates on each position in the sequence. To ensure smooth gradient flow during training, residual connections [43] are incorporated around both sub-layers, followed by layer normalization [51]. This means that the output of each sub-layer is obtained by applying layer normalization to the sum of the original input and the output of the corresponding sub-layer function.

The decoder (left half of Fig. 2.2) is also constructed using a series of N identical layers. Each layer contains three sub-layers. In addition to the two sub-layers present in the encoder, the decoder includes an additional sub-layer that performs multi-head attention over the encoder stack's output. Similar to the encoder, residual connections are incorporated around each sub-layer, followed by layer normalization. To maintain the autoregressive property, the self-attention sub-layer in the decoder stack is modified. This modification prevents positions from attending to subsequent positions, ensuring that each prediction at position i only depends on the known outputs at positions less than i. This masking, along with the fact that the output embeddings are shifted by one position, guarantees the desired dependency structure in the decoder's predictions.

2.5.2 Attention

The Transformer model utilizes multi-head attention in three distinct manners:

- For "encoder-decoder attention" layers, the queries originate from the previous decoder layer, while the memory keys and values come from the encoder's output. This enables each position in the decoder to attend to all positions in the input sequence. This approach resembles the typical encoder-decoder attention mechanisms found in sequence-to-sequence models [52–54].
- The encoder employs self-attention layers where the keys, values, and queries are derived from the same source, which in this case is the output of the preceding layer in the encoder. This allows each position in the encoder to attend to all positions in the previous encoder layer.

- Similarly, the decoder employs self-attention layers that enable each position in the decoder to attend to all positions in the decoder, including and preceding that specific position. To maintain the auto-regressive property, it is crucial to prevent the flow of information from future positions to past positions in the decoder. This is achieved by incorporating masking within the scaled dot-product attention. Specifically, all elements in the input to the softmax function that correspond to invalid connections are masked out (assign a value of ∞), thereby preserving the desired information flow. Please refer to Fig. 2.1 for a visual representation of this process.

2.5.3 Feed-Forward Networks

In addition to the attention sub-layers, each layer in both the encoder and decoder of the Transformer incorporates a fully connected feed-forward network. This network operates independently and uniformly in each position in the sequence. It consists of two linear transformations separated by a ReLU activation function. The feed-forward network is defined as,

$$FFN(x) = max(0, xW1 + b1)W2 + b2 \tag{2.4}$$

Although the linear transformations remain consistent across different positions, they utilize distinct parameters at each layer. Alternatively, this can be understood as two 1-dimensional convolutions.

2.5.4 Embeddings and Softmax

Like other models that transform sequences, learned embeddings are utilized in Transformers to convert the input and output tokens into vectors of dimension d_{model}. Additionally, standard learned linear transformations and softmax functions are employed to convert the decoder output into predicted probabilities for the next token. Weight sharing is adopted by utilizing the same weight matrix for both the embedding layers and the linear transformation preceding the softmax. This weight-sharing approach is akin to the method described in [55]. Notably, the weights are scaled in the embedding layers by $\sqrt{d_{model}}$.

2.5.5 Positional Encoding

As Transformer lacks recurrence and convolutional layers, it requires additional mechanisms to capture the sequential order of tokens. To address this, "positional encodings" is introduced into the input embeddings at the lowermost layers of both the encoder and decoder stacks. These positional encodings serve to convey information about the relative or absolute positions of the tokens within the sequence. Importantly, the positional encodings have the same dimensionality (model) as the embeddings, allowing them to be added together. There exist various options for implementing positional encodings, including both learned and fixed alternatives [54].

2.6 Vision Transformer

Vision Transformer (ViT) [13], is a variation of the Transformer model designed specifically for computer vision tasks like image classification and object detection. Unlike its original purpose in natural language processing, the Vision Transformer has been modified to handle visual data. The model architecture of ViT is shown in Fig. 2.3.

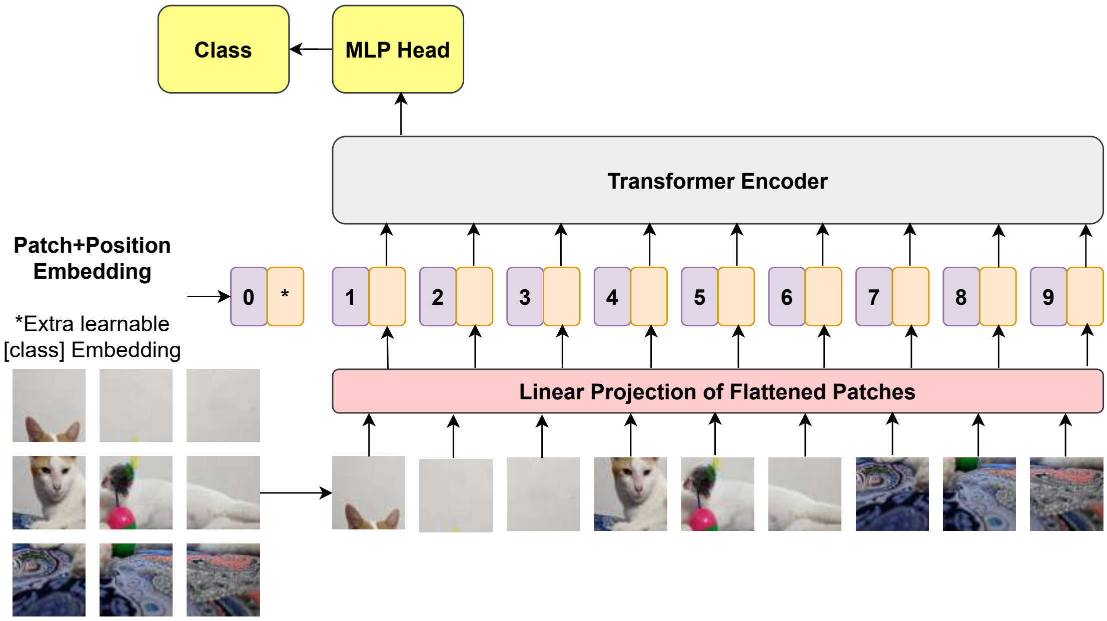

Figure 2.3: Vision Transformer Architecture.

The main concept of the Vision Transformer involves dividing the input image into a grid of patches with consistent sizes. These patches are then considered as sequence tokens and undergo linear embedding to generate token embeddings. These embeddings serve as input for the Transformer encoder. By utilizing self-attention mechanisms inspired by the original Transformer, the model can effectively capture spatial relationships and dependencies among the image patches.

The standard Transformer model operates on 1D sequences of token embeddings. To handle 2D images, the image is reshaped, denoted as $x \in \mathbb{R}^{H \times W \times C}$, into a sequence of flattened 2D patches $x_p \in \mathbb{R}^{N \times (P^2 \cdot C)}$. Here, (H, W) represents the original image resolution, C is the number of channels, (P, P) denotes the resolution of each image patch, and $N = HW/P^2$ represents the resulting number of patches. This N value serves as the effective input sequence length for the Transformer. Throughout all layers of the Transformer, a constant latent vector size D is used. Hence, we flatten the patches and map them to D dimensions using a trainable linear projection (see Eq. 2.5). These mapped patches are referred to as patch embeddings.

$$z_o = [x_{class}; {x_p}^1 E; {x_p}^2 E; ...; {x_p}^N E] + E_{pos},\ E \in \mathbb{R}^{(P^2 \cdot C)},\ E_{pos} \in \mathbb{R}^{(N+1) \times D} \tag{2.5}$$

$$z'_l = MSA(LN(z_{l-1})) + z_{l-1},\ l = 1...L \tag{2.6}$$

$$z_l = MLP(LN(z'_l)) + z'_l,\ l = 1...L \tag{2.7}$$

$$y = LN({z^0}_L) \tag{2.8}$$

Similar to the [class] token in BERT, a learnable embedding is prepended to the sequence of embedded patches $({z_L}^0 = x_{class})$. The state of this embedding at the output of the Transformer encoder $({z_L}^0)$ serves as the image representation y (Eq. 2.8). During both pre-training and fine-tuning, a classification head is attached to $({z_L}^0)$. The classification head is implemented as a multi-layer perceptron (MLP) with one hidden layer during pre-training and as a single linear layer during fine-tuning.

To retain positional information, we add position embeddings to the patch embeddings. We use standard 1D position embeddings, as we have not observed significant performance improvements when using more advanced 2D-aware position embeddings. The resulting sequence of embedding vectors is then fed into the encoder.

The Transformer encoder, as described by [46], consists of alternating layers of multi-headed self-attention and multi-layer perceptron (MLP) blocks (Equations 2.6 and 2.7). Layer normalization (LN) is applied before each block, and residual connections are added after each block, following the approach of [56] and [57].

The Vision Transformer (ViT) exhibits significantly fewer image-specific biases compared to Convolutional Neural Networks (CNNs). In CNNs, properties such as locality, two-dimensional neighborhood structure, and translation equivariance are inherently embedded in each layer throughout the entire model. However, in ViT, only the Multi-Layer Perceptron (MLP) layers possess local and translationally equivariant properties, while the self-attention layers operate on a global scale. The utilization of a two-dimensional neighborhood structure is minimal, primarily occurring at the beginning of the model through patch-based image segmentation and during fine-tuning to adjust position embeddings for images of varying resolutions. In ViT, the position embeddings at the initialization stage do not contain any information regarding the two-dimensional positions of the patches. Consequently, all spatial relationships between the patches must be learned from scratch during training. This minimal reliance on image-specific biases allows ViT to capture more diverse and flexible patterns in the data, making it suitable for a wide range of visual tasks.

Instead of using raw image patches, an alternative approach involves creating the input sequence from feature maps generated by a Convolutional Neural Network (CNN). In this hybrid model, the patch embedding projection E (Equation 1) is applied to patches extracted from a CNN feature map. In a specific scenario, the patches can have a spatial size of 1x1. This implies that the input sequence is formed by flattening the spatial dimensions of the feature map and projecting it to match the dimensions required by the Transformer model. The classification input embedding and position embeddings are then added to the sequence following the same procedure as described earlier.

2.7 Summary

In this chapter, we aimed to offer the foundational knowledge that supported the development of this research. CNNs and CNN+attention hybrid models initially gained popularity demonstrating state-of-the-art performances. But Transformers have been surpassing those and giving the research of deep learning a new direction specifically in the domains of computer vision and natural language processing.

Chapter 3

Literature Review

In this chapter, we will discuss the existing works related to the thesis. First, we will discuss the previous works on audio classification in medical environments. Then we will discuss different state-of-the-art algorithms and models including CNNs, hybrid networks, and Transformers used in audio classification problems. Finally, we present the related works on the Fourier transform in approximating or speeding up neural networks.

3.1 Audio Classification in Medical Environments

While there is a significant body of research focusing on sound event detection in urban or environmental contexts, there is a noticeable dearth of studies specifically addressing audio event detection in medical settings.

The paper [3] introduces FluSense, an innovative platform designed for syndromic surveillance of influenza-like illness in hospital waiting areas. The system utilizes contactless sensing technologies, such as microphones and thermal sensors, to capture and analyze environmental data, including cough sounds and crowd movements. By leveraging machine learning algorithms, FluSense can detect and track flu-like symptoms in real time without the need for direct contact with individuals. The paper provides a detailed description of the system's architecture and design, emphasizing its cost-effectiveness and unobtrusive surveillance capabilities in high-traffic areas. Field deployment of the system in a hospital waiting area is presented to demonstrate its effectiveness in detecting and monitoring influenza-like illness patterns. This research contributes to the development of non-invasive and innovative surveillance platforms for public health monitoring and disease control.

The authors in [58] focus on the development and implementation of an audio event detection system specifically tailored for in-home care settings. The system aims to monitor and detect various audio events relevant to the well-being and safety of individuals receiving care at home. The paper addresses the challenges associated with audio event detection in this context, including background noise and the need for real-time processing. It provides an overview of the system's architecture, encompassing audio data collection, feature extraction, and machine learning algorithms employed for event detection. The research also includes a performance evaluation of the system and highlights its potential applications in enhancing the quality of in-home care services. This work contributes to the advancement of audio-based monitoring systems that can improve the safety and care provided to individuals in a homecare environment.

The authors in [59] propose a study focused on automating the detection of conversational pauses in audio recordings of serious illness conversations in hospital settings. The research aims to develop a system that can automatically identify and quantify pauses in conversations related to serious illnesses. The study utilizes machine learning techniques to train a model on a dataset of audio recordings, enabling the detection of pauses based on audio features. The research contributes to the field of palliative care by providing a novel approach to objectively measure pauses in conversations, which can aid in assessing communication dynamics and improving the quality of care in hospital settings.

Various recent studies have investigated algorithms for recognizing coughs based on audio signals. For instance, cough recognition models have been trained using Mel-frequency cepstral coefficient (MFCC) in combination with Hidden Markov Model (HMM) [60–62]. Larson et al. and Amoh and Odame have employed spectrogram-based features to train cough recognition models [63, 64]. Additionally, other acoustic features such as LPC [65], Hilbert Marginal Spectrum [66], and Gammatone Cepstral Coefficient [67] have been utilized in conjunction with both static models like Random Forest [68] and Support Vector Machine [69], as well as temporal models like Hidden Markov Model and Recurrent Neural Network. Furthermore, more recent studies have explored various architectures of Convolutional Neural Networks (CNN) [63]. The study presented in [70] study focuses on the detection of cough events in audio recordings using moment theory.

The authors in [71] discuss the use of audio-processing techniques for cough and respiratory sound analysis, which can aid in early detection and monitoring of COVID-19 symptoms. The research also explores the application of signal processing techniques for analyzing physiological signals and biosensor data, which can provide valuable insights into the progression of the disease. Furthermore, the study delves into the use of speech and language processing techniques for COVID-19-related tasks, such as automatic speech recognition for speech-based diagnostics and sentiment analysis for assessing public opinions and emotions during the pandemic.

3.2 CNNs for Audio Classification Problems

Earlier studies have investigated audio classification models that rely on handcrafted features [4, 5]. However, the progress in deep neural networks has enabled the creation of comprehensive systems that directly associate spectrograms with event labels. In this field, Convolutional Neural Networks (CNNs) have emerged as a popular choice.

The authors in [6] propose a technique to enhance the representation of speech soundwaves by utilizing restricted Boltzmann machines (RBMs). These RBMs are trained on a substantial dataset of speech sounds, enabling them to extract significant features from the sound waves. This process aims to capture crucial information that can be utilized in speech recognition tasks.

The authors in [7] propose the adoption of an end-to-end learning framework for music audio analysis, where the entire process is directly learned from raw audio data. The authors emphasize the utilization of deep learning architectures, particularly convolutional neural networks (CNNs) and recurrent neural networks (RNNs), which enable automatic feature extraction from raw audio and the learning of temporal relationships within the music. The study demonstrates that this approach has the potential to improve the efficiency and effectiveness of music analysis tasks.

An end-to-end approach for speech emotion recognition using a deep convolutional recurrent network is introduced in [8]. Unlike traditional approaches that require manual feature engineering, this method automatically extracts relevant features from raw speech data. By doing so, it demonstrates the potential for achieving competitive performance in speech emotion recognition tasks.

In order to improve the capability of capturing broader contextual information, an alternative approach integrates self-attention mechanisms with convolutional neural networks (CNNs). These hybrid models, which combine CNNs with attention, have shown exceptional performance in a range of audio classification tasks, surpassing previous benchmarks.

The authors in [9] present PANNs, a collection of large-scale pretrained audio neural networks developed specifically for audio pattern recognition. The authors present the architectural design and training methodology of PANNs, emphasizing its scalability and effectiveness in addressing a wide range of audio classification tasks. By leveraging convolutional neural networks (CNNs) and training on a substantial amount of labeled audio data, the PANNs model is capable of handling extensive datasets efficiently. The training process follows a two-step approach: initial pretraining on a large dataset with audio tags, followed by fine-tuning on specific audio classification tasks. Notably, the scalability of PANNs enables it to effectively process datasets comprising millions of audio samples. Moreover, the pretrained nature of the model allows for transfer learning, facilitating its application to novel audio classification tasks with limited labeled data.

The paper in [10] presents the PSLA framework for audio tagging, which integrates various techniques such as pre-training, sampling, labeling, and aggregation. The PSLA framework improves the accuracy and robustness of audio tagging tasks, surpassing the performance of baseline methods.

The authors in [11] introduce a specialized streaming keyword spotting method designed for mobile devices. This approach effectively tackles the challenges of processing audio data in real-time under resource constraints. The proposed method achieves accurate keyword spotting performance while operating efficiently on mobile platforms.

An attention pooling-based representation learning method is proposed for speech emotion recognition in [12]. The proposed method utilizes attention mechanisms to improve the discriminative power of learned representations. Experimental results demonstrate that this approach effectively captures emotional cues and achieves competitive performance in speech emotion recognition tasks.

3.3 Transformers for Audio Classification Problems

Since attention-based models have achieved notable success in vision tasks [13–15], it prompts the inquiry of whether the presence of a CNN is still imperative for audio classification. In addressing this question, Gong et al. introduced the Audio Spectrogram Transformer (AST), an attention-based model that showcased exceptional performance surpassing other contemporary models [16]. Audio Spectrogram Transformer (AST) is the first ever convolution-free and purely attention transformer designed mainly for audio classification tasks. The model architecture of AST is illustrated in Fig. 3.1.

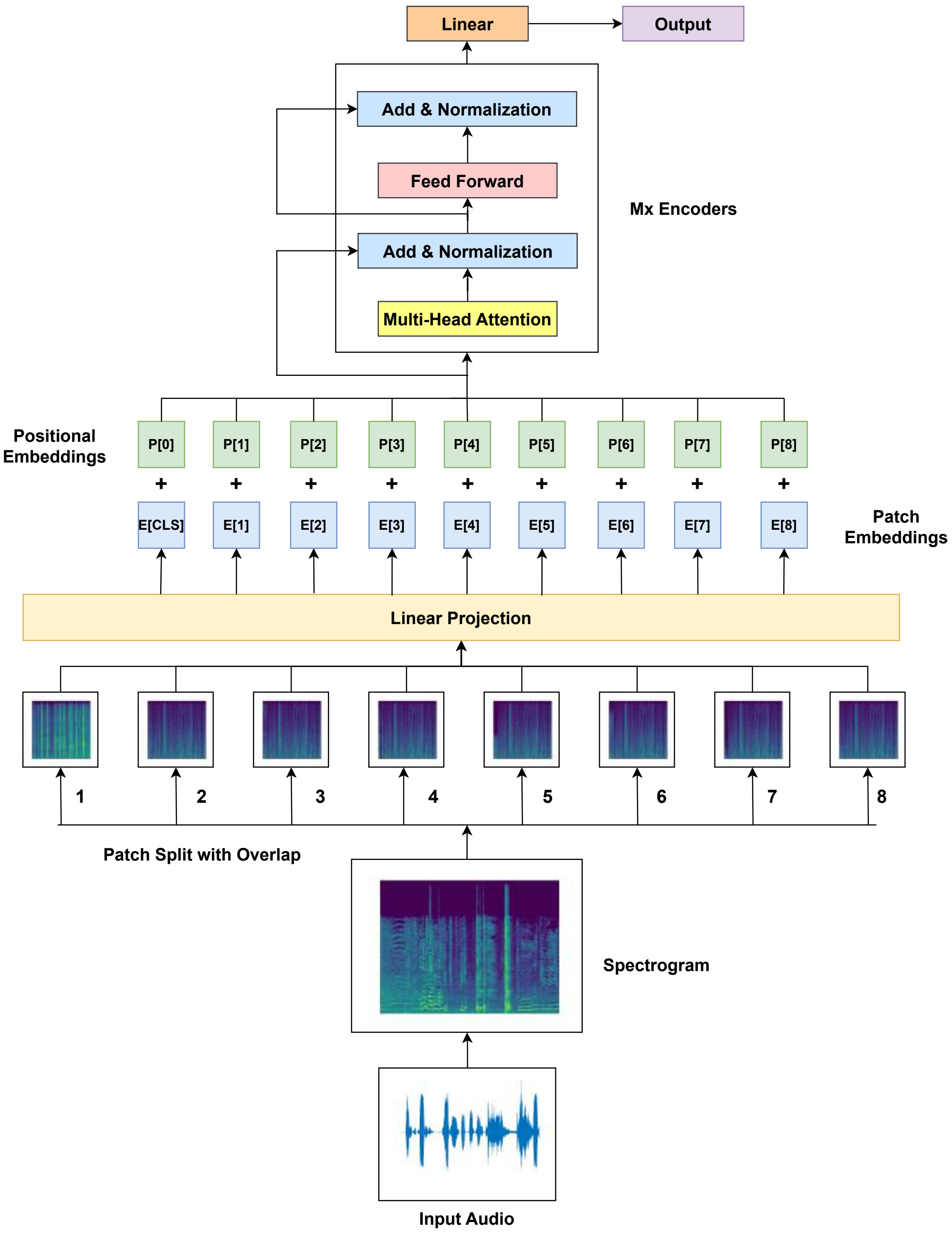

Figure 3.1: Model architecture of AST.

To begin, the original audio waveform is transformed into 128-dimensional log Mel filterbank features using a 25ms Hamming window every 10ms. This creates a spectrogram of size $128 \times 100t$, where t represents the duration of the audio in seconds. The spectrogram is then divided into N 16×16 patches, with an overlap of 6 in both time and frequency dimensions. The number of patches, N, is calculated as $12 \times [\frac{100t-16}{10}]$, and this determines the effective input sequence length for the Transformer. Each 16×16 patch is flattened into a 1D patch embedding of size 768 using a linear projection layer referred to as the patch embedding layer.

Since the Transformer architecture doesn't inherently capture the input order or the temporal order of the patches, a trainable positional embedding of size 768 is added to each patch embedding. This allows the model to capture the spatial structure of the 2D audio spectrogram. A [CLS] token is added at the beginning of the sequence, following a similar approach as in a previous work [72]. The resulting sequence is then fed into the Transformer encoder.

The AST architecture focuses on classification tasks, so only the encoder part of the Transformer is utilized. The original Transformer encoder architecture [46] is used without any modifications. This setup has advantages in terms of ease of implementation and reproducibility, as the standard Transformer architecture is readily available in frameworks like TensorFlow and PyTorch. Additionally, it facilitates transfer learning. The Transformer encoder employed has an embedding dimension of 768, 12 layers, and 12 heads, consistent with [13, 14]. The output of the [CLS] token from the Transformer encoder serves as the representation of the audio spectrogram. Finally, a linear layer with sigmoid activation is used to map the audio spectrogram representation to classification labels.

In terms of design, the patch embedding layer can be seen as a single convolution layer with a large kernel and stride size, while the projection layer in each Transformer block is akin to a 1×1 convolution. However, it should be noted that this design differs from traditional CNNs, which typically consist of multiple layers with smaller kernel and stride sizes. These Transformer models are often referred to as convolution-free to distinguish them from CNNs [13, 14].

Given the circumstances, utilizing the AST for sound event detection in medical environments would be a wise decision. However, Transformers face a drawback in terms of their higher data requirements for effective training [13]. This poses a challenge as medical environments often lack sufficient audio data to meet these demanding training needs. Additionally, transformers suffer from computational inefficiency due to the attention mechanism, which serves as the cornerstone of their architecture. The attention operation's time and space complexity is quadratic in relation to the context size, making it difficult for transformers to process inputs with extensive context sizes.

To tackle this scalability issue, recent research in Natural Language Processing (NLP) and Computer Vision (CV) has concentrated on finding solutions. Various methods, including kernel approximation, locality-sensitive hashing, and techniques like sparsity and low-rank decomposition, have been proposed to approximate or replace the attention process [24].

The authors in [17] present the Reformer as an efficient implementation of the Transformer. The Reformer tackles computational and memory constraints by utilizing locality-sensitive hashing and reversible residual layers. This approach enhances efficiency without compromising performance.

Compressive Transformers are proposed in [18] as an innovative architecture for handling long-range sequence modeling. The proposed approach utilizes iterative compression to efficiently capture global context and overcome the limitations of standard Transformers. Experimental results confirm the effectiveness of the method in capturing long-range dependencies within sequences.

The authors in [19] introduce Sparse Transformers as a method for generating long sequences. By incorporating a sparsity pattern in the attention mechanism, the approach allows for the efficient generation of high-quality sequences while minimizing computational resources.

Linformer, proposed in [20], reduces the quadratic computational complexity of self-attention to linear by approximating the attention matrix using low-rank factorization. This approximation enables the model to capture long-range dependencies while significantly reducing the computational requirements.

The paper in [21] introduces Fast Autoregressive Transformers as a variant that combines the strengths of RNNs and linear attention mechanisms. The model achieves efficient generation of autoregressive sequences with reduced computational complexity by decomposing the self-attention into linear operations.

The Synthesizer model presented in [22] reimagines the self-attention mechanism by introducing a synthesis step, which allows the model to generate new attention weights based on the original self-attention outputs. This synthesis step helps capture more diverse and informative attention patterns. Additionally, an aggregation step is introduced to combine the synthesized attention weights, providing a refined representation of the input sequence.

The authors in [23] introduce Performers which utilize an approximation approach based on random feature maps, enabling efficient and scalable attention computations. By applying the kernelized attention mechanism, Performers achieve linear complexity in both the number of input elements and the attention dimensions.

However, in the context of ASTs, no such approximation or alternative to the attention mechanism has been proposed thus far.

3.4 Fourier Transform in Neural Networks

The significance of Fourier analysis in exploring the universal approximation capabilities of neural networks has been demonstrated in previous studies [73, 74]. In practical applications, discrete Fourier Transforms (DFT) and the Fast Fourier Transform (FFT) have been successfully employed in various signal-processing tasks [75–79].

In the fields of Natural Language Processing (NLP) and Computer Vision (CV), the Fourier transform is a widely used operation for approximating or accelerating computations in CNNs [25–31], RNNs [32–34], and even transformers [35–37]. The Fast Fourier Transform (FFT) has proven to be particularly valuable in neural network architectures, notably in speeding up computations within Convolutional Neural Networks (CNNs) by leveraging the convolution property in the frequency domain. In Recurrent Neural Networks (RNNs), FFTs have been utilized to accelerate training and address issues such as exploding and vanishing gradients. Moreover, FFTs have been employed to approximate dense, linear layers, thereby reducing computational complexity.

3.4.1 Fourier Transform in CNNs

The authors propose a novel approach that combines the strengths of neural networks and FFT in [25]. By leveraging the neural networks' ability to learn complex patterns and features from input data, and exploiting the efficiency of the FFT algorithm for signal processing, they achieve faster and more accurate detection results.

In [26], the authors introduce a method to accelerate the training of convolutional neural networks using FFTs. They transform convolution operations into element-wise multiplications in the frequency domain, resulting in faster convergence and training times without compromising model accuracy. This approach contributes to improving the efficiency of training CNNs, especially for large-scale datasets.

To enhance computational efficiency, the authors in [27] propose the overlap-and-add technique, which breaks down input images and filters into smaller overlapping patches. They further utilize the FFT algorithm to compute convolutions in the frequency domain, resulting in improved computational efficiency.

The concept of Fourier Convolutional Neural Networks (FCNNs) is introduced in [28], where the authors employ the Fourier transform for efficient and effective convolution operations. Their framework incorporates architectural modifications and a phase activation function to handle the complex-valued frequency domain.

In [29], the authors propose an FFT-based approach for deploying deep learning models on embedded systems. They transform convolutional layers into frequency domain operations, achieving computational efficiency and addressing the constraints of embedded platforms.

In the paper [30], the authors present FFT-based split convolutions as a method to accelerate convolutional neural networks. They divide the convolutional filters into smaller sub-filters and perform convolutions in the frequency domain using FFT. By leveraging the convolution theorem and the efficient computation properties of FFT, this approach reduces computational complexity and accelerates inference time.

Lastly, [31] introduces Frequency-Domain Utilization Networks (FDUN) for generating visually appealing images. By operating in the frequency domain and manipulating the image's spectral content while preserving its structural information, the authors enable finer control over the visual characteristics of the generated images.

3.4.2 Fourier Transform in RNNs

The FN-RNN architecture, proposed in [32], utilizes recurrent neural network units that are enhanced with Fourier analysis components. These Fourier components effectively capture the frequency attributes of the input/output data, while the neural network units learn the temporal relationships and patterns present in the sequences. The integration of these two elements in FN-RNNs enables the modeling of both local and global temporal patterns, resulting in enhanced accuracy when dealing with sequential data.

In [33], the Forenet architecture is introduced, which employs recurrent network units augmented with Fourier components. The recurrent units effectively capture the temporal dependencies and patterns within time series data, while the Fourier components accurately capture the frequency characteristics. This combination allows Forenet to accurately model and predict time series data.

In [34], the authors propose Fourier Recurrent Units (FRUs) as an approach to learning long-term dependencies in sequential data. By incorporating Fourier analysis into the recurrent units, FRUs provide a more effective method for capturing both short and long-term dependencies. Consequently, this approach leads to improved performance in tasks involving sequential data.

3.4.3 Fourier Transform in Transformers

In the context of Transformer models, the use of DFTs has been indirectly explored in many prior works.

For instance, the Performer [23] employs random Fourier features to approximate a Gaussian representation of the softmax kernel, leading to the linearization of the Transformer self-attention mechanism.

The authors in [36] employ spectral filters to generate hierarchical features, demonstrating the effectiveness of filtered embeddings across different tasks. Their approach involves segregating Fourier frequencies rather than combining features through the transform.

FNet [37] is an attention-free transformer for NLP tasks where instead of approximating attention, is completely replaced with the Fourier Transform, which serves as an alternative mixing mechanism for hidden representations.

GFNet [80] designed specifically for image classification substitutes the self-attention layer in vision transformers with three primary operations: a 2D discrete Fourier transform, a component-wise multiplication between features in the frequency domain and trainable global filters, and a 2D inverse Fourier transform.

3.5 Summary

In this chapter, we discussed the literature on medical sound event detection and audio classification models. Deep neural networks have greatly advanced these systems, allowing for the direct association of spectrograms with event labels. Convolutional Neural Networks (CNNs) have become popular in this field. Considering the success of attention-based models in CV and NLP, the question arises whether CNNs are still necessary for audio classification. To address this, Audio Spectrogram Transformer (AST), a pure attention-based model was introduced that achieved exceptional performance, surpassing other contemporary models.

Considering the situation, it would be a smart choice to use the AST for sound event detection in medical settings. However, Transformers have a limitation regarding their higher data requirements for effective training. This poses a challenge in medical environments where there may not be enough audio data to meet these demands. Moreover, Transformers can be computationally inefficient due to their attention mechanism, which is a fundamental part of their architecture. Several techniques have been suggested to approximate or substitute the attention process, such as kernel approximation, locality-sensitive hashing, and approaches involving sparsity and low-rank decomposition. Some of the previous studies have highlighted the importance of the Fourier transform in investigating the approximation and acceleration capabilities of Transformers. However, as of now, no approximation or alternative to the attention mechanism has been proposed specifically for ASTs.

Chapter 4

Speech Source Separation using Wave-U-Net

One of the major concerns while capturing medical audio data for SED models is the possible violation of speech-privacy. To ensure the preservation of speech-privacy, in this research, we utilized the concept of source separation. The idea can be visualized in Fig. 4.1. At first, the input audio data, Y, is fed into a speech source separation model. This model separates the sources and we get two output signals. One of the outputs contains the separated speech sound and the other contains the rest of the sources. Each of the outputs has a time duration same as the input audio signal. Finally, the segments containing the other sources are passed through the SED model and it detects the medical audio events. Thus by separating speech sources before feeding into the SED model, speech-privacy is ensured.

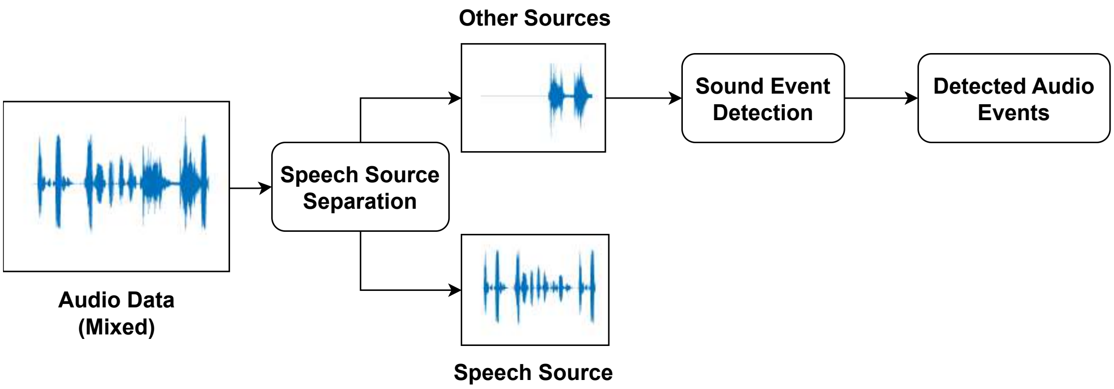

Figure 4.1: Speech source separation prior to SED.

4.1 Model Architecture

In this study, we have used the Wave-U-Net architecture [38] as the speech source separation model. The Wave-U-Net is a convolutional neural network for separating sound sources working directly on raw audio data. It is an adaptation of the U-Net architecture [81] to the one-dimensional time domain for performing end-to-end source separation. Through a series of downsampling and upsampling blocks involving 1D convolutions combined with a downsampling/upsampling process, features are computed on multiple scales/levels of abstraction and time resolution and then combined to make a prediction. The model architecture of Wave-U-Net used in this study is shown in Fig. 4.2.

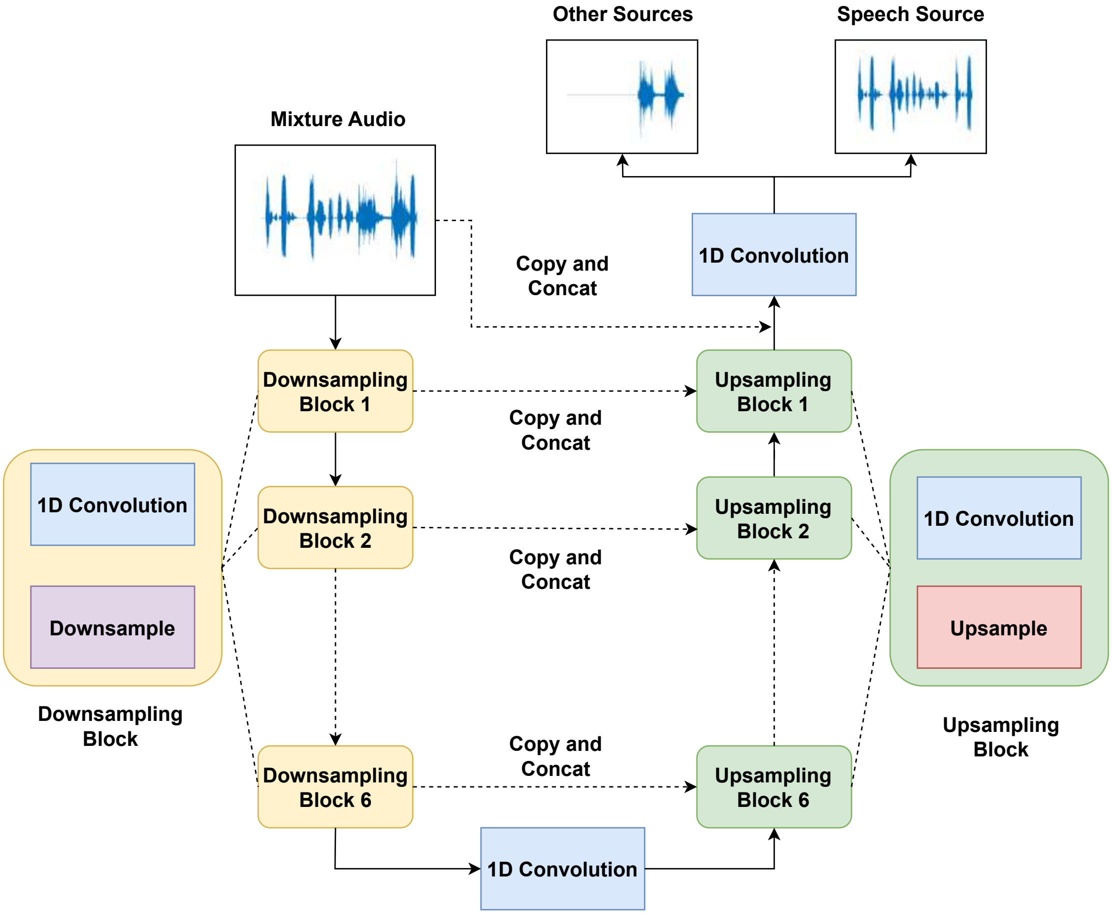

Figure 4.2: Model architecture of Wave-U-Net.

The depth of the model was set to be 6 (6 upsampling blocks and 6 downsampling blocks) considering both performance and training time. If the input audio data has a duration of T s, then two audio data each having the same duration, T are extracted from Wave-U-Net. Each output audio data contains only one sound source (either speech or others).

4.2 Dataset

4.2.1 MAudioSet

To focus specifically on audio events within the medical environment, we utilized a specific portion of AudioSet [82]. This subset, referred to as Medical AudioSet (MAudioSet) in this research, encompasses audio recordings of various events such as breath, cough, gasp, hiccup, sneeze, sniffle, speech, silence, and throat-clearing. These 9 classes serve as the primary categories, while any sound events outside of these classes are categorized under the "etc" class. Consequently, MAudioSet consists of a 10-class dataset comprising audio recordings with multiple labels. It comprises a cumulative duration of around 15 hours. For training and validation purposes, the dataset was divided into 5 folds. A visualization of a sample audio data of MAudioSet including the classes is shown in Fig. 4.3 and a summary of MAudioSet is presented in Table 4.1.

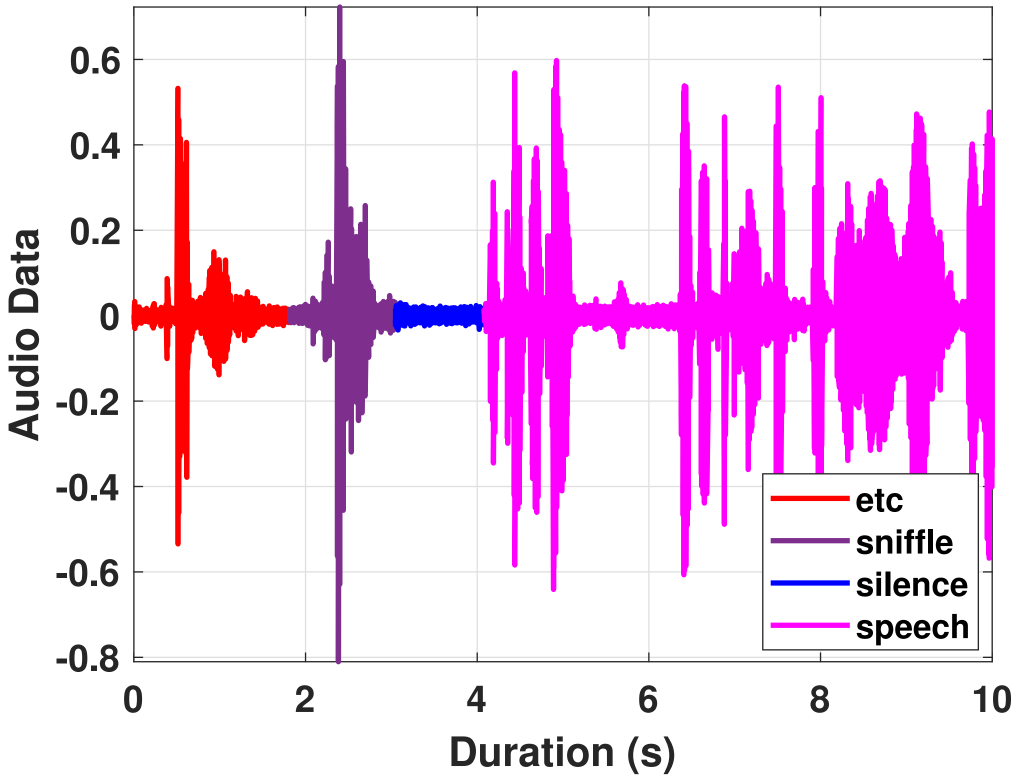

Figure 4.3: Visualization of a sample of MAudioSet.

Table 4.1: Summary of MAudioSet

Label	Class	Total Duration (s)
0	etc	18802.89
1	hiccup	42.48
2	throat-clearing	142.36
3	breathe	287.95
4	gasp	374.72
5	sneeze	654.71
6	sniffle	733.54
7	cough	7662.17
8	speech	20738.63
9	silence	1369.03

4.2.2 Synthetic Soundscapes

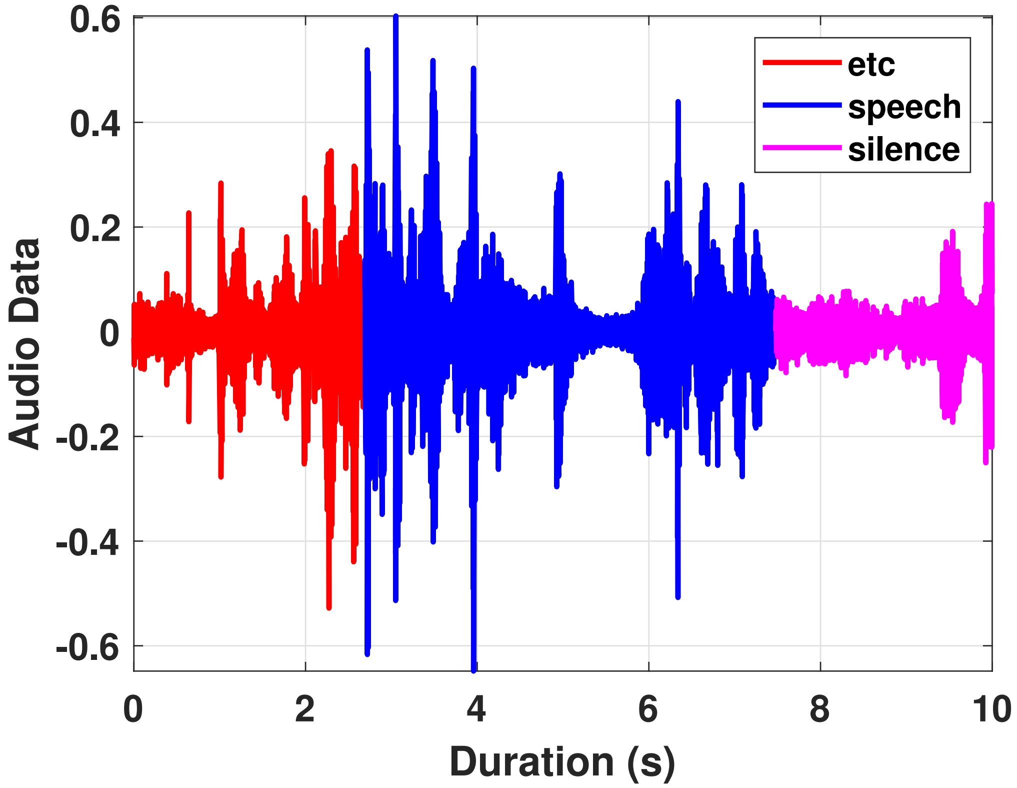

Figure 4.4: Visualization of a synthetic soundscape.

In addition to MAudioSet, a dataset of soundscapes, where each soundscape was created by combining and transforming a set of existing audio files, was used to train the speech source separation model. The existing audio files were taken from TIMIT [83], MU-

SAN [84], and MAudioSet datasets. Speech audio events were taken from TIMIT and MUSAN datasets. Other audio events were taken from the MAudioSet training corpus. Then the soundscapes were generated with the help of Scaper [85], a Python library. Each of the soundscapes contains speech in the background and other sources in the foreground. Each of the soundscapes generated has a time duration of 10s. The soundscapes were generated in such a way that the SNR is uniformly distributed between -10 to 25 dB. A total number of 2400 soundscapes were generated for the experiment. A visualization of a generated soundscape is shown in Fig. 4.4.

4.3 Wave-U-Net Training

Wave-U-Net was developed using the PyTorch framework [86]. NVIDIA GeForce GTX 1650 GPU was used as the hardware accelerator for training the model. The loss function for the training algorithm was *L1 (Least Absolute Deviations). Adam* was used as the optimizer algorithm with an initial learning rate of 0.001. The learning rate was decreased by a factor of 10 whenever the validation loss did not decrease or started increasing for consecutive epochs. The minimum learning rate set was 10^{-10}. The training was stopped early if there was no significant improvement in the validation loss for consecutive epochs. The summary of the training is provided in Table 4.2.

Table 4.2: Summary of the training configurations of WaveUNet

Accelerator	GPU
Loss function	L1
Optimizer	Adam
Initial learning rate	0.001
No. of epochs	2000
Execution time	12h 43m

4.4 Performance Evaluation

4.4.1 Evaluation Strategy and Metric

In this study, Source-to-Distortion Ratio (SDR) was used to evaluate the performance of Wave-U-Net. SDR can be defined as,

$$SDR = 10log_{10}(\frac{||s_{target}||^2}{||e_{interf} + e_{noise} + e_{artif}||^2}) \tag{4.1}$$

where s_{target} is the true source, and e_{interf}, e_{noise} and e_{artif} are error terms for interference, noise, and added artifacts, respectively [87]. A higher SDR indicates a better source separation.

Firstly, the SDR of Wave-U-Net is computed for each of the five folds of the validation subsets of MAudioSet, and finally, the average SDR is calculated.

4.4.2 Results

The performance of Wave-U-Net on the 5-folds of the MAudioSet validation subset is shown in Table 4.3.

Table 4.3: Performance of Wave-U-Net on the MAudioSet validation subset

	SDR					
	Fold-1	Fold-2	Fold-3	Fold-4	Fold-5	Average
Speech	10.15	10.96	10.52	10.77	10.88	**10.656**
Others	12.72	13.56	12.38	13.52	12.84	**13.004**
Overall	11.43	12.26	11.45	12.145	11.86	**11.829**

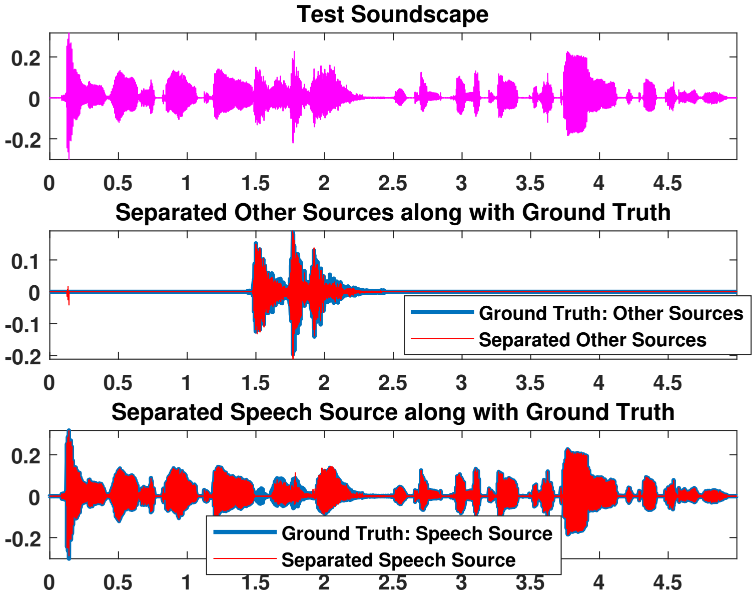

Figure 4.5: Visualization of the output of Wave-U-Net for a test sample along with the ground truth sources.

Wave-U-Net achieves an overall average SDR of 11.829 indicating a near-perfect source separation task. For better understanding, a visualization of the output of the model for a test sample along with the ground truth sources is shown in Fig. 4.5.

4.5 Summary

Wave-U-Net is implemented to preserve speech privacy by separating the speech source from the mixed audio. For a quantitative analysis of how well the model is able to separate the speech source, we have calculated the correlation coefficient between the ground truth speech source and the model's separated speech source. An average correlation coefficient of 0.94 (out of 1) is achieved from the test dataset. This is an indication that the model is able to separate the speech source quite well and thus it preserves speech-privacy. A scatter plot of the correlation coefficients for randomly selected 200 samples is shown in Fig. 4.6.

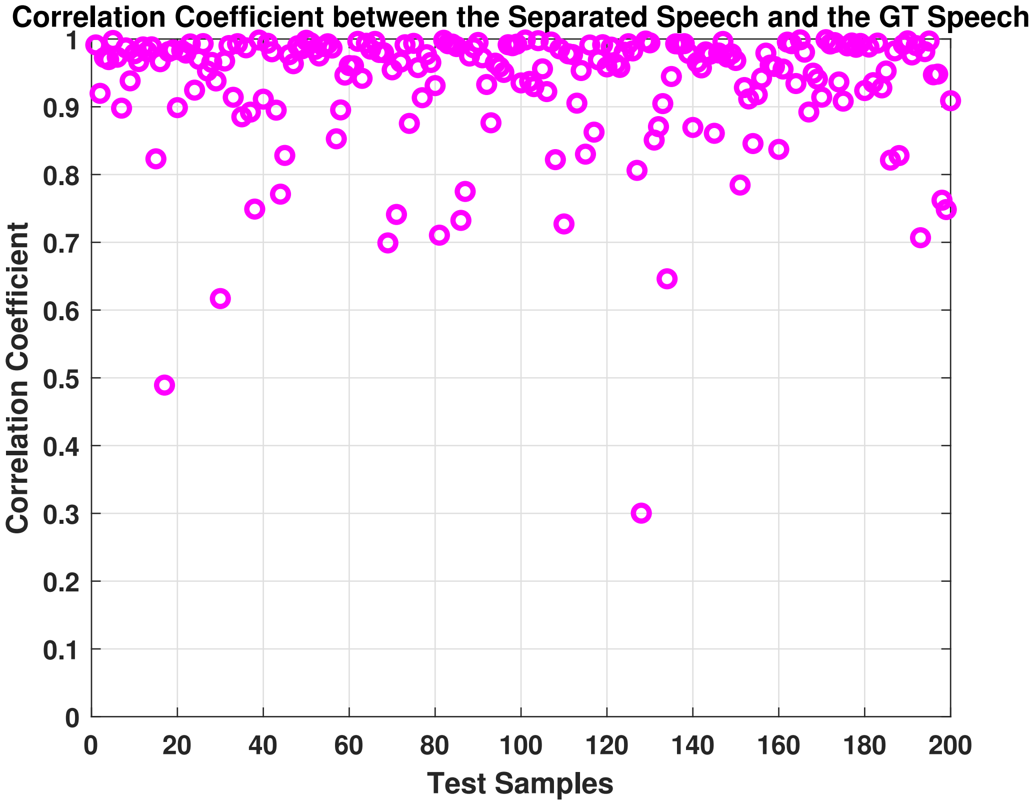

Figure 4.6: Scatter Plot of Correlation Coefficients between the Separated Speech and the Ground Truth (GT) Speech sources.

Chapter 5

Medical Sound Event Detection using ASFNet

In this chapter, we will discuss the details of our proposed Audio Spectrogram Fourier Network (ASFNet) for medical sound event detection. Let, Y be a recorded audio waveform in a medical or hospital environment that consists of multiple audio events. If $A = a_0, a_1, a_2, a_3, ..., a_{n-1}$ is the set of audio events in Y where n is the total number of events, then the goal is to develop an efficient SED model or algorithm to detect these n events. By efficient, we mean both performance improvement and model parameter reduction over state-of-the-art methods.

5.1 Model Architecture

The model architecture of ASFNet is illustrated in Fig. 5.1. ASFNet and AST have similar architectures [16]. The differentiating factor in ASFNet lies in its utilization of a Fourier sublayer, which replaces the self-attention sublayer found in each transformer encoder. Each Fourier sublayer is then followed by a feed-forward sublayer.

5.1.1 Input

Similar to AST, ASFNet operates on raw audio with a duration of t seconds as its input and transforms it into a spectrogram of size $128 \times 100t$. This spectrogram is then divided into N patches of size 16×16, with a 6-unit overlap in both the time and frequency dimensions. N represents the total number of patches and can be calculated as $N = 12 \times [\frac{100t-16}{10}]$.

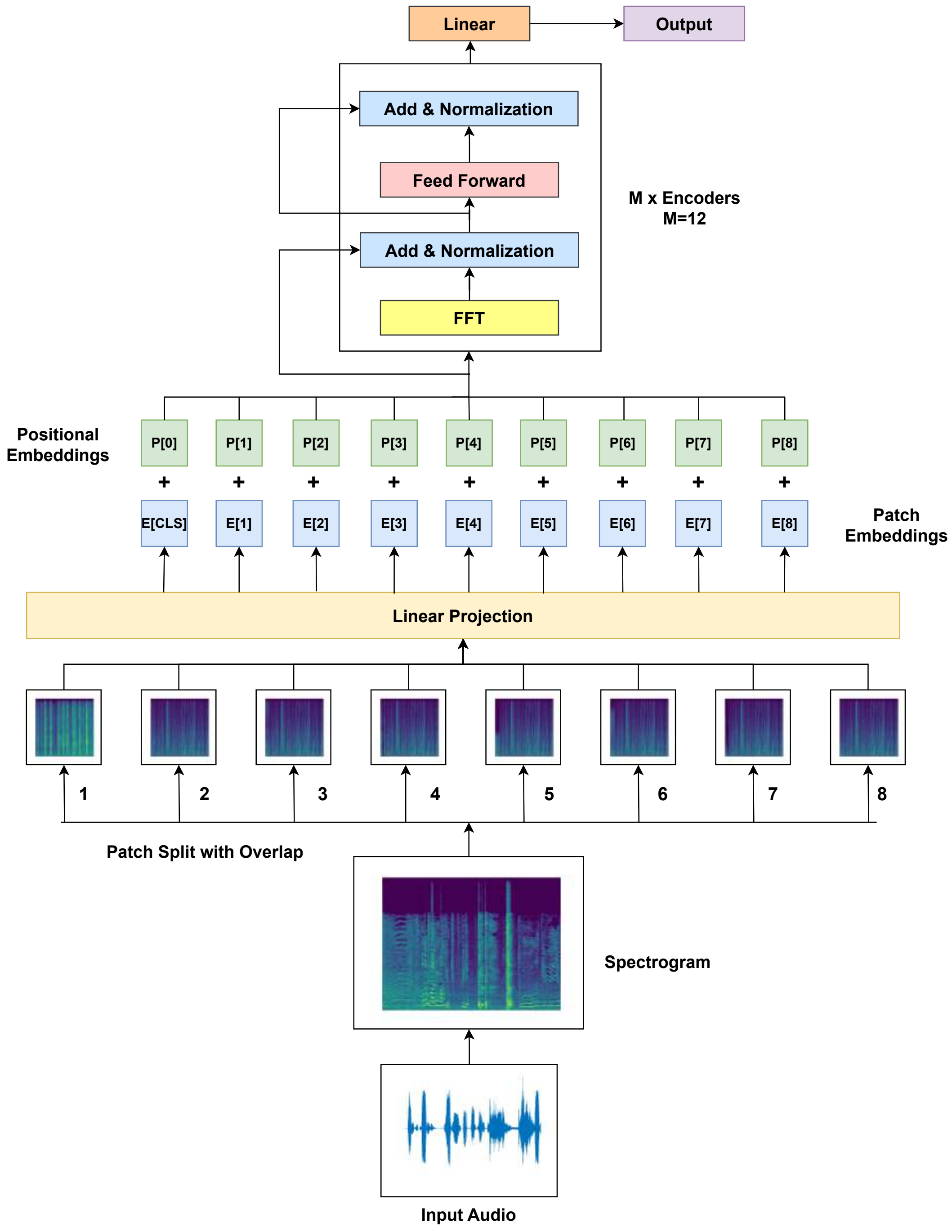

Figure 5.1: Model architecture of ASFNet.

5.1.2 Embeddings

Each 16×16 patch is flattened into a 1D patch embedding of size 768 using a linear projection or patch embedding layer. Additionally, a trainable positional embedding is added to each patch embedding to enable the model to perceive the spatial structure of the audio spectrogram.

Throughout all layers of the Transformer, we use a constant latent vector size D. Therefore, we flatten the patches and apply a trainable linear projection to map them to D dimensions. These transformed patches are known as patch embeddings.

5.1.3 Encoder

As ASFNet is designed specifically for audio classification tasks, it only employs the encoder layers of the transformer architecture. In our specific implementation, the encoder architecture comprises 12 layers with an embedding dimension of 768. The self-attention sublayers in the Transformer are replaced by the Fourier sublayers. The Fourier sublayer applies a 2D FFT to the input embeddings, employing a 1D FFT along the sequence dimension and another 1D FFT along the hidden dimension. This operation by the Fourier sublayer can be defined as,

$$y = \boldsymbol{R}(F_{seq}(F_h(x))) \tag{5.1}$$

where x is the embedding, F_{seq} and F_h are the FFTs along the sequence dimension and the hidden dimension respectively, and ***R*** is the real part of the Fourier Transform. From equation 5.1, it can be seen that only the real part of the total transformation is considered so that the following feed-forward sublayers or output layers do not require dealing with complex numbers.

5.1.4 Output

The audio spectrogram representation as the encoder's output is mapped to classification labels using a linear layer that applies a sigmoid activation function.

5.2 ASFNet Training

MAudioSet training and validation subsets were used to train and validate ASFNet respectively. As mentioned earlier in 4.2, MAudioSet is a multi-label audio dataset with 10 classes and is divided into 5 folds.

We used a similar training pipeline with [10, 16] for our experiments. Data augmentation including mixup [88] with a mixup ratio of 0.5, spectrogram masking [89] with a maximum time mask length of 192 frames, and a maximum frequency mask length of

48 bins was used. Weight averaging [90] and ensemble [91] were used as model aggregation approaches. ASFNet was trained with a batch size of 2. Adam [92] was used as the optimizer and binary cross-entropy was the loss function. We initiated the training process with an initial learning rate of 5e-5 and continued training the model for 25 epochs. Starting from the 3^{rd} epoch, the learning rate was halved every 2 epochs. Mean average precision (mAP) was used as the main evaluation metric. ASFNet was implemented using the PyTorch framework [86]. Google Colaboratory's A100 and V100 GPUs [93] were used as the hardware accelerator for training the model. The summary of the model training is provided in Table 5.1 and the training loss curve is shown in Fig. 5.2.

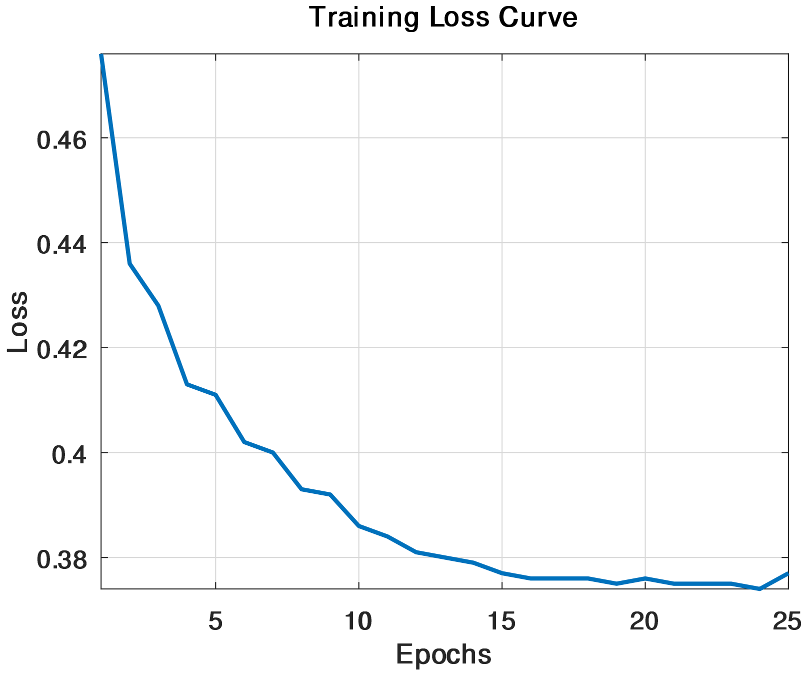

Figure 5.2: ASFNet Training loss curve.

Table 5.1: Summary of the experimental configurations of ASFNet

Accelerator	GPU
Loss function	Binary cross-entropy
Optimizer	Adam
Initial learning rate	5e-5
No. of epochs	25
Batch size	2
Evaluation Metric	mAP

5.3 Performance Evaluation

This section presents a discussion of the outcomes obtained from the proposed ASFNet and the achieved results are then compared to those of previous research studies. The performance of ASFNet was evaluated in terms of both precision and efficiency.

5.3.1 Evaluation Strategy and Metrics

This study involved the implementation and training of three different versions of ASFNet. Likewise, three variations of AST were also implemented to ensure a fair comparison. The distinctions among these versions can be understood by looking at Table 5.2.

Table 5.2: Variants of ASFNet and AST

Variant	Embedding dimension	Attention heads
AST-Tiny	192	3
AST-Small	384	6
AST	768	12
ASFNet-Tiny	192	-
ASFNet-Small	384	-
ASFNet	768	-

All the variants were trained and evaluated on the 5-fold of training and validation subsets of MAudioSets respectively. The metric used for the performance evaluation is Mean Average Precision (mAP). Initially, the mAPs of ASFNet and other baseline methods are computed for each validation subset fold, and subsequently, the overall average is determined. The efficiency of the model is evaluated in terms of the number of model parameters and the model size. Finally, a quantitative summary of results showcasing the relative change ASFNet offers in terms of both performance and efficiency is provided.

5.3.2 Tools and Resources Used for Implementation

All of the models were implemented using the PyTorch framework [86]. PyTorch is a numerical library designed for implementing machine learning algorithms on a large scale. It includes built-in functionality for automatically calculating gradients using the backpropagation algorithm. Additionally, PyTorch offers efficient implementations of operations such as convolutions, matrix multiplications, and other tasks commonly associated with deep neural networks, particularly optimized for NVIDIA CUDA-enabled GPUs [94]. The models were trained on the A100 and V100 GPUs provided by Google Colaboratory [93] with 52 GB RAM.

5.3.3 Classification Performance

Table 5.3 summarizes the ASFNet variants' performances, along with a comparison to existing methods.

Table 5.3: Performance comparison of ASFNet and other methods on the MAudioSet validation subset

	Model architecture	**mAP**					
		Fold-1	Fold-2	Fold-3	Fold-4	Fold-5	Average
PSLA [10]	CNN+Attention	0.3967	0.4080	0.3675	0.3954	0.3755	0.389
AST-Tiny [16]	Pure attention	0.4018	0.4083	0.3709	0.3900	0.3525	0.385
AST-Small [16]	Pure attention	0.4016	0.4232	0.3835	0.3960	0.3841	0.398
AST [16]	Pure attention	0.4140	0.4296	0.3779	0.4085	0.3762	0.401
AST-Ensemble [16]	Pure attention	0.4188	0.4389	0.3832	0.4093	0.3880	0.408
ASFNet-Tiny	Attention free, FFT	0.4334	0.5131	0.4119	0.4438	0.4365	0.448
ASFNet-Small	Attention free, FFT	**0.5160**	0.5195	0.4169	0.4336	0.4324	0.464
ASFNet	Attention free, FFT	0.4557	0.5309	0.4451	**0.4783**	0.4526	0.473
ASFNet-Ensemble	Attention free, FFT	0.4539	**0.5315**	**0.4508**	0.4758	**0.4573**	**0.474**

As mentioned in Section 5.2, weight averaging and ensemble were used to boost the performance of ASFNet. Note that, weight averaging and ensemble do not increase the model size. Using weight averaging, ASFNet achieves an average mAP of 0.473 among the five folds.

For the ensemble approach, ASFNet was trained with three different settings, precisely three different numbers of patches. The ensemble model achieves an average mAP of 0.474. It can clearly be seen from the results that all the variants of ASFNet outperform

AST variants and PSLA.

For additional examination, we also explored the effectiveness of ASFNet when excluding the speech and silence classes. This decision was motivated by two main factors. Firstly, if the speech source separation model performs accurately and successfully separates speech data, the need for ASFNet to detect speech events diminishes. Secondly, as the silence class does not contain any medical information, its detection is not essential for diagnosis. The results are shown in Table 5.4 and ASFNet demonstrates superior performance in this case too.

Table 5.4: Performance comparison of ASFNet and other methods on the MAudioSet validation subset without the speech and silence events

	Model architecture	**mAP**					
		Fold-1	Fold-2	Fold-3	Fold-4	Fold-5	Average
PSLA [10]	CNN+Attention	0.4267	0.4380	0.4075	0.4354	0.4155	0.425
AST-Tiny [16]	Pure attention	0.4318	0.4383	0.4009	0.4200	0.3825	0.415
AST-Small [16]	Pure attention	0.4316	0.4532	0.4135	0.4260	0.4141	0.431
AST [16]	Pure attention	0.4440	0.4596	0.4079	0.4385	0.4062	0.431
AST-Ensemble [16]	Pure attention	0.4488	0.4689	0.4132	0.4393	0.4180	0.437
ASFNet-Tiny	Attention free, FFT	0.4634	0.5331	0.4619	0.4938	0.4765	0.485
ASFNet-Small	Attention free, FFT	**0.5160**	0.5195	0.4169	0.4336	0.4324	0.464
ASFNet	Attention free, FFT	0.4957	0.5309	0.4751	**0.5283**	0.5026	0.506
ASFNet-Ensemble	Attention free, FFT	0.4939	**0.5315**	**0.5008**	0.5258	**0.5073**	**0.511**

5.3.4 Model Efficiency

We also compared ASFNet with other models in terms of model parameters and size and the comparison is presented in Table 5.5. ASFNet demonstrates improved performance on MAudioSet while providing faster inference time.

Table 5.5: Number of model parameters and size of the models

	Model Parameters	**Model Size**
PSLA [10]	4 M	16 MB
AST-Tiny [16]	6 M	23.0 MB
AST-Small [16]	22 M	86.4 MB
AST [16]	87 M	334.7 MB
AST-Ensemble [16]	87 M	334.7 MB
ASFNet-Tiny	3 M	13.6 MB
ASFNet-Small	14 M	54.2 MB
ASFNet	56 M	216.4 MB
ASFNet-Ensemble	56 M	216.4 MB

5.3.5 Overall Performance

Overall, when taking into account both performance and efficiency, ASFNet demonstrates impressive results. Even the lowest average mAP among ASFNet variants (ASFNet-Tiny) surpasses that of existing models. If performance boosting is considered, ensembling ASFNet architectures with different settings is the optimal choice. The divergence in outcomes between AST and ASFNet variants, as well as the impact of the ensemble approach, can be observed in Fig. 5.3.

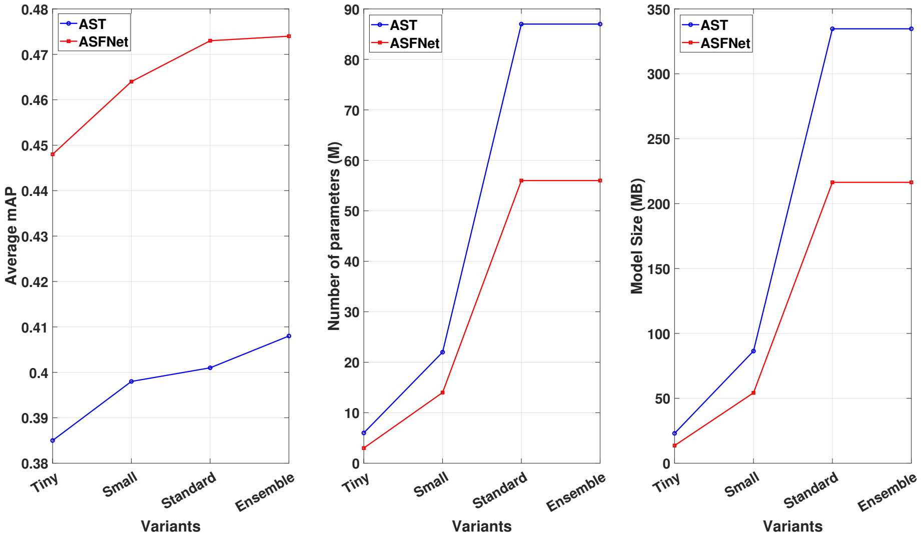

Figure 5.3: Visualization of the performance and efficiency comparison between AST and ASFNet variants.

Moreover, Table 5.6 presents a concise summary of the results for a better understanding, showcasing the quantitative improvements that ASFNet offers over AST.

Table 5.6: Quantitative Improvements of ASFNet

	Relative Average mAP Improvement (%)	**Relative Model Parameters Reduction (%)**	**Relative Model Size Reduction (%)**
ASFNet-Tiny	16.36	50.00	40.87
ASFNet-Small	16.58	36.36	37.26
ASFNet	17.95	35.63	35.34
ASFNet-Ensemble	16.17	35.63	35.34

5.4 Ablation Study

To explore the impact of Fourier sublayers, we designed and trained hybrid models that combine both Attention and Fourier mechanism. Instead of completely replacing all self-attention sublayers with Fourier sublayers, we selectively replaced a few while leaving the remaining sublayers unchanged. The standard AST has a depth of 12. For this experiment, we replaced m numbers of self-attention sublayers with Fourier sublayers, and the rest of the $12 - m$ sublayers were self-attention sublayers.

5.4.1 Effect of Fourier Sublayers on the Classification Performance

Fig. 5.4 shows the effect of varying m, or in other words, Fourier sublayers on the performance of AST.

Except in a few cases, for all the folds, mAP increases with the increment of m. The average mAP of the five folds is always upwards, and clearly, the worst and best performances are with $m = 0$ (purely attention, AST) and $m = 12$ (attention-free, ASFNet) respectively.

5.4.2 Effect of Fourier Sublayers on the Model Effeciency

We also compared the model parameters and sizes in those cases. The results are illustrated in Fig. 5.5.

As Fourier sublayers increase, the model parameters and size decrease in a similar manner to the improvement in performance.

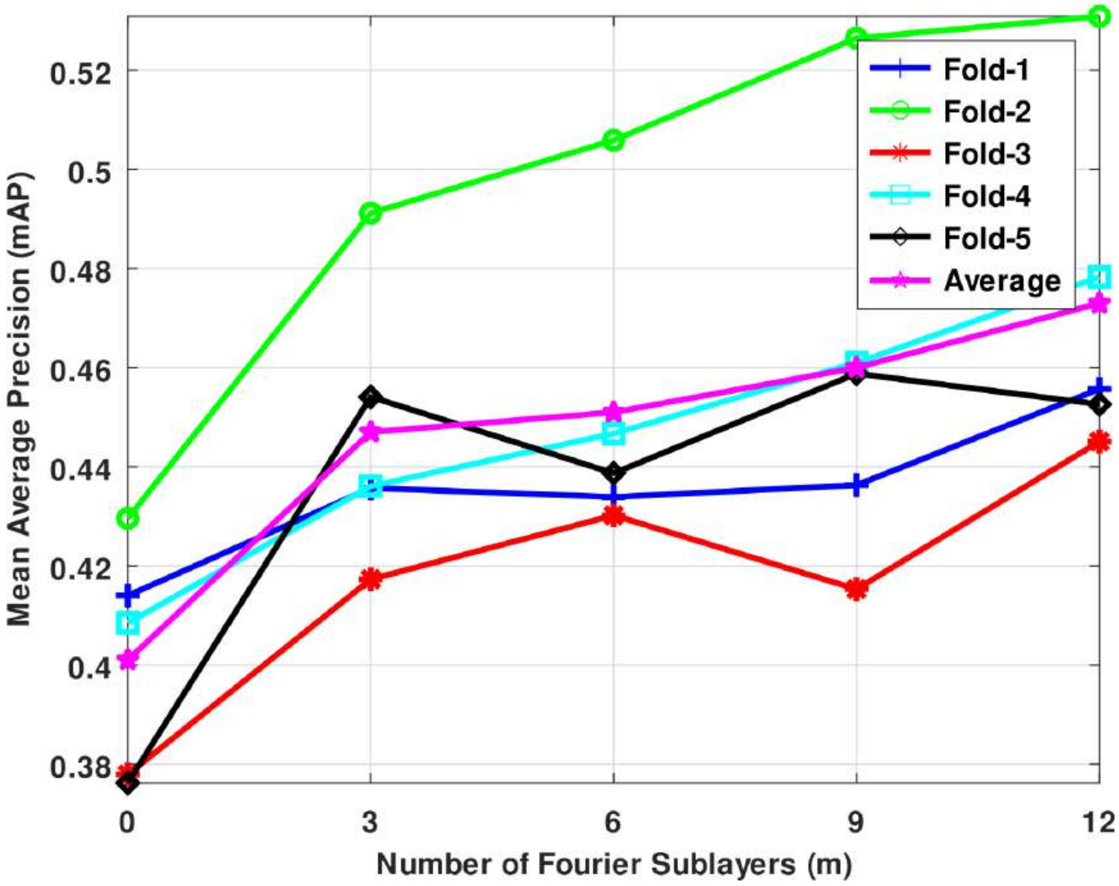

Figure 5.4: Effect of Fourier sublayers on the classification performance.

5.4.3 ASFNet without Positional Embeddings

Due to the positional details captured by the Fourier Transform, ASFNet achieves comparable performance even without using positional embeddings. However, we chose to include positional embeddings to achieve the best performance, facilitate a more direct comparison with AST and ensure a fair evaluation. Table 5.7 presents the performance of ASFNet trained without the positional embeddings. Even without the positional embeddings, ASFNet achieves an average mAP of 0.469 which is 85% of what is achieved with positional embeddings.

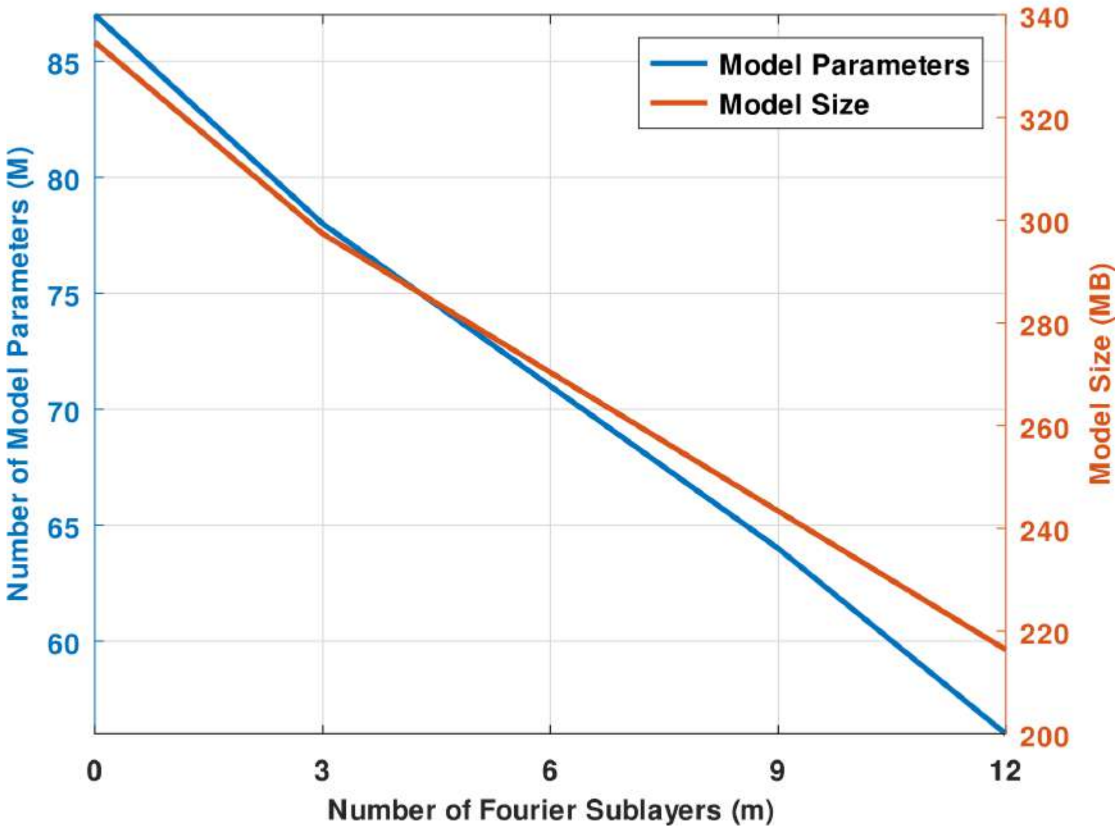

Figure 5.5: Effect of Fourier sublayers on the model efficiency.

Table 5.7: Performance of ASFNet without positional embeddings

Fold	mAP	Model Parameters	Model Size
1	0.4471		
2	0.5351		
3	0.4440		
4	0.4657		
5	0.4518		
Average	0.469	56 M	216.4 MB

5.4.4 Insight

The Fourier Transform provides a highly efficient means of combining embeddings, enabling comprehensive access to all embeddings for the feed-forward sublayers. By leveraging the inherent characteristics of the Fourier Transform, we can interpret each alternate encoder block as utilizing consecutive Fourier and inverse Fourier Transforms. This dynamic conversion of input between the temporal and frequency domains harnesses the multiplication of frequency domain coefficients, which corresponds to convolution in the time domain with a corresponding set of coefficients. Thus, ASFNet can be conceptualized as a process that alternates between multiplications and convolutions.

5.5 Summary

ASFNet adopts the architecture of the AST encoder but replaces the self-attention sublayers with Fourier sublayers. Despite being trained with limited data, ASFNet demonstrates promising results in terms of both performance and efficiency. On average, it outperforms other methods with a 16.17% relative improvement in average mAP while requiring relatively 35.63% fewer model parameters and having a relatively 35.34% smaller model size.

Chapter 6

Conclusions

In this chapter, we will review and summarize the work done so far. Then we will clarify and point out future prospects for improvement and expansion.

6.1 Summary

In this thesis, we utilized the Wave-U-Net architecture as a speech source separation model which is able to preserve the speech-privacy while capturing the audio data by separating the speech sources in the first place. Wave-U-Net achieves an overall SDR of 11.829 indicating a near-perfect source separation.

The main highlight of this research is the introduction of the Audio Spectrogram Fourier Network (ASFNet), an attention-free deep neural network that can be implemented for precise and efficient sound event detection in the hospital or medical environment. ASFNet follows the architecture of the AST encoder but replaces the self-attention sublayers with Fourier sublayers and demonstrates encouraging outcomes in terms of both performance and efficiency in spite of training with a limited amount of data. ASFNet outperforms the other methods showing 16.17% relative improvement in the average mAP and it is able to achieve this performance with fewer model parameters and smaller model size. Even the ASFNet variant with fewer model parameters achieves a better mAP than that of the AST variant with the highest model parameters. Given its lightweight nature, we anticipate that ASFNet will be highly effective when deployed in resource-limited medical settings and edge devices for diverse healthcare applications.

Thus ASFNet, together with Wave-U-Net can be instrumental in speech-privacy aware and low-resource medical sound event detection or monitoring systems.

6.2 Future Prospects

We propose some prospects for improvement and elaboration of the model –

- We showed that ASFNet achieves superior outcomes in terms of both performance and efficiency in spite of training with a limited amount of data. We could collect a large-scale medical audio dataset to train our ASFNet and see how far the performance improves.
- We trained and evaluated ASFNet with 10 classes. We could also investigate if the model performs equally to distinguish among further complex sound events such as different heart sounds: normal, murmur, artifacts, etc.
- We could also train and evaluate ASFNet on the full AudioSet to see how well ASFNet is as a general audio classification model.
- We claimed that the proposed method could be integrated with low-resource medical systems or devices for real-time classification. We could verify that by implementing a software version of the proposed method in such a system.
- We utilized Wave-U-Net as a pre-processing module to separate speech sources and maintain speech-privacy. We could also try to merge Wave-U-Net with ASFNet and develop a better single model that not only separates speech sources but also detects other sound events.

References

[1] Piczak, K.J. “Esc: Dataset for environmental sound classification.” In “Proceedings of the 23rd ACM international conference on Multimedia,” pp. 1015–1018, 2015

[2] Cartwright, M., Cramer, J., Mendez, A.E.M., Wang, Y., Wu, H.H., Lostanlen, V., Fuentes, M., Dove, G., Mydlarz, C., Salamon, J. et al. “Sonyc-ust-v2: An urban sound tagging dataset with spatiotemporal context.” *arXiv preprint arXiv:2009.05188*, 2020

[3] Al Hossain, F., Lover, A.A., Corey, G.A., Reich, N.G. and Rahman, T. “Flusense: a contactless syndromic surveillance platform for influenza-like illness in hospital waiting areas.” *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, volume 4, no. 1:pp. 1–28, 2020

[4] Eyben, F., Weninger, F., Gross, F. and Schuller, B. “Recent developments in opensmile, the munich open-source multimedia feature extractor.” In “Proceedings of the 21st ACM international conference on Multimedia,” pp. 835–838, 2013

[5] Schuller, B., Steidl, S., Batliner, A., Vinciarelli, A., Scherer, K., Ringeval, F., Chetouani, M., Weninger, F., Eyben, F., Marchi, E. et al. “The interspeech 2013 computational paralinguistics challenge: Social signals, conflict, emotion, autism.” In “Proceedings INTERSPEECH 2013, 14th Annual Conference of the International Speech Communication Association, Lyon, France,” , 2013

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

[7] Dieleman, S. and Schrauwen, B. “End-to-end learning for music audio.” In “2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP),” pp. 6964–6968. IEEE, 2014

[8] Trigeorgis, G., Ringeval, F., Brueckner, R., Marchi, E., Nicolaou, M.A., Schuller, B. and Zafeiriou, S. "Adieu features? end-to-end speech emotion recognition using a deep convolutional recurrent network." In "2016 IEEE international conference on acoustics, speech and signal processing (ICASSP)," pp. 5200–5204. IEEE, 2016

[9] Kong, Q., Cao, Y., Iqbal, T., Wang, Y., Wang, W. and Plumbley, M.D. "Panns: Large-scale pretrained audio neural networks for audio pattern recognition." *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, volume 28:pp. 2880–2894, 2020

[10] Gong, Y., Chung, Y.A. and Glass, J. "Psla: Improving audio tagging with pretraining, sampling, labeling, and aggregation." *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, volume 29:pp. 3292–3306, 2021

[11] Rybakov, O., Kononenko, N., Subrahmanya, N., Visontai, M. and Laurenzo, S. "Streaming keyword spotting on mobile devices." *arXiv preprint arXiv:2005.06720*, 2020

[12] Li, P., Song, Y., McLoughlin, I.V., Guo, W. and Dai, L.R. "An attention pooling based representation learning method for speech emotion recognition.", 2018

[13] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S. et al. "An image is worth 16x16 words: Transformers for image recognition at scale." *arXiv preprint arXiv:2010.11929*, 2020

[14] Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A. and Jégou, H. "Training data-efficient image transformers & distillation through attention." In "International conference on machine learning," pp. 10347–10357. PMLR, 2021

[15] Yuan, L., Chen, Y., Wang, T., Yu, W., Shi, Y., Jiang, Z.H., Tay, F.E., Feng, J. and Yan, S. "Tokens-to-token vit: Training vision transformers from scratch on imagenet." In "Proceedings of the IEEE/CVF international conference on computer vision," pp. 558–567, 2021

[16] Gong, Y., Chung, Y.A. and Glass, J. "AST: Audio Spectrogram Transformer." In "Proc. Interspeech 2021," pp. 571–575, 2021

[17] Kitaev, N., Kaiser, Ł. and Levskaya, A. "Reformer: The efficient transformer." *arXiv preprint arXiv:2001.04451*, 2020

[18] Rae, J.W., Potapenko, A., Jayakumar, S.M. and Lillicrap, T.P. "Compressive transformers for long-range sequence modelling." *arXiv preprint arXiv:1911.05507*, 2019

[19] Child, R., Gray, S., Radford, A. and Sutskever, I. "Generating long sequences with sparse transformers." *arXiv preprint arXiv:1904.10509*, 2019

[20] Wang, S., Li, B.Z., Khabsa, M., Fang, H. and Ma, H. "Linformer: Self-attention with linear complexity." *arXiv preprint arXiv:2006.04768*, 2020

[21] Katharopoulos, A., Vyas, A., Pappas, N. and Fleuret, F. "Transformers are rnns: Fast autoregressive transformers with linear attention." In "International Conference on Machine Learning," pp. 5156–5165. PMLR, 2020

[22] Tay, Y., Bahri, D., Metzler, D., Juan, D., Zhao, Z. and Zheng, C. "Synthesizer: Rethinking self-attention in transformer models. arxiv 2020." *arXiv preprint arXiv:2005.00743*, volume 2, 2020

[23] Choromanski, K., Likhosherstov, V., Dohan, D., Song, X., Gane, A., Sarlos, T., Hawkins, P., Davis, J., Mohiuddin, A., Kaiser, L. et al. "Rethinking attention with performers." *arXiv preprint arXiv:2009.14794*, 2020

[24] Zhai, S., Talbott, W., Srivastava, N., Huang, C., Goh, H., Zhang, R. and Susskind, J. "An attention free transformer." *arXiv preprint arXiv:2105.14103*, 2021

[25] El-Bakry, H.M. and Zhao, Q. "Fast object/face detection using neural networks and fast fourier transform." *International Journal of Computer and Information Engineering*, volume 1, no. 11:pp. 3748–3753, 2007

[26] Mathieu, M., Henaff, M. and LeCun, Y. "Fast training of convolutional networks through ffts." *arXiv preprint arXiv:1312.5851*, 2013

[27] Highlander, T. and Rodriguez, A. "Very efficient training of convolutional neural networks using fast fourier transform and overlap-and-add." *arXiv preprint arXiv:1601.06815*, 2016

[28] Pratt, H., Williams, B., Coenen, F. and Zheng, Y. "Fcnn: Fourier convolutional neural networks." In "Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2017, Skopje, Macedonia, September 18–22, 2017, Proceedings, Part I 17," pp. 786–798. Springer, 2017

[29] Lin, S., Liu, N., Nazemi, M., Li, H., Ding, C., Wang, Y. and Pedram, M. “Fft-based deep learning deployment in embedded systems.” In “2018 Design, Automation & Test in Europe Conference & Exhibition (DATE),” pp. 1045–1050. IEEE, 2018

[30] Chitsaz, K., Hajabdollahi, M., Karimi, N., Samavi, S. and Shirani, S. “Acceleration of convolutional neural network using fft-based split convolutions.” *arXiv preprint arXiv:2003.12621*, 2020

[31] Goldberg, K., Shapiro, S., Richardson, E. and Avidan, S. “Rethinking fun: Frequency-domain utilization networks.” *arXiv preprint arXiv:2012.03357*, 2020

[32] Koplon, R. and Sontag, E.D. “Using fourier-neural recurrent networks to fit sequential input/output data.” *Neurocomputing*, volume 15, no. 3-4:pp. 225–248, 1997

[33] Zhang, Y.Q. and Chan, L.W. “Forenet: Fourier recurrent networks for time eries prediction.”, 2000

[34] Zhang, J., Lin, Y., Song, Z. and Dhillon, I. “Learning long term dependencies via fourier recurrent units.” In “International Conference on Machine Learning,” pp. 5815–5823. PMLR, 2018

[35] Choromanski, K., Likhosherstov, V., Dohan, D., Song, X., Gane, A., Sarlos, T., Hawkins, P., Davis, J., Belanger, D., Colwell, L. et al. “Masked language modeling for proteins via linearly scalable long-context transformers.” *arXiv preprint arXiv:2006.03555*, 2020

[36] Tamkin, A., Jurafsky, D. and Goodman, N. “Language through a prism: A spectral approach for multiscale language representations.” *Advances in Neural Information Processing Systems*, volume 33:pp. 5492–5504, 2020

[37] Lee-Thorp, J., Ainslie, J., Eckstein, I. and Ontanon, S. “Fnet: Mixing tokens with fourier transforms.” *arXiv preprint arXiv:2105.03824*, 2021

[38] Stoller, D., Ewert, S. and Dixon, S. “Wave-u-net: A multi-scale neural network for end-to-end audio source separation.” *arXiv preprint arXiv:1806.03185*, 2018

[39] LeCun, Y., Boser, B., Denker, J.S., Henderson, D., Howard, R.E., Hubbard, W. and Jackel, L.D. “Backpropagation applied to handwritten zip code recognition.” *Neural computation*, volume 1, no. 4:pp. 541–551, 1989

[40] Zhou, Y.T. and Chellappa, R. "Computation of optical flow using a neural network." In "ICNN," pp. 71–78, 1988

[41] Rumelhart, D.E., Hinton, G.E. and Williams, R.J. "Learning representations by back-propagating errors." *nature*, volume 323, no. 6088:pp. 533–536, 1986

[42] Robbins, H. and Monro, S. "A stochastic approximation method." *The annals of mathematical statistics*, pp. 400–407, 1951

[43] He, K., Zhang, X., Ren, S. and Sun, J. "Deep residual learning for image recognition." In "Proceedings of the IEEE conference on computer vision and pattern recognition," pp. 770–778, 2016

[44] Huang, G., Liu, Z., Van Der Maaten, L. and Weinberger, K.Q. "Densely connected convolutional networks." In "Proceedings of the IEEE conference on computer vision and pattern recognition," pp. 4700–4708, 2017

[45] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V. and Rabinovich, A. "Going deeper with convolutions." In "Proceedings of the IEEE conference on computer vision and pattern recognition," pp. 1–9, 2015

[46] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł. and Polosukhin, I. "Attention is all you need." *Advances in neural information processing systems*, volume 30, 2017

[47] Cheng, J., Dong, L. and Lapata, M. "Long short-term memory-networks for machine reading." *arXiv preprint arXiv:1601.06733*, 2016

[48] Parikh, A.P., Täckström, O., Das, D. and Uszkoreit, J. "A decomposable attention model for natural language inference." *arXiv preprint arXiv:1606.01933*, 2016

[49] Paulus, R., Xiong, C. and Socher, R. "A deep reinforced model for abstractive summarization." *arXiv preprint arXiv:1705.04304*, 2017

[50] Lin, Z., Feng, M., Santos, C.N.d., Yu, M., Xiang, B., Zhou, B. and Bengio, Y. "A structured self-attentive sentence embedding." *arXiv preprint arXiv:1703.03130*, 2017

[51] Ba, J.L., Kiros, J.R. and Hinton, G.E. "Layer normalization." *arXiv preprint arXiv:1607.06450*, 2016

[52] Wu, Y., Schuster, M., Chen, Z., Le, Q.V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K. et al. “Google’s neural machine translation system: Bridging the gap between human and machine translation.” *arXiv preprint arXiv:1609.08144*, 2016

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

[54] Gehring, J., Auli, M., Grangier, D., Yarats, D. and Dauphin, Y.N. “Convolutional sequence to sequence learning.” In “International conference on machine learning,” pp. 1243–1252. PMLR, 2017

[55] Press, O. and Wolf, L. “Using the output embedding to improve language models.” *arXiv preprint arXiv:1608.05859*, 2016

[56] Weissenborn, D., Täckström, O. and Uszkoreit, J. “Scaling autoregressive video models.” *arXiv preprint arXiv:1906.02634*, 2019

[57] Baevski, A. and Auli, M. “Adaptive input representations for neural language modeling.” *arXiv preprint arXiv:1809.10853*, 2018

[58] van Hengel, P. and Anemüller, J. “Audio event detection for in-home care.” In “Int. Conf. on Acoustics (NAG/DAGA),” , 2009

[59] Manukyan, V., Durieux, B.N., Gramling, C.J., Clarfeld, L.A., Rizzo, D.M., Eppstein, M.J. and Gramling, R. “Automated detection of conversational pauses from audio recordings of serious illness conversations in natural hospital settings.” *Journal of Palliative Medicine*, volume 21, no. 12:pp. 1724–1728, 2018

[60] Matos, S., Birring, S.S., Pavord, I.D. and Evans, H. “Detection of cough signals in continuous audio recordings using hidden markov models.” *IEEE Transactions on Biomedical Engineering*, volume 53, no. 6:pp. 1078–1083, 2006

[61] Takahashi, S.y., Morimoto, T., Maeda, S. and Tsuruta, N. “Cough detection in spoken dialogue system for home health care.” In “Eighth International Conference on Spoken Language Processing,” , 2004

[62] Cai, L. and Wang, Y. “A phase-based active contour model for segmentation of breast ultrasound images.” In “2013 6th International Conference on Biomedical Engineering and Informatics,” pp. 91–95. IEEE, 2013

[63] Amoh, J. and Odame, K. “Deep neural networks for identifying cough sounds.” *IEEE transactions on biomedical circuits and systems*, volume 10, no. 5:pp. 1003–1011, 2016

[64] Larson, E.C., Lee, T., Liu, S., Rosenfeld, M. and Patel, S.N. “Accurate and privacy preserving cough sensing using a low-cost microphone.” In “Proceedings of the 13th international conference on Ubiquitous computing,” pp. 375–384, 2011

[65] Barry, S.J., Dane, A.D., Morice, A.H. and Walmsley, A.D. “The automatic recognition and counting of cough.” *Cough*, volume 2, no. 1:pp. 1–9, 2006

[66] Le, S. and Hu, W. “Cough sound recognition based on hilbert marginal spectrum.” In “2013 6th International Congress on Image and Signal Processing (CISP),” volume 3, pp. 1346–1350. IEEE, 2013

[67] Liu, J.M., You, M., Li, G.Z., Wang, Z., Xu, X., Qiu, Z., Xie, W., An, C. and Chen, S. “Cough signal recognition with gammatone cepstral coefficients.” In “2013 IEEE China Summit and International Conference on Signal and Information Processing,” pp. 160–164. IEEE, 2013

[68] Ho, T.K. “Random decision forests.” In “Proceedings of 3rd international conference on document analysis and recognition,” volume 1, pp. 278–282. IEEE, 1995

[69] Cortes, C. and Vapnik, V. “Support-vector networks.” *Machine learning*, volume 20, no. 3:pp. 273–297, 1995

[70] Monge-Alvarez, J., Hoyos-Barceló, C., Dahal, K. and Casaseca-de-la Higuera, P. “Audio-cough event detection based on moment theory.” *Applied Acoustics*, volume 135:pp. 124–135, 2018

[71] Deshpande, G. and Schuller, B. “An overview on audio, signal, speech, & language processing for covid-19.” *arXiv preprint arXiv:2005.08579*, 2020

[72] Kenton, J.D.M.W.C. and Toutanova, L.K. “Bert: Pre-training of deep bidirectional transformers for language understanding.” In “Proceedings of naacL-HLT,” volume 1, p. 2, 2019

[73] Cybenko, G. “Approximation by superpositions of a sigmoidal function.” *Mathematics of control, signals and systems*, volume 2, no. 4:pp. 303–314, 1989

[74] Barron, A.R. “Universal approximation bounds for superpositions of a sigmoidal function.” *IEEE Transactions on Information theory*, volume 39, no. 3:pp. 930–945, 1993

[75] Minami, K.i., Nakajima, H. and Toyoshima, T. “Real-time discrimination of ventricular tachyarrhythmia with fourier-transform neural network.” *IEEE transactions on Biomedical Engineering*, volume 46, no. 2:pp. 179–185, 1999

[76] Gothwal, H., Kedawat, S., Kumar, R. et al. “Cardiac arrhythmias detection in an ecg beat signal using fast fourier transform and artificial neural network.” *Journal of Biomedical Science and Engineering*, volume 4, no. 04:p. 289, 2011

[77] Mironovova, M. and Bíla, J. “Fast fourier transform for feature extraction and neural network for classification of electrocardiogram signals.” In “2015 Fourth International Conference on Future Generation Communication Technology (FGCT),” pp. 1–6. IEEE, 2015

[78] Zhang, Z., Wang, Y. and Wang, K. “Fault diagnosis and prognosis using wavelet packet decomposition, fourier transform and artificial neural network.” *Journal of Intelligent Manufacturing*, volume 24:pp. 1213–1227, 2013

[79] Li, Z., Kovachki, N., Azizzadenesheli, K., Liu, B., Bhattacharya, K., Stuart, A. and Anandkumar, A. “Fourier neural operator for parametric partial differential equations.” *arXiv preprint arXiv:2010.08895*, 2020

[80] Rao, Y., Zhao, W., Zhu, Z., Lu, J. and Zhou, J. “Global filter networks for image classification.” In “Advances in Neural Information Processing Systems (NeurIPS),” , 2021

[81] Ronneberger, O., Fischer, P. and Brox, T. “U-net: Convolutional networks for biomedical image segmentation.” In N. Navab, J. Hornegger, W.M. Wells, and A.F. Frangi, editors, “Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015,” pp. 234–241. Springer International Publishing, Cham, 2015

[82] Gemmeke, J.F., Ellis, D.P., Freedman, D., Jansen, A., Lawrence, W., Moore, R.C., Plakal, M. and Ritter, M. “Audio set: An ontology and human-labeled dataset for audio events.” In “2017 IEEE international conference on acoustics, speech and signal processing (ICASSP),” pp. 776–780. IEEE, 2017

[83] Garofolo, J., Lamel, L., Fisher, W., Fiscus, J., Pallett, D., Dahlgren, N. and Zue, V. “TIMIT Acoustic-Phonetic Continuous Speech Corpus.”, 1993

[84] Snyder, D., Chen, G. and Povey, D. “MUSAN: A Music, Speech, and Noise Corpus.”, 2015. ArXiv:1510.08484v1

[85] Salamon, J., MacConnell, D., Cartwright, M., Li, P. and Bello, J.P. “Scaper: A library for soundscape synthesis and augmentation.” In “2017 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA),” pp. 344–348. IEEE, 2017

[86] Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L. and Lerer, A. “Automatic differentiation in pytorch.”, 2017

[87] Vincent, E., Gribonval, R. and Févotte, C. “Performance measurement in blind audio source separation.” *IEEE transactions on audio, speech, and language processing*, volume 14, no. 4:pp. 1462–1469, 2006

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

[89] Park, D.S., Chan, W., Zhang, Y., Chiu, C.C., Zoph, B., Cubuk, E.D. and Le, Q.V. “Specaugment: A simple data augmentation method for automatic speech recognition.” *arXiv preprint arXiv:1904.08779*, 2019

[90] Izmailov, P., Podoprikhin, D., Garipov, T., Vetrov, D. and Wilson, A.G. “Averaging weights leads to wider optima and better generalization.” *arXiv preprint arXiv:1803.05407*, 2018

[91] Breiman, L. “Bagging predictors.” *Machine learning*, volume 24:pp. 123–140, 1996

[92] Kingma, D.P. and Ba, J. “Adam: A method for stochastic optimization.” *arXiv preprint arXiv:1412.6980*, 2014

[93] Bisong, E. and Bisong, E. “Google colaboratory.” *Building machine learning and deep learning models on google cloud platform: a comprehensive guide for beginners*, pp. 59–64, 2019

[94] Chetlur, S., Woolley, C., Vandermersch, P., Cohen, J., Tran, J., Catanzaro, B. and Shelhamer, E. “cudnn: Efficient primitives for deep learning.” *arXiv preprint arXiv:1410.0759*, 2014