An Analysis of Audio Classification Techniques Using Deep Learning Architectures

by

Mohammed Safwat Imran
20341046
Afia Fahmida Rahman
17101240
Sifat Tanvir
17101454
Hamim Hassan Kadir
16101031
Junaid Iqbal
17101286

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science

Department of Computer Science and Engineering
Brac University
October 2020

Declaration

It is hereby declared that

1. The thesis submitted is my/our own original work while completing degree at Brac University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. We have acknowledged all main sources of help.

Student's Full Name & Signature:

Mohammed Safwat Imran
20341046

Afia Fahmida Rahman
17101240

Sifat Tanvir
17101454

Hamim Hassan Kadir
16101031

Junaid Iqbal
17101286

Approval

The thesis/project titled "An Analysis of Audio Classification Techniques" submitted by

1. Mohammed Safwat Imran (20341046)
2. Afia Fahmida Rahman (1701240)
3. Sifat Tanvir (17101454)
4. Hamim Hassan Kadir (16101031)
5. Junaid Iqbal (17101286)

Of Summer, 2020 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science on October 5, 2020.

Examining Committee:

Supervisor:
(Member)

Moin Mostakim
Lecturer
Department of Computer Science and Engineering
Brac University

Head of Department:
(Chair)

Mahbubul Alam Majumdar
Professor
Department of Computer Science and Engineering
Brac University

Abstract

Failure to classify audio data with high efficiency causes major setbacks in audio processing, voice recognition and noise cancellation. In order to find the best possible neural network models for audio classification, this paper shows the steps in the experiments done on our newly designed CF Model and CFClean Model in both CNN and RNN, and compares the results with some existing models such as DCNN and Piczak-CNN. To get a clear view on the consistency of the results, three different datasets have been experimented on: UrbanSound8k, FSDKaggle2018 and ESC-50. This paper also sheds light on which dataset performs best in terms of train and test accuracy and loss percentage. Moreover, this paper also dives deep into the reasons behind particular models and datasets performing better than the others. Finally, this paper shows what influence envelope function, normalization, segmentation, regularization techniques and dropout layers have in the overall progress.

Keywords: Audio Classification; Neural Network; Deep Learning; Convolutional Neural Network; Recurrent Neural Network

Acknowledgement

Firstly, we are grateful to the almighty for giving us the strength and willpower to get through the hardships during quarantine and successfully complete our thesis.
Secondly, to our advisor Mr. Moin Mostakim sir for his timely and insightful support. He always managed to help us whenever we needed and guided us to where we are today.
Thirdly, to Dr. Md. Golam Robiul Alam sir for providing us with all the guidelines and instructions regarding thesis.
Fourthly, to Ayesha Abed Library for providing us with adequate thesis help and plagiarism report.
And finally to our parents and loved ones without whom this would have been impossible. We are now on the verge of our graduation because of their constant mental and financial support.

Table of Contents

List of Figures

List of Tables

Nomenclature

The next list describes several symbols & abbreviation that will be later used within the body of the document

α Alpha

CF Classify

CNN Convolutional Neural Network

DCT Discrete Cosine Transform

FFT Fast Fourier Transformation

LSTM Long Short-Term Memory

MFCC MEL Frequency Cepstral Coefficient

ReLU Rectified Linear Units

RNN Recurrent neural network

STFT Short-Time Fourier Transform

Chapter 1

Introduction

1.1 Motivation

Despite the gradual growth in the audio processing industry, there have been numerous reports of leading brands of hardware and software not having met their needs as expected. Nowadays even with the advancement of technology, we still have not been able to develop a proper way of getting rid of the noises of audio platforms completely, and failure to classify audio data with high efficiency is one of the reasons behind it.

In our everyday life, whenever we speak using electronic medium, we hear a lot of background sounds apart from the user's voice from the other end. For example, hearing car horn, children crying, air conditioning or drilling while talking over phone is a common phenomenon. Some of the background sounds can be classified as noises. Disturbance due to noises while communicating remotely can serve as fatal at times because noises can create confusion, misinformation or misinterpretation of information.

In addition, in the music industry, the sound composer, engineers and other people who are associated need to use highly expensive instruments to reduce noise from audio. The electronics companies use expensive materials to make high-accuracy noise cancellation recorders, microphones etc. For this reason, the users need to buy expensive instruments and gadgets. This may be feasible to a professional user but for the new-comers and amateurs or who just joined the industry with less budget cannot afford these benefits.

Classifying audio can be used in medical purposes as well such as detecting Pneumonia using the sounds made by the breathing of the patients. It can also be used in refining audio evidences. For example, extracting important information from audio clips, eliminating noises and confusion, voice recognition and recreating audio samples.

If a reliable and high performing classification model can be achieved which would be able to distinguish each type of sound from an audio clip, it would lead to the solvency of many of our everyday problem. Therefore, in our work, we have tried to compare different neural network models to compare the performance in classifying audio datasets, and the reasoning behind the outperforming or underperforming of the models.

1.2 Aims and Objectives

The following are the aims and objectives of our research:

To compare different audio classification models: In our research we intend to compare multiple Neural Network Models of both Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN) in classifying audio data.

To understand the reasoning behind outperforming or under-performing of the models: After critical observation, we wish to determine the variables that are responsible for the models performance and pinpoint why certain models perform better than the others. We also want to compare the datasets used and determine the quality of these datasets and how difference of the characteristics of each dataset influences the overall performance of the classification models.

To identify and separate sounds from distinct sound sources: The goal we want to achieve is to separate relevant sounds from the raw sound file. We are using CNN and RNN to classify the different sounds into different classes. The individuals and avail the choice to hear either one of their statements.

To identify unwanted environmental audio: While recording sound or communication, a common problem faced is making out what we want to hear amidst the background noise. Unwanted sound makes the audio sounds unpleasant and sometimes inaudible. Often the background noise distracts or disrupts us from listening to the important part of the audio. To overcome this problem, we shall use the classes we received from the trained datasets and only keep the relevant classes of audio.

The rest of the paper is organized as follows:
Chapter 2 includes the various related works done in the similar fields. Chapter 3 explains the methodology of our approaches in classifying audio data. Chapter 4 describes the results. Chapter 5 explains the findings of our thesis, which is the culmination of all our experiments. Finally, chapter 6 concludes the paper with a brief idea of our future work.

Chapter 2

Related Work

Audio classification is used in audio refining, speech recognition and noise cancellation and so on. Many significant contributions have been made in these fields. For instance, traditional Automatic-Speech Recognition (ASR) systems have already been developed which works on the principles of the Gaussian mixture model [2]. However, instead of using Gaussian models, we shall be using the Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN) to classify audio data, which are models for processing datasets based on deep learning. CNN is a mathematical construct that is composed of a few layers [23]. Methods for separating overlapping audio files have also been developed over-time, such as Deep Clustering and Deep Attractor Networks, both of which are neural networks [25]. We have decided to use the three typical phases of CNN: convolution, pooling and fully connecting [23] to perform our tasks. We also used RNN which consists of two layers followed by four ReLU units. Even using Convolutional Deep Belief Networks, certain approaches have proved themselves to make unsupervised classification of huge datasets of hand-tailored audio data possible [3]. The existence of such research suggests that whichever goal we are trying to pursue is feasible and possible, hence our main objective is to use CNN and RNN in such a manner that our system learns from the data provided which we know we can attain. These works also give us a view of the extent our future works can be.

Research has also been conducted to distinguish between environmental sounds by [16], using Gaussian Mixture Model, Deep Neural Network, Recurrent Neural Network, Convolutional Neural Network and i-vector. It was found that DNN was the best performing model while an RNN and CNN fusion provided the best average in accuracy albeit, slower. Another comparative analysis was done on using different Machine learning algorithms to classify different audio signals in [6] which described how this procedure can be achieved by treating it as a pattern recognition problem in two stages, extraction and classification. The authors in [6] have also described several purposes of these techniques such speech recognition, automatic bandwidth allocation, which allows for a telephone network to decide how much bandwidth should be allocated for transmission, that is, music requiring higher while no bandwidth for no background noise, allowing for improved efficiency and also audio database indexing, which is currently done manually and is a very tedious task. [6] have used two k-Nearest Neighbor and Support Vector Machine algorithms, concluding that SVM is more accurate while taking less time. Although primarily

used in visual recognition contexts, convolutional architectures have also been applied in speech and music analysis [8], even though is it takes a while to train the program,it shows that convolutional neural networks can be effectively applied in environmental sound classification tasks even with limited datasets and simple data augmentation. A recent venture called Audio AI by [28] aims to isolate vocals from an audio source such as a song by using Short-Time Fourier Transform (STFT) to expose the structure of human speech and then using Convolutional Neural Network (CNN)'s to identify and distinguish between voiced and unvoiced sections in a song, estimate the frequency of the fundamental overtime and apply a mask to capture the harmonic content. In his algorithm, he has treated audio signals as images to expose spatial patterns, as two-dimensional frequency against time graphs. It is observed that using CNN's Audio AI successfully separates the vocals from the instrumental section from the audio source.

A number of related works also worked on the datasets that we focused on. In [29], Urban8k Dataset was used to prove that an important piece for resolving (music) audio problems are the architectures alone using deep neural networks. ELM (Extreme Learning Machine) model has been used and an accuracy of 89.82% was achieved using temporal models. In [27], convolutional neural network and tensor deep stacking network were used to classify environmental sounds of the ESC50 dataset based on the generated spectrograms of these sounds. In [14], the authors tried to prove with the ESC50 dataset that deep convolutional neural networks, which are designed specifically for object recognition in images, can be successfully trained to classify spectral images of environmental sounds. Using Convolutional Recurrent Neural Network (CRNN) 60.0% accuracy was achieved with Spectrogram, MFCC, and CRP combined and 60.3% with just Spectrogram.

In [19], the authors classified audio data with DCNN model using UrbanSound8k and ESC-50 datasets. Authors in [8] also worked with the UrbanSound8k and ESC-50 datasets and classified audio using Piczak-CNN model. In [8] and [19], accuracies of 72.70% and 81.90% were attained using the UrbanSound8k dataset respectively and 64.90% and 68.10% were attained using the ESC-50 dataset, which we would be aiming to surpass in our work.

Chapter 3

Methodology

This chapter provides a detailed description of the theories and processes that have been used by us to classify audio data.

3.1 Audio Pre-Processing

Before starting the the experiments, the datasets were pre-processed. The pre-processing was done using normalization, fast fourier transformation, short-time fourier transformation, MEL filterbank and MELfrequency cepstral coefficient.

3.1.1 Signal Normalization

Audio files have different shapes of waves in a file. We collect data from some kind of sensors and mostly they are microphones. Microphones usually have a bit depth of 16 which can generate 2 to 16 integers in a time domain to generate waves. Normalizing the data is a part of prepossessing. To normalize the signal we apply pre-emphasis filter. There are several reasons behind normalizing a signal. In a signal higher frequencies have less magnitude than the lower frequencies. Also to balance the audio noise ratio and numerical complexities in further calculation we use pre-emphasis filters on the signal data. In a signal x we apply the following equation:

$$y(t) = x(t) - \alpha x(t-1) \tag{3.1}$$

Here α is the filter coefficient having a value of 0.95 or 0.97 [11].

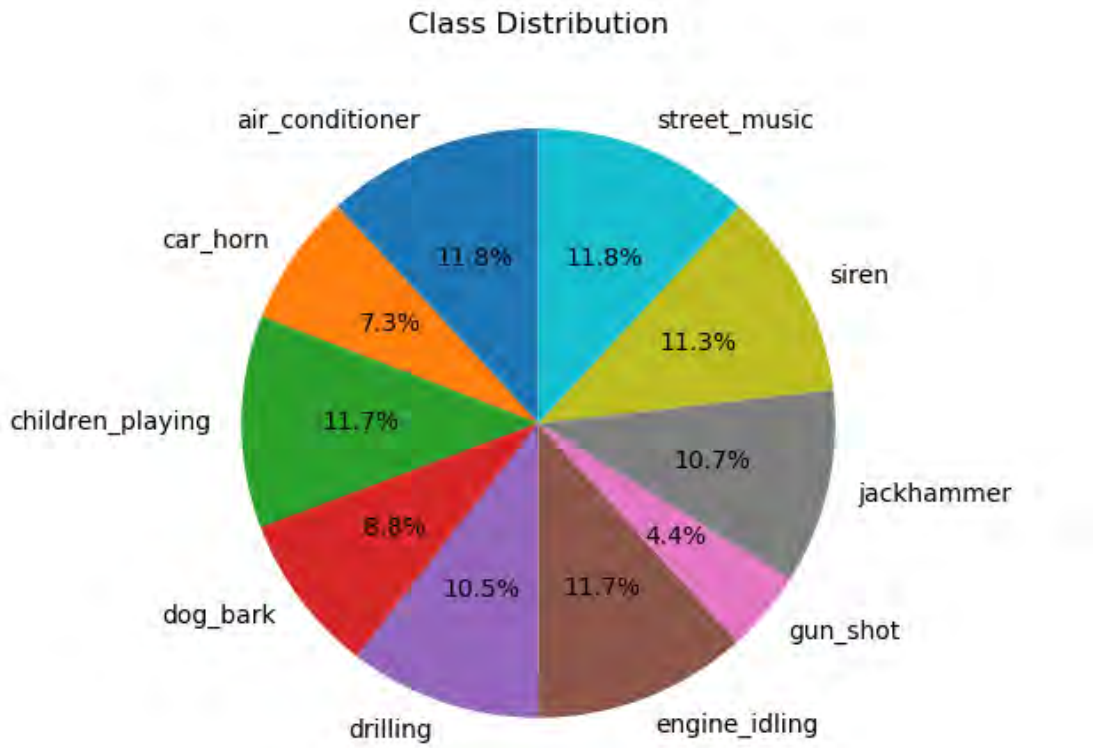

Figure 3.1: *Pie Chart of The Audio Classes Defined in UrbanSound8k Dataset*

3.1.2 Fast Fourier Transformation (FFT)

Waves signals of the audios are difficult to read and recognize for further work. To represent the data in different ways we use fourier transformation which generates a Frequency-Magnitude graph called Periodogram. In this case, specifically we use Fast Fourier Transformation to generate Periodogram. Periodogram is easier to understand the audio signals. A periodogram plots the maximum frequency of 22kHz over Magnitude. The reason behind 22kHz is, our microphone records audios from the environment highest at 44.1kHz. To represent this it generates nyquist frequency which is half of the sample frequency. So We actually can get 22kHz frequency from our environment by using our microphone. Any frequency above 22kHz our microphone will not read. Using fast fourier transformation we take the real values of FFT to generate a periodogram. The goal of the process is to make a spectrogram. A spectrogram is a stack of periodograms adjacent to each other over time. Which generates an image based on the magnitude of frequency to give an idea of visual representation of an audio file. For spectrograms we generate 4 spectrograms for each 1/10th of a second of the audio file. This contains enough information to generate Spectrogram. Before that we also can down sample our periodogram if our audio files do not contain higher frequency waves in it. Usually in our environment we ignore the higher frequencies. For this reason, it will help us to generate better spectrograms to process our data. Figure 3.2 and Figure 3.3 show the fourier transformation of the classes.

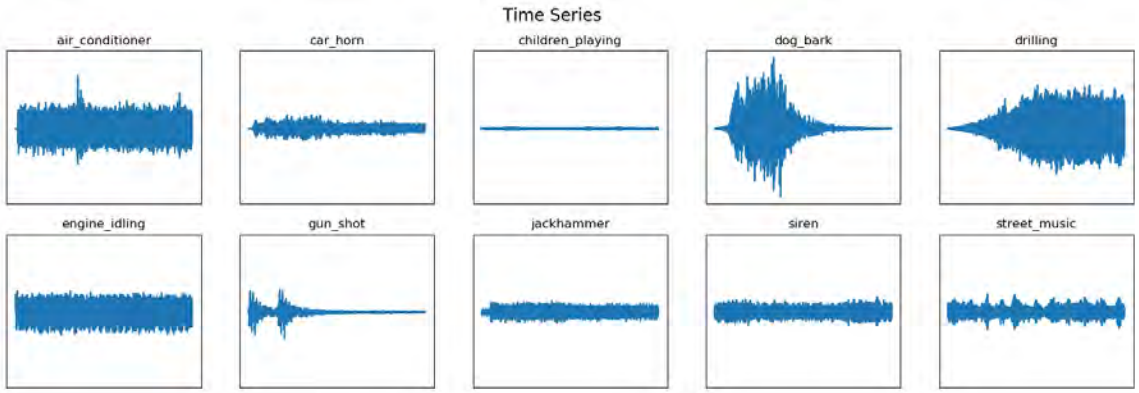

Figure 3.2: *Time Series Plot including frequencies above Nyquist Frequency*

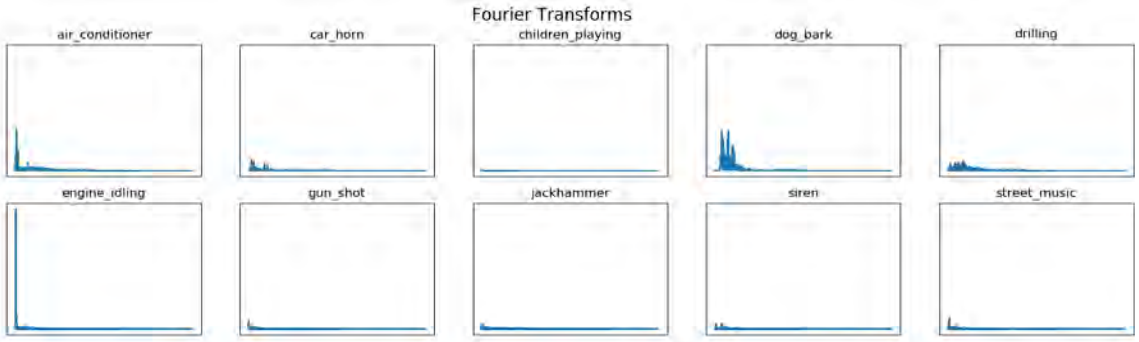

Figure 3.3: *Fourier Transforms of the classes*

3.1.3 Short-Time Fourier Transformation (STFT)

Short-Time Fourier Transform is not just about stacking up periodograms to generate a Spectrogram. Instead of stacking up periodograms it overlaps them. The reason behind the method is that, frequency of an audio signal changes over time. It is a continuous process. So if we take a part of an audio and assume it is stationary, it will help us. In Short-Time Fourier Transform firstly we split the sample signal in small frames and step forward the frame window. First of all we take a part of the sample audio and we take 1 second of it. Now we split the audio in frames. Each frame has a length of 20-40 ms. For our experiment we used 25 ms. Now we step this frame 10 ms forward. Now the new frame has 15 ms common with the previous frame. Here the overlap happens. This process generates 400 samples. The idea behind this is we need a frame small enough but having maximum information so we take the size of 25 ms. The step size 10 ms made 15ms common with the adjacent frames. Which is nearly 60% of it. This helps generate nice contours to assume that we have static audio data. After slicing the data into frames we apply the window function. Overlaps of the frames generate bell curves. Which is called a hamming window. Hamming window is a window function that is used in FFT.

This helps tapering the wave shape of the frames where 2 overlaps. This helps to avoid spectral leakage and counteract the assumption of FFT which is that the data we have is infinite. The equation of a hamming window is:

$$w[n] = 0.54 - 0.46Cos(\frac{2\pi n}{N}) \tag{3.2}$$

where $0 \leq n \leq N$ and N = the window length [4].
This gives us 400 points of FFT but for FFT we need a value based on power of 2. Typically 512 is used. So after 400 it adds 112 zeros after it. After this it generates Periodograms for further filtration process.

3.1.4 MEL Filter Bank

Mel filter bank concept is using triangular filters in the power spectrum. The triangular filter is dependent on the MEL Scale. Usually there are 25-40 filters in the Mel filter bank for use. Mel Filter bank equation is a log equation that generates a filter system that acts like how a human being identifies noise. Humans can easily differentiate between low frequency sounds. One can identify which sound is 70Hz and which one is 170Hz. When the frequency gets higher, human cochlea, which is an organ inside the ear that reacts on different frequencies, can not differentiate. For example 17070Hz and 17170Hz human cochlea can not differentiate. In Mel filter bank works on this principle. It distinguishes low frequencies in short range but expands its range when the frequency is higher. We in this experiment used 26 Mel filters. But depending on purposes it increased till 40 usually. The equations [33] we used for 26 filers are following:

$$M(f) = 1125 * \ln \frac{1+f}{700} \tag{3.3}$$

$$M^{-1}(m) = 700 * ((\exp \frac{m}{1125}) - 1) \tag{3.4}$$

This equation allows us to generate an image of a matrix of size 26x100. For 26 Mel filters we get 26 Values vertically and for 1 second of data it generates 100 values horizontally stacking up each other. In this stage the generated images shown in Figure 3.4 are still hard to recognize which image represents what sound.

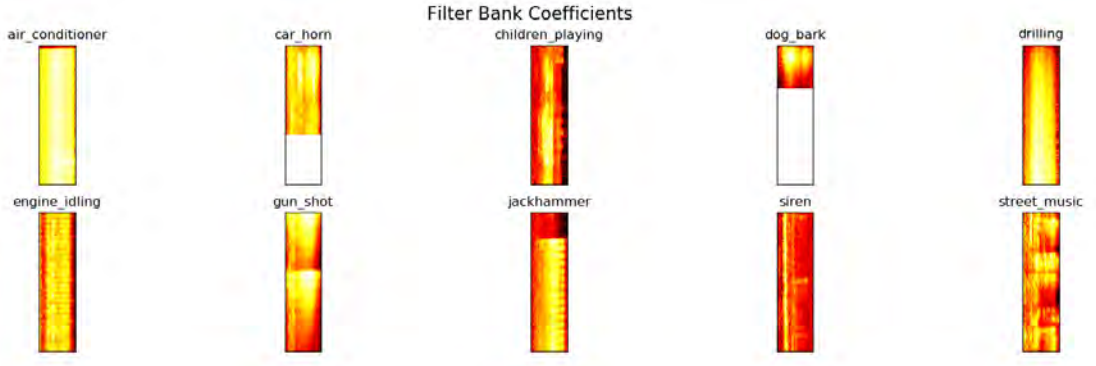

Figure 3.4: *Filter Bank Coefficients of the classes*

3.1.5 MEL Frequency Cepstral Coefficient (MFCC)

This is the final stage of the preprocessing of the signal data. MFCC is a more compact version of the image that was generated using Mel filterbank. MFCC uses Discrete Cosine Transform to Decorelate a lot of energies from the previous energy band. Discrete Cosine Transform (DCT) is a technique which is highly used for image and audio compression. In Fourier transform it works with both sine and cosine functions for compression but DCT is more effective because it works on only the cosine functions. This made it highly effective on compression techniques. In DCT compression it discards specific bits which contain high coefficients to compress data. This is called applying a low pass analog filter. This reduces high frequency data and smoothen the edges. This is the main principle of the DCT to compress audio and image data. In our case MFCC compresses the energy bank image from 26x100 matrix to 13x100 matrix. Which allows to compress the higher frequencies to lower frequencies. This allows the different sounds having identical images. By the end of MFCC data pre-processing phase completes. Figure 3.5 shows the images generated after this stage.

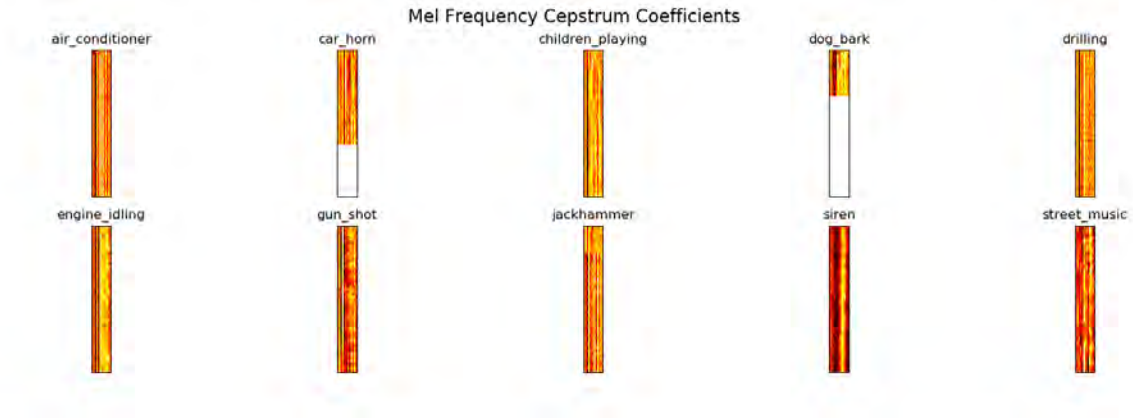

Figure 3.5: *Mel Frequency Cepstrum Coefficients*

3.2 Classification Models

The MFCC function discussed in the previous section is what yields the spectrograms illustrated in figure 3.5. This allows the sequential, audio data to be depicted as image data to our deep learning algorithms. The algorithms used are as follows:

3.2.1 Convolutional Neural Network (CNN)

Convolutional Neural Network is a neural network with multiple layers stacked up which includes pooling layers, convolutional layers, etc. [24]. It has been considered to be an effective mechanism for the classification of image data, eg. spectrograms instead of sequential data, such as audio [8], [10], [13], [18]. In our studies, the raw, sequential audio data have been converted to sequential spectrograms, which are still treated as images, to help our CNN reach its optimum effectiveness.

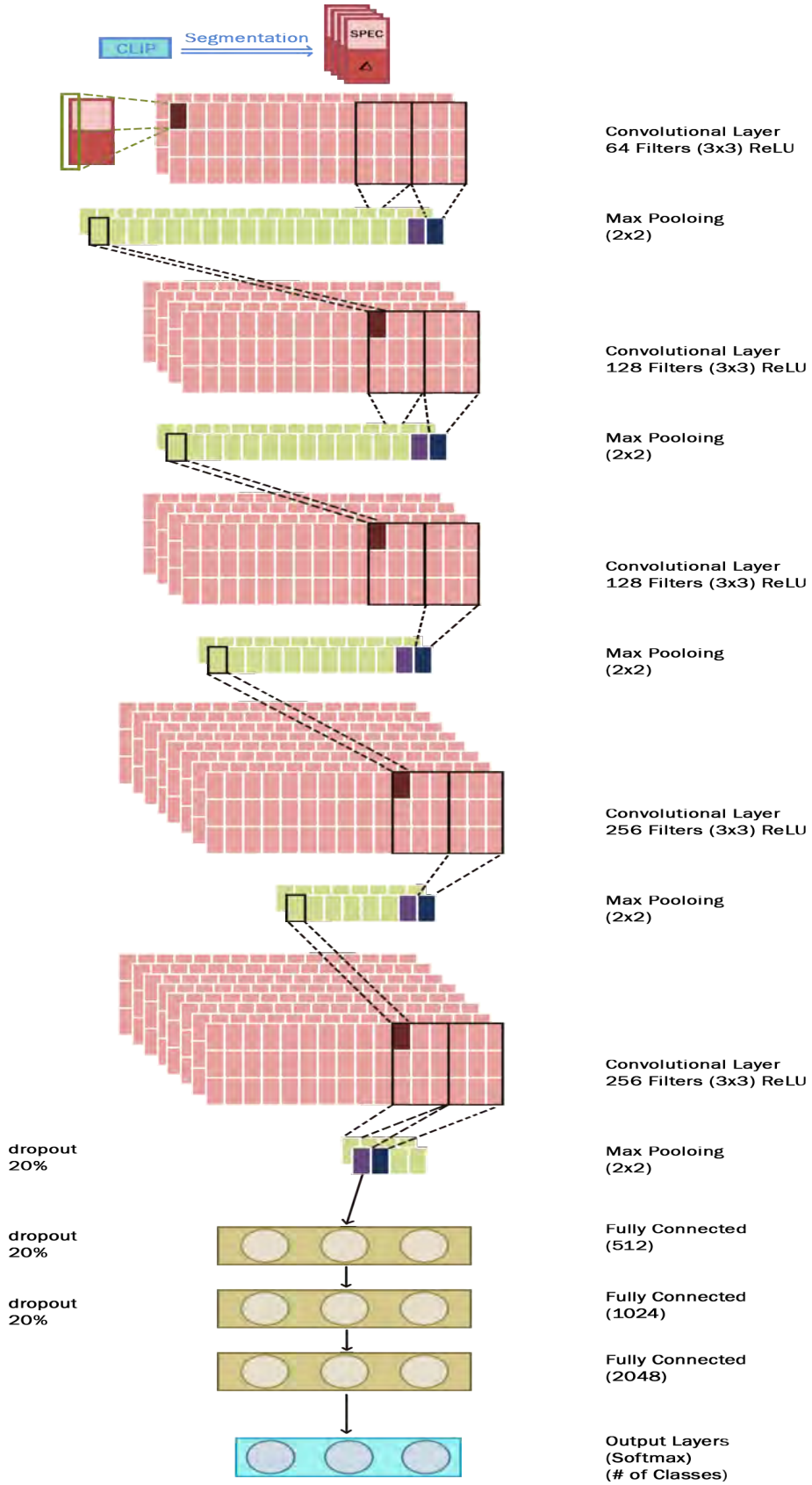

Figure 3.6: *Flowchart of data within the CNN*

As shown in Figure 3.6 total of five 3 x 3 convolutional layers, each followed by

a 2 x 2 pooling layer were used to learn the features. Dropout technique with a probability of 0.2 has been used to avoid overfitting and its risks. ReLU (Rectified Linear Units) were used to model non-linear outputs from the layers [8]. We have used the Adam [15] function to optimize the learning rate by passing a coefficient of 0.0001. Categorical cross-entropy along with softmax components [17] have been to lower down the losses associated with CNN.

3.2.2 Recurrent Neural Network (RNN)

Recurrent Neural Networks work on the principle of hidden states where Long Short-Term Memory (LSTM) units [1] help store information as required by the layers. Since we want our system to compute consecutive units and their data in the most robust form possible as they are all sequential and dependent on each other, we are using the dropout operator with a coefficient of 0.5. Our system made use of two RNN layers followed by four ReLU units [12] to help avoid overfitting, which are shown in Figure 3.7. Like our CNN system, this system too has made use of the components categorical cross-entropy alongside softmax to help reduce losses while training the datasets.

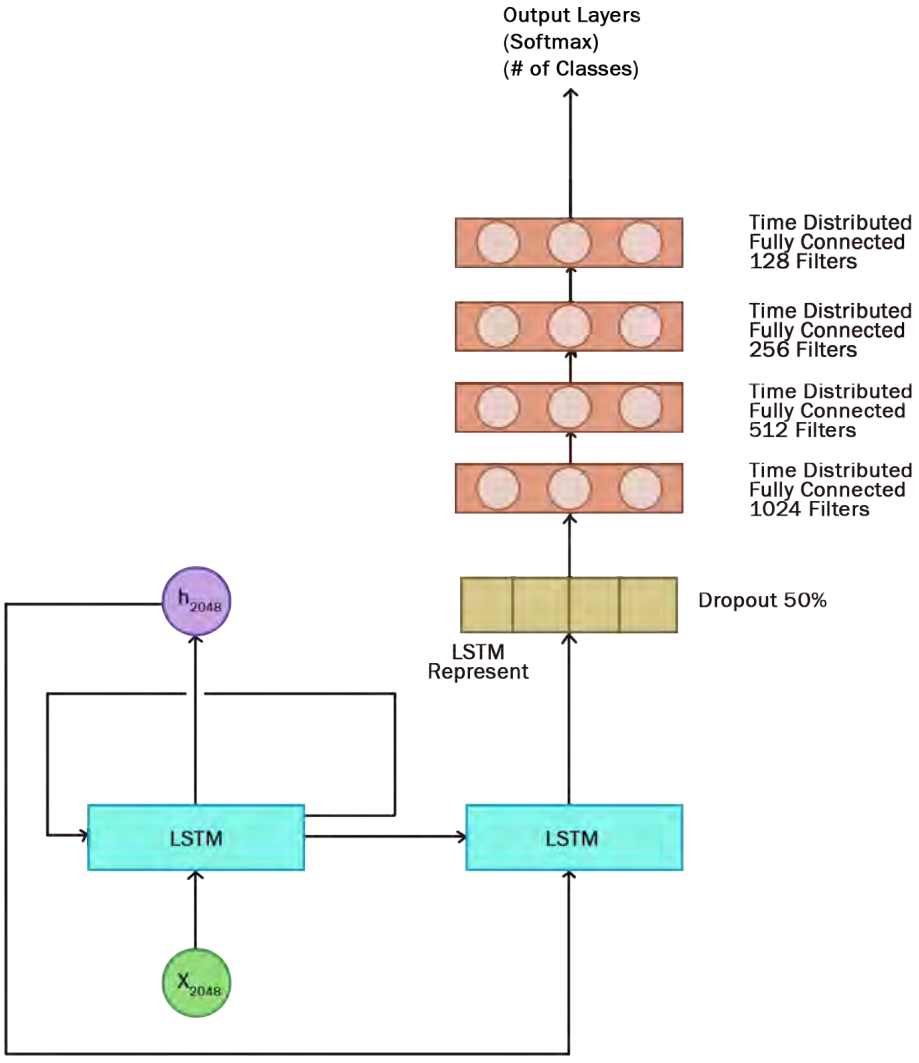

Figure 3.7: *Flowchart of data within the RNN*

Chapter 4

Experimental Analysis

4.1 Datasets

Three publicly available datasets were used, UrbanSound8K [5], ESC-50 [9] and FSDKaggle2018 [22] for model training and testing. UrbanSound8K contains 8732 sound excerpts of around 4 seconds each divided into 10 categories: air conditioner, car horn, children playing, dog bark, drilling, engine idling, gun shot, jackhammer, siren, and street music. ESC-50 contains 2000 recordings of around 5 seconds divided into 50 different classes of sounds divided into 5 categories such as animal sounds, natural sounds, human-speech sounds, interior/domestic sounds and urban noises. The FSDKaggle2018 train dataset contains 9473 samples of sounds divided into 41 classes, with audio samples of varying lengths between 300ms to 30s given that these were manually recorded and depended on user preferences of how long they should be recorded. All audio samples in the aforementioned datasets were recorded using 44.1 KHz sampling rate. Samples were chosen at random from the distribution during each cycle of training for both models.

4.2 Model approaches

Conventionally both mel-spectrograms and mel frequency cepstrum coefficients (mfcc) are used for feature extraction. MFCC is based on what human ears can detect and accounts for the fact that small changes in high frequency sounds are not perceivable, [7]. Two different approaches which will be further referred to 'CF' and 'CFClean' were taken, to analyze which produced better results, while keeping the CNN/RNN model architecture and the learning rate (0.0001) the same while differing in batch size of epoch's due to the different number of samples that were produced for each approach. In both models, the 'Adam' optimizer was used, which has been found to converge faster [15]. Both models were trained for 100 epochs. The loss was calculated using categorical cross entropy which calculates loss by taking the probability of all outputs which have been wrongly classified, otherwise known as softmax loss [21]. The CNN model architecture was a modified version of that described in previous research conducted by Piczak [8]. The models were trained and tested using the Keras library with a Tensorflow backend on an Nvidia 1080Ti GPU with 11GB memory.

4.2.1 CF Model

The 'CF' (acronym for classify) model was having 44.1khz as the sampling rate of the audio clips, taking each audio clip as one sample input for training, not accounting for class imbalance of the distribution. No other processing was done other than using the librosa library from python to extract mfcc features from the audio. The data was divided into a 75:25 ratio for the train-test split, a lower ratio in the testing data would have been insufficient given the lack of samples and would not be a good evaluation of the performance of the model.

4.2.2 CFClean Model

The audio clips were down sampled to 16Khz as most changes in sounds in the aforementioned datasets occur in lower frequency domains and a lower sample rate would mean fewer datapoints hence compacting data and allowing better training times. A signal envelope function was used in order to detect which parts of the audio signal are relevant for the algorithm, which is called noise floor detection. This was achieved by using a rolling window function in python and calculating the mean of the signal of each audio clip and only keeping the parts of the signal which were greater than the mean. Since audio data is a changing signal over time, the number of samples were calculated by taking twice the sum of the length of all the audio and dividing them into 0.1s segments compared to the 'CF' model which took the features from the entire clip allowing fewer number of samples. This allowed for a larger number of samples and hence the data was split into train and test samples in the ratio 9:1. These ideas were inspired by the work of Adams [32]. The input samples were normalized before by taking the minimum and the maximum between all matrices and then dividing them by the difference between maximum and minimum. Previous research has determined that normalization of data allows better performance in classification purposes, [30].

Table 4.1 shows the number of samples, learning rate, normalization status and optimizer name for the datasets used in both CF model and CFClean model.

<table>
<tr><th rowspan="2">Model</th><th rowspan="2">Dataset</th><th colspan="2">Number of samples</th><th rowspan="2">Learning Rate</th><th rowspan="2">Normalization</th><th rowspan="2">Optimizer</th></tr>
<tr><th>Train</th><th>Test</th></tr>
<tr><td rowspan="3">CF</td><td>UrbanSound8k</td><td>6549</td><td>2183</td><td rowspan="6">0.0001</td><td rowspan="3">No</td><td rowspan="6">Adam</td></tr>
<tr><td>FSDKaggle2018</td><td>7104</td><td>2369</td></tr>
<tr><td>ESC-50</td><td>1500</td><td>500</td></tr>
<tr><td rowspan="3">CFClean</td><td>UrbanSound8k</td><td>155227</td><td>17248</td><td rowspan="3">Yes</td></tr>
<tr><td>FSDKaggle2018</td><td>1008384</td><td>112043</td></tr>
<tr><td>ESC-50</td><td>155207</td><td>17246</td></tr>
</table>

Table 4.1: Model designs

4.3 Results

It has been observed that in both cases the CNN model outperformed their RNN counterparts and this is not surprising since CNN's have been leading in classification tasks in computer vision [26]. It is also more difficult to optimize RNN due to vanishing and exploding gradient problems and also longer training times, [31]. In every dataset, the CFClean approach has output better accuracies. Even though some results indicate very high testing loss values, this is the nature of categorical cross entropy, [21]. Given that the output values are probabilities of each labelled class and the model predicts the output of a sample depending on the highest probability and this determines the accuracy value. However, all other classes may not have a probability of 0 and hence this adds to the loss value, meaning that the loss function and accuracy are not directly negatively correlated and the model is penalized highly for any probability values of wrongly predicted labels.

4.3.1 Tabulated Results

Below, in table 4.2, we have presented our results compared to other publications of accuracy results on these datasets.

Model	**Dataset**	**Architecture**	**Accuracy (%)**		**Loss (%)**	
			Train	**Test**	**Train**	**Test**
CF	UrbanSound8k	CNN	97.57	85.89	7.82	85.89
		RNN	98.66	83.69	4.54	104.27
	FSDKaggle2018	CNN	93.26	54.88	22.37	339.34
		RNN	86.11	41.54	46.68	460.59
	ESC-50	CNN	88.87	45.60	41.17	314.31
		RNN	98.20	32.40	6.48	722.21
CFClean	UrbanSound8k	CNN	98.38	94.52	4.42	22.25
		RNN	97.00	92.53	10	143.10
	FSDKaggle2018	CNN	94.26	87.62	23.00	54.81
		RNN	79.05	64.21	72.75	180.07
	ESC-50	CNN	96.89	87.88	9.58	66.06
		RNN	85.13	81.09	48.17	72.22
Piczak-CNN [8]	UrbanSound8k	CNN	-	72.70	-	-
	ESC-50	CNN	-	64.90	-	-
DCNN [19]	UrbanSound8k	CNN	-	81.90	-	-
	ESC-50	CNN	-	68.10	-	-

Table 4.2: Accuracy and loss values for our proposed models

From table 4.2, it is visible that the greatest success achieved (94.52% test accuracy and 22.25% test loss) is through the CFClean model in CNN architecture with the UrbanSound8k dataset. This is a better result produced than by Piczak-CNN model in [8] and DCNN model in [19].

4.3.2 Graphical Outputs

Graphical outputs of the accuracy and loss of CF Model and CFClean Model on UrbanSound8k, ESC-50 and FSDKaggle2018 datasets in both CNN and RNN are shown below.

UrbanSound8k CNN

Figure 4.1 reflects the accuracy and loss with respect to epoch of CFModel and CFClean Model in CNN using UrbanSound8k dataset.

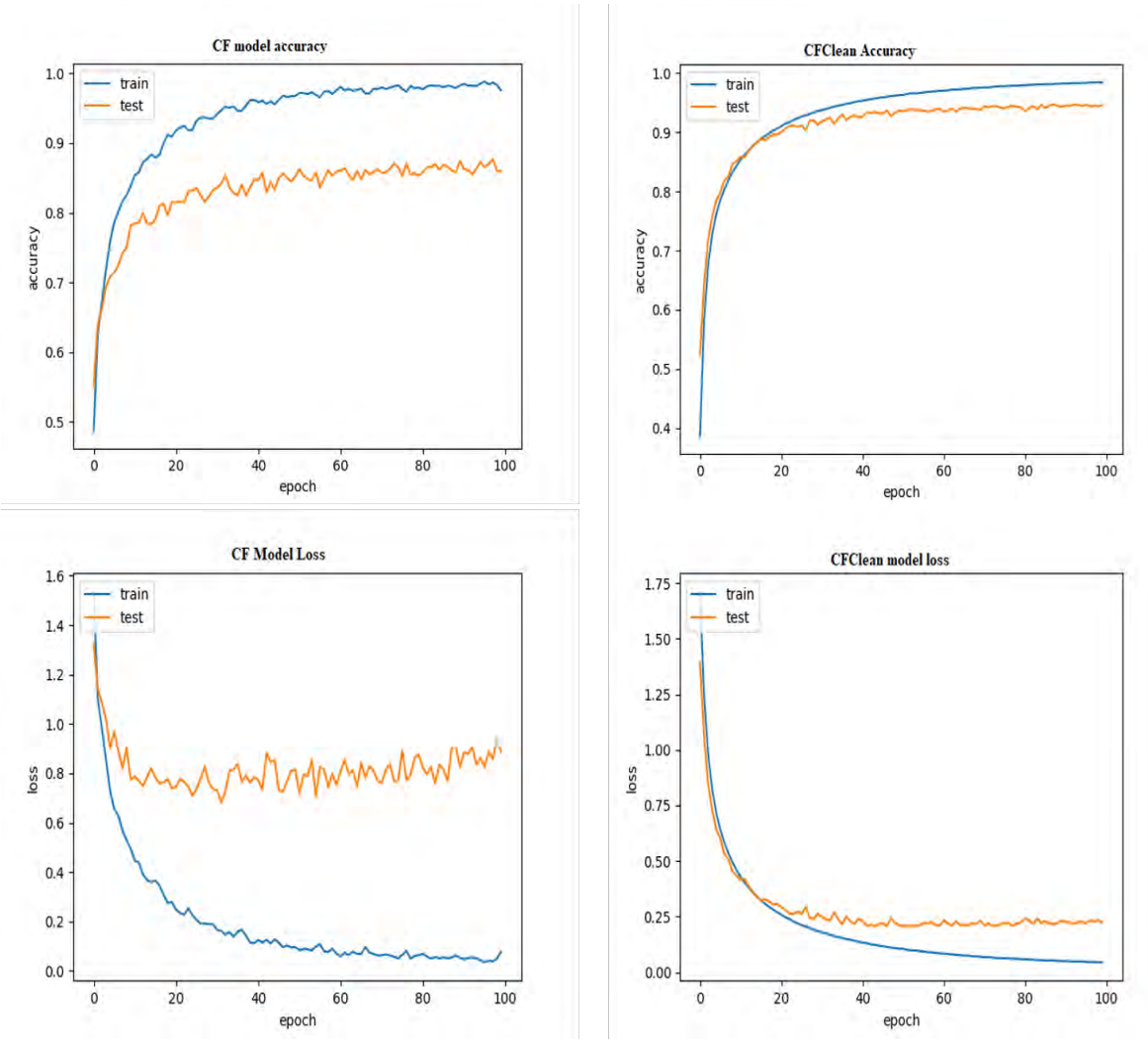

Figure 4.1: *Graphical Representation of Accuracy and Loss with respect to epoch of CF Model and CFClean Model in CNN using UrbanSound8k dataset*

UrbanSound8k RNN

Figure 4.2 reflects the accuracy and loss with respect to epoch of CFModel and CFClean Model in RNN using UrbanSound8k dataset.

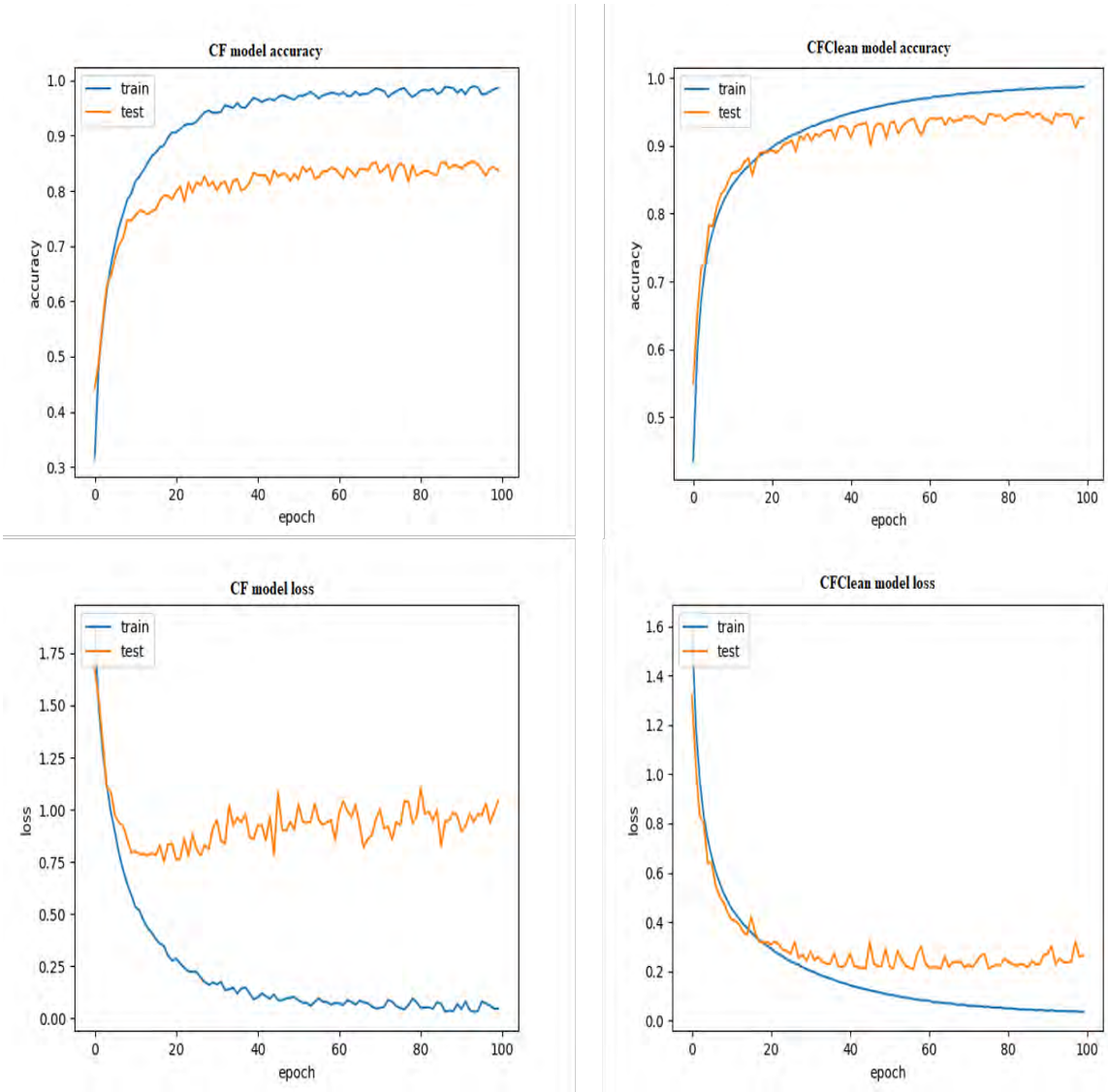

Figure 4.2: *Graphical Representation of Accuracy and Loss with respect to epoch of CF Model and CFClean Model in RNN using UrbanSound8k dataset*

ESC-50 CNN

Figure 4.3 reflects the accuracy and loss with respect to epoch of CFModel and CFClean Model in CNN using ESC-50 dataset.

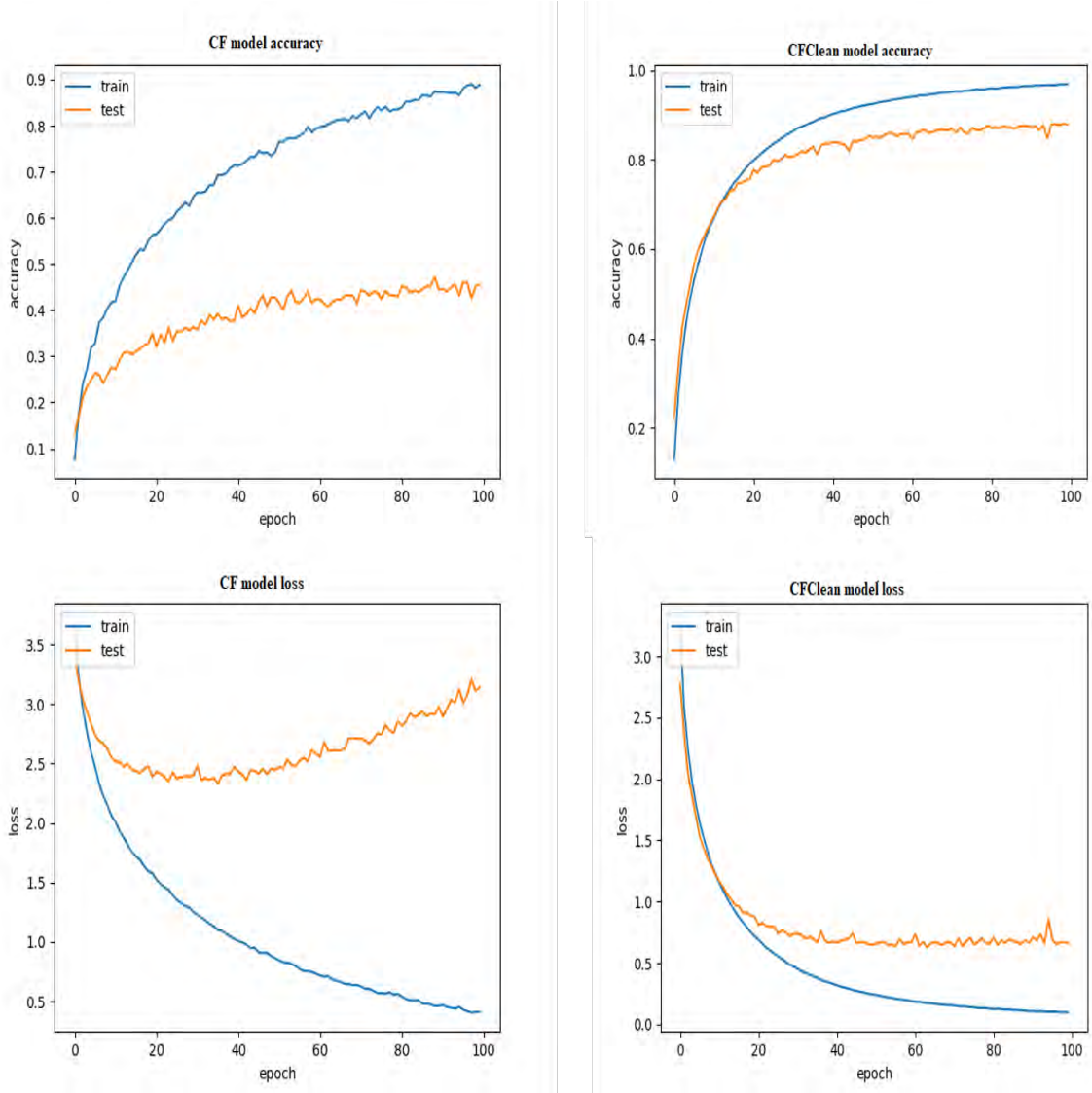

Figure 4.3: *Graphical Representation of Accuracy and Loss with respect to epoch of CF Model and CFClean Model in CNN using ESC-50 dataset*

ESC-50 RNN

Figure 4.4 reflects the accuracy and loss with respect to epoch of CFModel and CFClean Model in RNN using ESC-50 dataset.

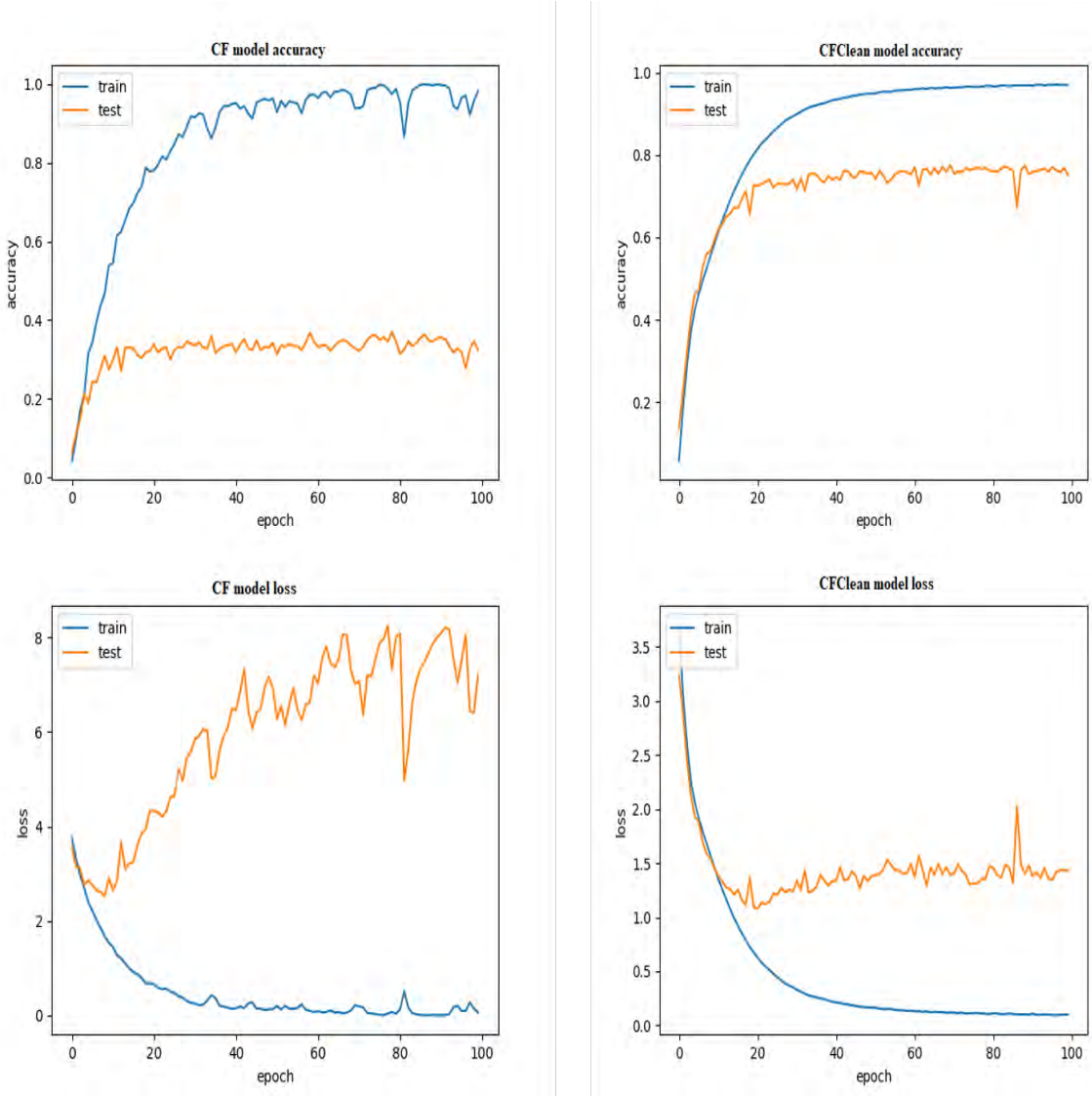

Figure 4.4: *Graphical Representation of Accuracy and Loss with respect to epoch of CF Model and CFClean Model in RNN using ESC-50 dataset*

FSDKaggle2018 CNN

Figure 4.5 reflects the accuracy and loss with respect to epoch of CFModel and CFClean Model in CNN using FSDKaggle2018 dataset.

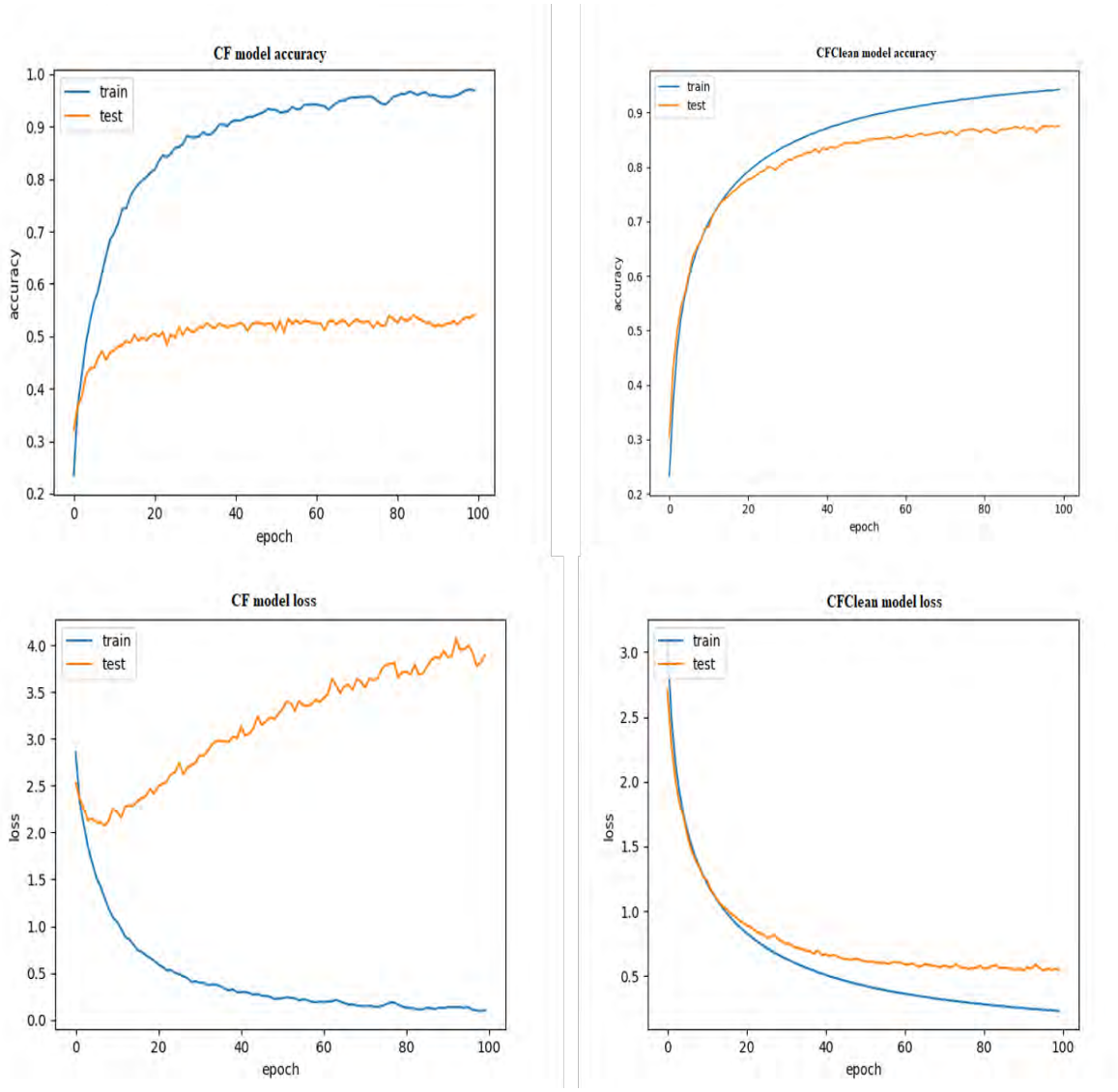

Figure 4.5: *Graphical Representation of Accuracy and Loss with respect to epoch of CF Model and CFClean Model in CNN using FSDKaggle2018 dataset*

FSDKaggle2018 RNN

Figure 4.6 reflects the accuracy and loss with respect to epoch of CFModel and CFClean Model in RNN using FSDKaggle2018 dataset.

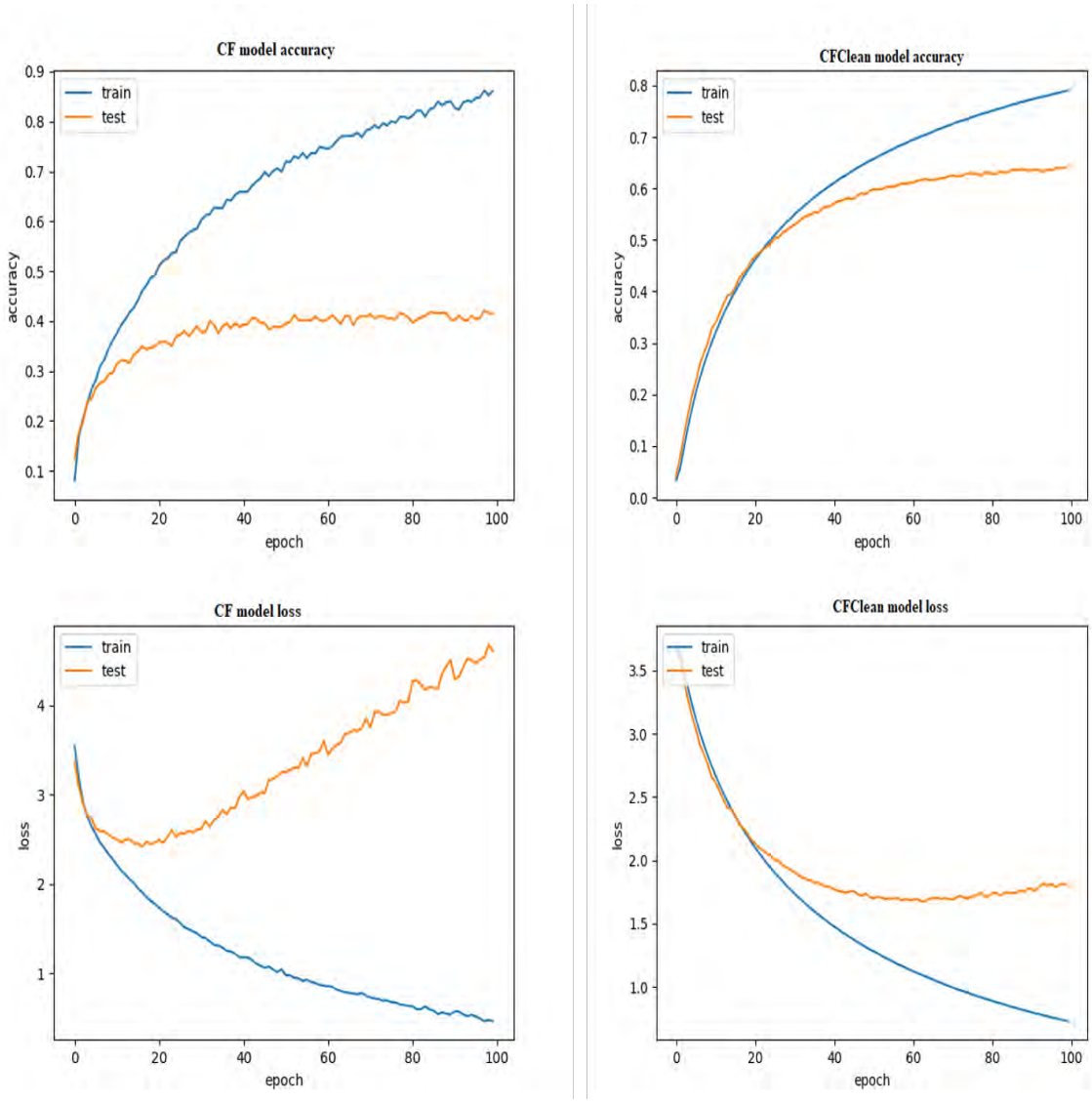

Figure 4.6: *Graphical Representation of Accuracy and Loss with respect to epoch of CF Model and CFClean Model in RNN using FSDKaggle2018 dataset*

Chapter 5

Findings

In this chapter we shall point out our findings and explain the reasons behind those.

5.1 CNN's have outperformed RNN's in audio classification

The differences in the accuracy of predictions are significant. In the UrbanSound8k dataset, the CFClean model has outperformed the CF model by approximately 9%, in both its CNN and RNN networks. In the FSDKaggle2018 dataset, the CNN network of the CFClean model outperformed the CF model by approximately 33% and 23% in its RNN counterpart. The highest difference has been found in the ESC-50 dataset where the CFClean model outperformed the CF model by approximately 42% and 49% in their CNN and RNN networks respectively. Even so, our RNN network in the CFClean model has outperformed previously published work using CNN architectures as can be seen in table 4.2 and we have concluded that this is because of our preprocessing methods on the audio signal. All results have shown that CNN's outperform RNN's in classification tasks and this may be due to the fact that RNN's are better at tasks which rely on the context of previously found values and follow a certain sequence, through the use of feedback loops [26]. We can see in table 4.2 that even though both models perform relatively well on the UrbanSound8k dataset, the RNN models in all instances perform worse, the massive difference between train and test accuracies indicate that even though it might be doing well on the train set, it fails to predict on new data implying overfitting.

5.2 CFClean model has outperformed CF model

The CFClean model performed better than the CF model in every dataset. To have a better understanding, table 5.1 shows a more in depth version of table 4.2, focusing just on the accuracies of CF model and CFClean model in different architectures and datasets.

Datasets	Architecture	Train Accuracy (%)		Test Accuracy (%)	
		CF Model	CFClean Model	CF Model	CFClean Model
UrbanSound8k	CNN	97.57	98.38	85.89	94.52
	RNN	98.66	97.00	83.69	92.53
FSDKaggle2018	CNN	93.26	94.26	54.88	87.62
	RNN	86.11	79.05	41.54	64.21
ESC-50	CNN	88.87	96.89	45.60	87.88
	RNN	98.20	85.13	32.40	81.09

Table 5.1: Comparison between the performances of CF Model and CFClean Model

From table 5.1, we can see that terms of test accuracy CFClean model has always outperformed CF model. CFClean model classified audio with the highest accuracy of 94.52% and the lowest of 64.21% whereas CF model had the highest accuracy of 85.89% and the lowest of 32.40%. One interesting thing to notice from the table is that CF model beat CFClean model in RNN architecture in all cases in terms of train accuracy. However, because of having a heavy disparity between the train and test accuracies of CF model, and the test accuracy being the deciding factor in calculating the overall efficiency, we have come to the conclusion that CFClean model has been more efficient.

5.3 Envelope function, normalization and segmentation have enhanced overall progress

Normalization of the matrices input into the model have also allowed for better generalization of the values. Without normalization, abnormally high or low values would prevent the model from learning the relevance of other features by, for example, giving larger weights to extremely high values. As mentioned earlier, even though high categorical cross entropy does not directly imply that the model is performing poorly, the difference in train and test accuracies indicate that some amount of over fitting has occurred. The removal of dead space in the audio using the envelope function, segmentation of audio clips and normalization of the input matrix have helped the classification process tremendously.

As mentioned in the previous section, CFClean model has been more efficient than CF model which has been largely due to the division of the audio clips into 0.1s segments which increased the sample size and allowed for more training data available for the model. This has helped the model to be able to learn the features necessary to make more accurate predictions. Clips of 1s and 2s segments were also experimented with but they have produced worse results and producing clips of smaller lengths than 0.1s was not feasible since that would dramatically increase the training time and given the pandemic and lack of resources at hand to train the model with, this was not possible.

5.4 Classification on UrbanSound8k dataset has been more effective than on FSDKaggle2018 and ESC-50 datasets

From table 5.1, it is evident that both models have performed well on the UrbanSound8k dataset and it has been realized this is because the dataset is more robust than the others. All sound clips are less than 4 seconds in length, whereas the FSDKaggle2018 dataset has sound clips which are between 300ms to 30s in length, which not only varies the distribution of the dataset but also means that the model learns some features of some of the labels better than the others, given that the probability distribution of the samples is unbalanced. This also means that longer recordings may have background or external noise different from the labelled sound. The UrbanSound8k dataset has only 10 classes while having 8732 clips of audio meaning that each class has around 800 samples, while the ESC-50 has 2000 audio clips in 50 classes meaning that each class has around 40 samples, the FSDKaggle2018 dataset has 9473 clips divided into 41 classes, each class having around 230 samples. Given the scarcity of training data available per class in both ESC-50 and FSD2018Kaggle datasets, this might also be an indication about why both models were not able to learn features as well as they did on the UrbanSound8k dataset. The FSD2018Kaggle dataset also has clips divided into two segments which are manually verified or not, most of which are not, both categories have been used in the training procedure. Given that they have not been verified whether they are clean recordings have been labelled correctly, these factors have also impacted model performance. However, the CFClean model has proved to be better even under such circumstances, given how the audio signal has been preprocessed.

5.5 Regularization techniques and adding more dropout layers have not been helpful

Even with regularization techniques or adding more dropout only reduced the performance even further by lowering test accuracies, hence it has been determined that using this model, this is the best performance that can be achieved. This means that even though the model is performing well on the training set, it fails to generalize in some instances and hence does not do well on unforeseen audio data. According to convention, increasing the data size, reducing the number of layers and adding regularization and dropout could have improved results. Unfortunately, given new training data was not available, adding regularization or more dropout layers or using a simpler, shallower neural network did not help either.

Chapter 6

Conclusion and Future Work

Our paper started off as a way to remove noise from audio clips but first we needed to find an effective method to classify noise in order to remove the part of the audio signal which was undesirable. We have observed ways of cleaning and preprocessing audio signals which would produce impressive results even with shallow neural networks and have also suggested why such processing might be necessary in order to produce better accuracies and lower loss results. Transfer learning methods and data augmentation methods could have been employed in order to produce better results but we have not attempted such techniques, again due to lack of knowledge and resources.

In the future we hope to find ways to classify multiple classes of noise/sounds in a single clip of audio along with attempting to remove such parts of the signal which is undesirable with the use of multitask learning [20]. Separating layers of audio already exist using CNN architectures [28]. However, we want to attempt our own techniques in achieving this task and eventually find more efficient methods in order to isolate noise/speech/instrument sounds or whatever is desired from other background or external noises are present.

Bibliography

[1] S. Hochreiter and J. Schmidhuber, "Long short-term memory", *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997. DOI: 10.1162/neco.1997.9.8.1735. eprint: https://doi.org/10.1162/neco.1997.9.8.1735. [Online]. Available: https://doi.org/10.1162/neco.1997.9.8.1735.

[2] J. Bilmes, "Gaussian models in automatic speech recognition", in. Jan. 2009, pp. 521–555. DOI: 10.1007/978-0-387-30441-0_29.

[3] H. Lee, P. Pham, Y. Largman, and A. Ng, "Unsupervised feature learning for audio classification using convolutional deep belief networks", Jan. 2009, pp. 1096–1104.

[4] A. Bhatnagar, R. Sharma, and R. Kumar, "Analysis of hamming window using advance peak windowing method", Jul. 2012.

[5] J. Salamon, C. Jacoby, and J. Bello, "A dataset and taxonomy for urban sound research", Nov. 2014. DOI: 10.1145/2647868.2655045.

[6] P. Mahana and G. Singh, "Comparative analysis of machine learning algorithms for audio signals classification", *International Journal of Computer Science and Network Security (IJCSNS)*, vol. 15, no. 6, p. 49, 2015.

[7] A. H. Mansour, G. Z. A. Salh, and K. A. Mohammed, "Article: Voice recognition using dynamic time warping and mel-frequency cepstral coefficients algorithms", *International Journal of Computer Applications*, vol. 116, no. 2, pp. 34–41, Apr. 2015, Full text available.

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

[9] ——, *ESC: Dataset for Environmental Sound Classification*, version V2, 2015. DOI: 10.7910/DVN/YDEPUT. [Online]. Available: https://doi.org/10.7910/DVN/YDEPUT.

[10] H. Zhang, I. McLoughlin, and Y. Song, "Robust sound event recognition using convolutional neural networks", in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 559–563.

[11] H. Fayek, *Speech processing for machine learning: Filter banks, mel-frequency cepstral coefficients (mfccs) and what's in-between*, Apr. 2016. [Online]. Available: https://haythamfayek.com/2016/04/21/speech-processing-for-machine-learning.html.

[12] H. B. Sailor and H. A. Patil, "Novel unsupervised auditory filterbank learning using convolutional rbm for speech recognition", *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 12, pp. 2341–2353, 2016.

[13] J. Salamon and J. P. Bello, "Deep convolutional neural networks and data augmentation for environmental sound classification", *CoRR*, vol. abs/1608.04363, 2016. arXiv: 1608.04363. [Online]. Available: http://arxiv.org/abs/1608.04363.

[14] V. Boddapati, A. Petef, J. Rasmusson, and L. Lundberg, "Classifying environmental sounds using image recognition networks", *Procedia Computer Science*, vol. 112, pp. 2048–2056, 2017, Knowledge-Based and Intelligent Information Engineering Systems: Proceedings of the 21st International Conference, KES-20176-8 September 2017, Marseille, France, ISSN: 1877-0509. DOI: https://doi.org/10.1016/j.procs.2017.08.250. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1877050917316599.

[15] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization", *CoRR*, vol. abs/1412.6980v9, 2017. arXiv: 1412.6980v9. [Online]. Available: https://arxiv.org/abs/1412.6980v9.

[16] J. Li, W. Dai, F. Metze, S. Qu, and S. Das, "A comparison of deep learning methods for environmental sound", *CoRR*, vol. abs/1703.06902, 2017. arXiv: 1703.06902. [Online]. Available: http://arxiv.org/abs/1703.06902.

[17] W. Liu, Y. Wen, Z. Yu, and M. Yang, "Large-margin softmax loss for convolutional neural networks", *CoRR*, vol. abs/1612.02295v4, 2017. arXiv: 1612.02295v4. [Online]. Available: https://arxiv.org/abs/1612.02295v4.

[18] L. Wyse, "Audio spectrogram representations for processing with convolutional neural networks", *CoRR*, vol. abs/1706.09559, 2017. arXiv: 1706.09559. [Online]. Available: http://arxiv.org/abs/1706.09559.

[19] X. Zhang, Y. Zou, and W. Shi, "Dilated convolution neural network with leakyrelu for environmental sound classification", Aug. 2017, pp. 1–5. DOI: 10.1109/ICDSP.2017.8096153.

[20] Y. Zhang and Q. Yang, "An overview of multi-task learning", *National Science Review*, vol. 5, no. 1, pp. 30–43, Sep. 2017.

[21] R. G. Bruballa, May 2018. [Online]. Available: https://gombru.github.io/2018/05/23/cross_entropy_loss/.

[22] E. Fonseca, M. Plakal, F. Font, D. P. W. Ellis, X. Favory, J. Pons, and X. Serra, *General-purpose tagging of freesound audio with audioset labels: Task description, dataset, and baseline*, 2018. arXiv: 1807.09902 [cs.SD].

[23] R. Yamashita, M. Nishio, R. Do, and K. Togashi, "Convolutional neural networks: An overview and application in radiology", *Insights into Imaging*, vol. 9, pp. 611–629, 2018.

[24] Z. Zhang, S. Xu, S. Cao, and S. Zhang, "Deep convolutional neural network with mixup for environmental sound classification", *CoRR*, vol. abs/1808.08405, 2018. arXiv: 1808.08405. [Online]. Available: http://arxiv.org/abs/1808.08405.

[25] P. Appeltans, J. Zegers, and H. Van hamme, *Practical applicability of deep neural networks for overlapping speaker separation*, Dec. 2019.

[26] E. Corporation and E. Marketing, *Rnn vs cnn for deep learning: Let's learn the difference*, Aug. 2019. [Online]. Available: https://blog.exxactcorp.com/lets-learn-the-difference-between-a-deep-learning-cnn-and-rnn/.

[27] A. Khamparia, D. Gupta, N. G. Nguyen, A. Khanna, B. Pandey, and P. Tiwari, "Sound classification using convolutional neural network and tensor deep stacking network", *IEEE Access*, vol. 7, pp. 7717–7727, 2019.

[28] A. Koretzky, *Audio ai: Isolating vocals from stereo music using convolutional neural networks*, https://towardsdatascience.com/audio-ai-isolating-vocals-from-stereo-music-using-convolutional-neural-networks-210532383785, Feb. 2019.

[29] J. Pons and X. Serra, "Randomly weighted cnns for (music) audio classification", in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 336–340.

[30] D. Singh and B. Singh, "Investigating the impact of data normalization on classification performance", *Applied Soft Computing*, p. 105 524, 2019, ISSN: 1568-4946. DOI: https://doi.org/10.1016/j.asoc.2019.105524. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1568494619302947.

[31] G. So, *Should we abandon lstm for cnn?*, Mar. 2019. [Online]. Available: https://medium.com/ai-ml-at-symantec/should-we-abandon-lstm-for-cnn-83accaeb93d6.

[32] S. Adams, *Audio-classification (kapre version)*, https://github.com/seth814/Audio-Classification, Apr. 2020.

[33] J. Lyons, *Mel frequency cepstral coefficient (mfcc) tutorial*, http://practicalcryptography.com/miscellaneous/machine-learning/guide-mel-frequency-cepstral-coefficients-mfccs/, Published Date is not specified.