Proceedings of the Sixth International Conference on Inventive Computation Technologies [ICICT 2021]
IEEE Xplore Part Number: CFP21F70-ART; ISBN: 978-1-7281-8501-9

An Analysis of Audio Classification Techniques using Deep Learning Architectures

Mohammed Safwat Imran
Dept. of Computer Science
Brac University
Dhaka, Bangladesh
safwatimran@gmail.com

Afia Fahmida Rahman
Dept. of Computer Science and Engineering
Brac University
Dhaka, Bangladesh
afia.f.rahman@gmail.com

Sifat Tanvir
Dept. of Computer Science and Engineering
Brac University
Dhaka, Bangladesh
sifattanvirsami@gmail.com

Hamim Hassan Kadir
Dept. of Computer Science and Engineering
Brac University
Dhaka, Bangladesh
hamim.kadir@gmail.com

Junaid Iqbal
Dept. of Computer Science and Engineering
Brac University
Dhaka, Bangladesh
ishmamjunaid@gmail.com

Moin Mostakim
Dept. of Computer Science and Engineering
Brac University
Dhaka, Bangladesh
mostakim@bracu.ac.bd

2021 6th International Conference on Inventive Computation Technologies (ICICT) | 978-1-7281-8501-9/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICICT50816.2021.9358774

***Abstract* - Failure to classify audio data with high efficiency causes major setbacks in audio processing, voice recognition, and noise cancellation. To find the best possible neural network models for audio classification, this article shows the steps in the experiments performed in the newly designed CF Model and CF Clean Model in both CNN and RNN and compares the results with some existing models such as DCNN and PiczakCNN. To get a clear view of the consistency of the results, three different datasets have been experimented on, which are UrbanSound8k, FSDKaggle2018 and ESC-50. It also performs best in terms of the training and testing dataset based on accuracy and loss percentage. Finally, this article shows influences about the envelope function, normalization, segmentation, regularization techniques, and dropout layers in the overall progress.**

***Index Terms*—Audio Classification, Neural Network, Deep Learning, Convolutional Neural Network, Recurrent Neural Network**

I. Introduction

A. 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. In our everyday life, whenever using the electronic media, a lot of background sounds are heared apart from the user's voice from the other end. Disturbance due to noises while communicating remotely can serve as fatal at times because noises can create confusion, misinformation, or misinterpretation of information. To perfectly distinguish between noises and actual sounds, a reliable method of classifying audio data is a must. Classifying audio can be used for 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 evidence. Therefore, in many sectors, audio processing, refining or noise cancellation are applied, and audio classification serves as the main basis of these.

B. Aims and Objective

The following are the aims and objectives of this article:

a) Multiple Neural Network models of both Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN) have been compared in classifying audio data.
b) The constrained variables for the models' performance have been sought and pinpointed why certain models perform better than others. A comparison among different datasets has been made to determine the quality of these datasets and how the difference in the characteristics of each dataset influences the overall performance of the classification models.
c) Envelope function, normalization, segmentation, regularization, and dropout layers have been used and observed to determine whether or not these techniques enhance the overall audio classification.

The rest of the article is organized as follows: Section II includes the various related works done in similar fields. Section III explains the methodology of our approaches in classifying audio data. Section IV describes the results. Section V explains the findings of our thesis, which is the culmination of all our experiments. Finally, Section VI concludes the research work with a brief idea of the future work.

II. Related Works

Many significant contributions have been made in the same or related 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, the Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN) were used to classify audio data, which are models for processing datasets based on deep learning.

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 trying to pursue is feasible and possible, the main objective is to use CNN and RNN in such a manner that our system learns from the data provided.

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 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 as 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 [27] 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 over time and apply a mask to capture the harmonic content. In his algorithm, the author 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.

Several related works have also worked on the datasets. In [28], Urban8k Dataset was used to prove that an important piece for resolving (music) audio problems are the architectures alone using deep neural networks. Extreme Learning Machine (ELM) model has been used and an accuracy of 89.82% was achieved using temporal models. In [26], a convolutional neural network and tensor deep stacking network were used to classify the environmental sounds of the ESC-50 dataset based on the generated spectrograms of these sounds. In [14], the authors tried to prove with the ESC-50 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 the DCNN model using UrbanSound8k and ESC-50 datasets. Authors in [8] also worked with the UrbanSound8k and ESC-50 datasets and classified audio using the 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 would be aiming to surpass in our work.

III. METHODOLOGY

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

A. Audio Pre-Processing

Before starting the experiments, the datasets were preprocessed. The pre-processing was done using normalization, Fast Fourier Transformation, Short-Time Fourier Transformation, Mel Filterbank, and Mel Frequency Cepstral Coefficient.

1) Signal Normalization: Audio files have different shapes of waves in a file. Microphones usually have a bit depth of 16 which can generate 2 to 16 integers in a time domain to generate waves. To normalize the signals apply a 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 use pre-emphasis filters on the signal data. In a signal x, apply the following equation:

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

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

2) Fast Fourier Transformation (FFT): Wave signals of the audios are difficult to read and recognize for further work. To represent the data in different ways use the Fourier Transformation which generates a Frequency-Magnitude graph called Periodogram. In this case, specifically, Fast Fourier Transformation is used to generate a periodogram. A periodogram is easier to understand the audio signals. A periodogram plots the maximum frequency of 22 kHz over Magnitude. The reason behind 22 kHz is, our microphone records audios from the environment highest at 44.1 kHz. To represent this it generates Nyquist frequency which is half of the sampling frequency. Here, obtained 22 kHz frequency from our environment by using our microphone. Any

frequency above a 22 kHz for our microphone will not be read. A spectrogram is produced which is a stack of periodograms adjacent to each other over time and also generates an image based on the magnitude of frequency to give an idea of the visual representation of an audio file. For spectrograms, generate 4 spectrograms for each $1/10^{th}$ of a second of the audio file. This contains enough information to generate Spectrogram.

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 audio signal changes over time. It is a continuous process. In Short-Time Fourier Transform initially to split the sample signal into small frames and step forward the frame window and take 1 second of it. Now, split the audio into frames and each frame has a length of 20-40 ms. For our experiment, 25 ms is used and this frame takes 10 ms forward. Now the new frame has 15 ms in common with the previous frame. Here, it overlaps to process and generates 400 samples. The idea behind this is to have a small frame which is enough to have maximum information so that 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 to generate contours to assume that to receive a static audio data. After slicing the data into frames, apply the window function. Overlaps of the frames generate bell curves which is called a Hamming Window. It is a window function that is used in FFT and helps to taper the wave shape of the frames where there are 2 overlaps. This is to avoid spectral leakage and counteract of the assumption of FFT which is infinite. The equation of a Hamming window is:

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

where $0 \leq n \leq N$ and N = the window length [4].

This gives us 400 points of FFT but for FFT the value based on the power of 2. Typically 512 are used, so after 400, it adds 112 zeros. After this, it generates periodograms for the further filtration process.

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 70 Hz and which one is 170 Hz. When the frequency gets higher, the human cochlea, which is an organ inside the ear that reacts on different frequencies, cannot differentiate. For example, 17070Hz and 17170 Hz human cochlea cannot differentiate. Mel filter bank works on this principle. It distinguishes low frequencies in short-range but expands its range when the frequency is higher. In this experiment used a total of 26 Mel filters. But depending on purposes it increased till 40 usually. The equations [34] a total of 26 filers are used for the following:

$$M(f) = 1125 * \ln\frac{1+f}{700} \quad (3)$$

$$M^{-1}(m) = 700 * (\left(exp\frac{m}{1125}\right) - 1) \quad (4)$$

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

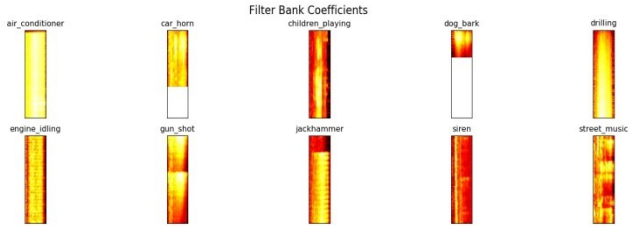

Fig. 1: *Filter Bank Coefficients of the classes*

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 decorrelate a lot of energies from the previous energy band. Discrete Cosine Transform (DCT) is a technique that is highly used for image and audio compression. In DCT compression it discards specific bits that contain high coefficients to compress data. This is called applying a low pass analog filter. This reduces high-frequency data and smoothens 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 compressing the higher frequencies to lower frequencies. This allows different sounds from having identical images. Data pre-processing phase completes by the end of the MFCC. Figure 2 shows the images generated after this stage.

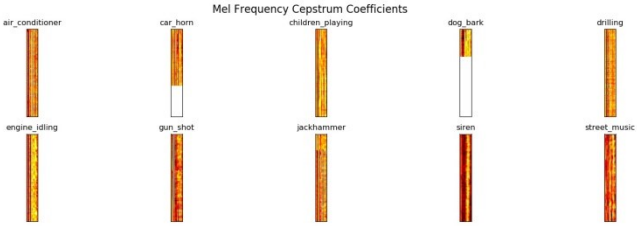

Fig. 2: *Mel Frequency Cepstrum Coefficients*

B. Classification Models

The MFCC function discussed in the previous section is what yields the spectrograms illustrated in figure 2. This allows the sequential, audio data to be depicted as image data to our deep learning algorithms. The algorithms used are as follows:

1) Convolutional Neural Network (CNN): Convolutional Neural Network is a neural network with multiple layers stacked up which includes pooling layers, convolutional layers, etc. [23].

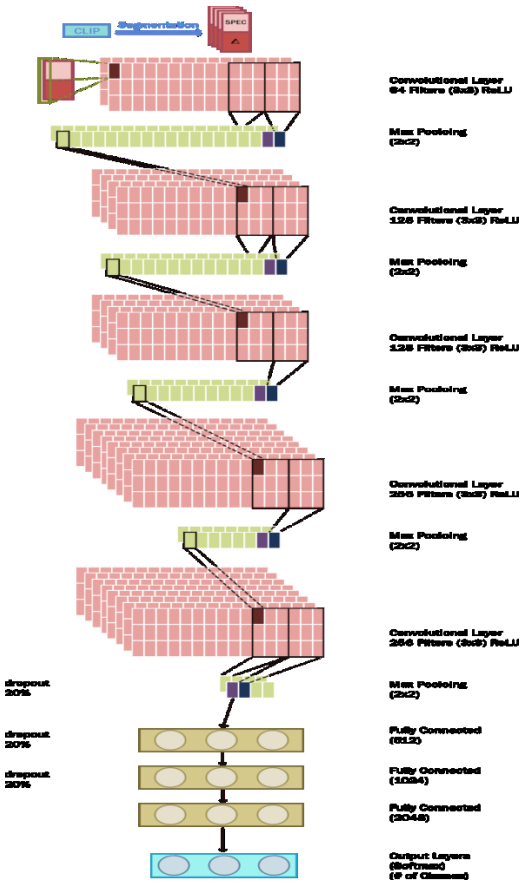

Fig. 3: *CNN Architecture*

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.

A total of five 3 x 3 convolutional layers, each followed by a 2 x 2 pooling layer were used to learn the features. The 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]. The Adam [15] function is used to optimize the learning rate by passing a coefficient of 0.0001. Categorical cross-entropy along with softmax components [17] has been to lower down the losses associated with CNN. Figure 3 shows a representation of the CNN Architecture used in our proposed models.

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. This system is used to compute consecutive units and their data in the most robust form possible as they are all sequential and dependent on each other. The dropout operator is used with a coefficient of 0.5. The proposed system made use of two RNN layers followed by four ReLU units [12] to help avoid overfitting. Like our CNN system, this system too has made use of the component's categorical cross-entropy alongside softmax to help reduce losses while training the datasets. Figure 4 shows a representation of the RNN Architecture used in our proposed models.

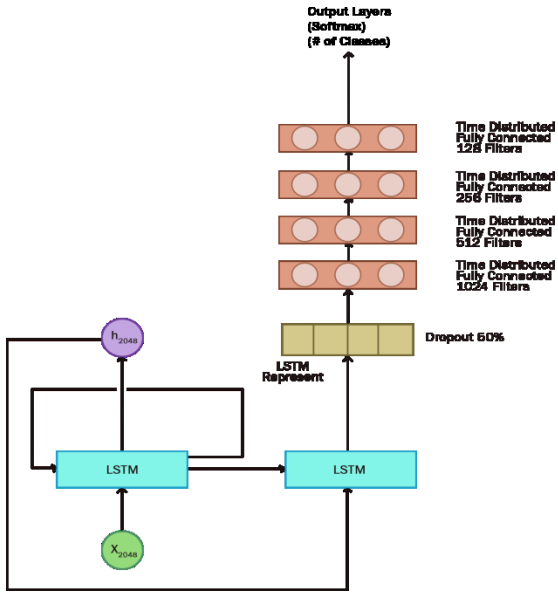

Fig. 4: *RNN Architecture*

IV. Experimental Analysis

A. 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, gunshot, 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 a 44.1 kHz sampling rate. Samples were chosen at random from the distribution during each cycle of training for both models.

B. *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 as '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 a batch size of epochs 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 11 GigaByte memory.

1) CF Model: The 'CF' (acronym for classify) model was having 44.1 kHz as the sampling rate of the audio clips, taking each audio clip as one sample input for training, not accounting for the 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.

2) CFClean Model: The audio clips were downsampled to 16 kHz 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 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 was 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 a 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, [29]. [33] includes the code of the experiments done using both the CF model and CFClean model and figure 5 shows the comparative workflow of the two models.

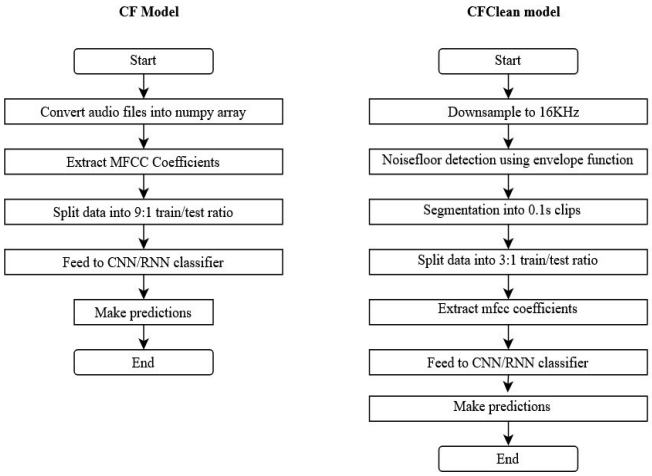

Fig. 5: *CFModel and CFClean Model*

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

TABLE I: Model designs

Model	Dataset	Number of samples		Learning Rate	Normalization	Optimizer
		Train	Test			
CF	UrbanSound8k	6549	2183	0.0001	No	Adam
	FSDKaggle2018	7104	2369			
	ESC-50	1500	500			
CFClean	UrbanSound8k	155227	17248		Yes	
	FSDKaggle2018	1008384	112043			
	ESC-50	155207	17246			

A smaller test sample was taken for the CFClean model compared to the CF model because the total number of samples available in the former were much higher, due to segmentation. Using a 9:1 ratio on the CF model would not be suitable as the number of test samples would then be too little for the results to be reliable.

C. *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 [25]. It is also more difficult to optimize RNN due to vanishing and exploding gradient problems and also longer training times, [30]. 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 labeled 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.

D. *Tabulated Results*

In table II, the results are compared against other research works and the accuracy results on these datasets were displayed.

TABLE II: Accuracy and loss values for our proposed models

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	-	-

From table II, 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].

V. FINDINGS

In this section, the findings of the research work were explained.

A. *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 ESC50 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 II and 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 because RNN's are better at tasks that rely on the context of previously found values and follow a certain sequence, through the use of feedback loops [25]. In Table II, both the 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. CNN's are widely known to outperform RNN's in classification problems, especially in image recognition [26]. The classification of the FSD Kaggle 2018 dataset suffer due to the asymmetry of the distribution of classes in the dataset and also the chances of external noise other than the labeled class being present in the samples, given they were not manually verified. Table II process the data, clean the data and obtains the accuracy of the RNN architecture to improve prediction accuracy by 22.67%.

B. *CFClean model has outperformed the CF model*

The CFClean model performed better than the CF model in every dataset. In Table II, the terms of test accuracy CFClean model have always outperformed the CF model.
CFClean model classified audio with the highest accuracy of 94.52% and the lowest of 64.21% whereas the 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 the CF model overcomes the 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 the CF model, and the test accuracy being the deciding factor in calculating the overall efficiency, and concluded that the CFClean model has been more efficient.

C. *Envelope function, normalization, and segmentation have enhanced overall performance*

Normalization of the matrices' input into the model has 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 implement that the model is performing poorly, the difference in train and test accuracies indicates that some amount of overfitting has occurred. The removal of dead space in the audio using the envelope function, segmentation of audio clips, and normalization of the input matrix has helped the classification process tremendously. As mentioned in the previous section, the CFClean model has been more efficient than the 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.

D. *Classification on UrbanSound8k dataset has been more effective than on FSDKaggle2018 and ESC-50 datasets*

From table II, 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 FSD Kaggle 2018 dataset has sound clips that 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 labeled 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 of 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 labeled 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.

E. *Regularization techniques and adding more dropout layers have not been helpful*

Even with regularization techniques or adding more dropouts 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 the 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.

VI. Conclusion and Future Work

The goal of this paper was to find the most suitable audio classification model and the dataset after trying out different combinations of models and datasets. CFClean model in CNN architecture has had the best accuracy, especially on the UrbanSound8k dataset. Maximum accuracy of 94.52% has been achieved which is quite satisfactory and better than the accuracy of other existing methods. In the future, there are some 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 [27]. However, we an attempt is made for our techniques in achieving this task and eventually find more efficient methods to isolate noise, speech, and instrument sounds, or whatever is desired from other backgrounds or external noises are present. In this article, the differences in the performance of CNN and RNN in audio classification were summarized. As our work interests revolve around audio classification, detection, and noise marginalization using neural networks to work with modified versions of the neural networks. For example, adding capsules to CNN to create Capsule Neural Networks [31] and other related evolving deep learning neural network architectures [24].

References

[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: 10.1007/9780-387-30441-0_29.

[3] H. Lee, P. Pham, Y. Largman, and A. Ng, "Unsupervised feature learning for audio classification using convolu-tional 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 classifi-cation," *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 Classifi-cation*, 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] 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.

[24] A. Bashar, "Survey on evolving deep learning neural network architectures," *Journal of Artificial Intelligence and Capsule Networks*, vol. 2019, pp. 73–82, Dec. 2019. DOI: 10.36548/jaicn.2019.2.003.

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

[26] 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.

[27] 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.

[28] 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.

[29] D. Singh and B. Singh, "Investigating the impact of data normalization on classification performance," *Applied Soft Computing*, p. 105524, 2019, ISSN: 1568-4946. DOI: https://doi.org/10.1016/j.asoc.2019.105524. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1568494619302947.

[30] 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.

[31] T. Vijayakumar, "Comparative study of capsule neural network in various applications," *Journal of Artificial Intelligence and Capsule Networks*, vol. 01, pp. 19–27, Sep. 2019. DOI: 10.36548/jaicn.2019.1.003.

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

[33] M. S. Imran, *Safwatimran/audio-classification*. [Online]. Available: https://github.com/SafwatImran/Audio-Classification.

[34] 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.