

# Extensive Study of Speech Emotion Recognition(SER) using Mel Spectrograms

Shiva Kalyan Sunder Diwakaruni

*Department of Computer Science and Engineering,  
Jawaharlal Nehru Technological University, Hyderabad, India*

Mohammed Abrar Pasha

*Department of Electronics and Communication Engineering,  
Jawaharlal Nehru Technological University, Hyderabad, India*

Nissankara Manas Anand

*Department of Electronics and Communication Engineering,  
Jawaharlal Nehru Technological University, Hyderabad, India*

Dr. Supreethi K.P.

*Department of Computer Science and Engineering,  
Jawaharlal Nehru Technological University, Hyderabad, India*

**Abstract-** This paper describes an ameliorated method of Speech Emotion Recognition(SER) using the Deep Neural Network architecture which includes the implementation of CNN layers alongside the usage of Adaptive and Linear Layers in the model. The speech feature Spectrogram is being used as they form the most effective way to input data into the Deep Learning Models. Since our model expects all samples to have equal proportions, we have transformed all the mono files to stereo by duplicating the first channel to the second. Raw audio data was standardized by either truncating the samples or padding them with silence to maintain uniformity among the data samples. In addition to this, the sampling rate was also standardized to have the same dimensions. The developed trained model achieved a best accuracy of 72% on complete classification.

**Keywords-**Speech Emotion Recognition(SER), Spectrograms, Deep Neural Network(DNN), Transfer Learning.

## I. INTRODUCTION

Speech Emotion Recognition is the method of identifying emotional quotients of speech irrespective of the actual semantic contents. It is intuitive of humans to easily perceive emotions as a part of a natural process from quite an early age. However, inculcating the same into machines is something that the researchers and scientists have been trying to do for quite some time now. Modernization aims at helping humans lead an easier life through innovative approaches which include replacing humans with machines or robots that could interact and provide viable solutions to the problems that arise. Hence adding emotions to machines has proved to be a breakthrough approach in transforming machines to act more like humans. In order for machines to have an emotional connection with humans, it is paramount for the innovators to provide with not only with understanding of what we speak but also a crystal-clear comprehension of the emotions that are to be conveyed. This could warrant a fully meaningful conversation between a human and a machine and there is always a sense of trust between the parties involved. Perceiving emotions could guarantee appropriate responses from robots. Implementation of age detection by using SER is already in practice to restrict children below a certain age limit from playing games that include violence. The device in hand recognizes the emotions and classifies the age according to the frequency of the speech signal. Once the age is estimated, the device can decide if the determined age is of the permissible value.

Section 2 walks you through the related work carried out in order to derive inferences and come up with innovative approaches to perform the experiment. Section 3 deals with the methods involved in the study. Section 4 explains the experimental setup and the results of the study and finally the Conclusion is mentioned in section 5.

## II. RELATED WORK

There has been a tremendous upsurge in the utilization of Deep Learning Models in speech recognition. Novel approaches have taken over the conventional methodologies for SER. There has been an abundance of work related to the utilization of spectral features that has been widely published. The Mel Spectrogram has proved out to be a revelation in effectively determining emotions from speech.

In [1], Ruhul Amin Khalil shed some light on why Deep Learning Techniques have come to the forefront and are being considered as front runners for speech-based emotion recognition. It also covers databases used, emotions extracted and the contributions made towards SER.

Yixiong Pan in [2] used SVM for 3 class emotion classification on Berlin Database of Emotional Speech [3] and achieved 95.1% accuracy which to our knowledge, is the best result published in this particular area of research.

Pavol Harar in [4] describes about the usage of Deep Neural Network (DNN) architecture with conventional, pooling and fully-connected layers. It reported an accuracy of 69.55% on file prediction on three classes of German Corpus (Berlin Database of Emotional Speech). Suraj Tripathi in [5] proposed an SER method based on speech features and speech transcriptions(text). The combined spectrogram text-model gave a class accuracy of 69.5% and an overall accuracy of 75.1% whereas the combined MFCC text model gave the same class accuracy but the overall accuracy increased to 76.1%. The research was performed on the IEMOCAP dataset and was considered to be the most accurate identification.

Margaret Lech in [6] worked on instantaneous recognition of speech using a pretrained image classification network. AlexNet was used in this experiment. The results showed an overall accuracy of 82% when trained on the Berlin Emotional Speech (EMO-DB) with 7 classes of emotions

Maheshwari Selvaraj in [7] used both spectral and prosodic features for SER as both contained emotional information. Mel-frequency Cepstral Coefficients is used as one of the spectral features. Radial Basis Function and Back-Propagation Network are used to recognize emotions based on selected features and it was proven that the Radial Basis Function came out on top.

In 2019, Sandeep Kumar Pandey in [8] utilized various deep learning techniques to capture the emotional state. Architectures such as Convolutional Neural Network (CNN) and Long-Short Term Memory were used to test the capturing capability from a number of speech representations such as Mel Spectrogram, Magnitude Spectrogram and MFCC on two datasets namely EMO-DB and IEMOCAP.

H.M. Fayek in [9] used DNN to recognise emotions from one second frames of raw speech spectrograms on eINTERFACE and SAVEE database. Their experiment achieved an accuracy of 60.53% on all the six emotions

Going through these works filled us with the required erudition to explore further about the implementation of the SER techniques.

## III. MATERIALS AND METHODOLOGIES

### 3.1 Dataset-

Every machine learning task requires a training set of samples and SER is no exception to it. Three different types of databases have been defined for speech emotion recognition namely, simulated, semi-natural and natural speech collection. Labels of trained speakers reading same text with different emotions form majority of the simulated database. Semi-Natural collections consist of recordings of actors while enacting their roles displaying varied emotions. On the other hand, Natural Datasets are accumulated from TV shows, YouTube videos as such and the emotions are then labelled by human listeners.

Majority of the datasets that we have used in our research fall under the category of Simulated Collections. They include, EMO-DB (Berlin Emotional Speech Database), TESS (Toronto Emotional Speech Database),

RAVDESS Emotional Speech Database and CREMA-D and SAVEE. These are the standardized collections of emotions and this makes the process of comparing results easy.

The collections from the aforementioned datasets were amalgamated to form a single entity. A brief explanation about each of the datasets is given below.

EMO-DB (Berlin Emotional Speech Database) [3] - It comprises of recordings from 10 professional speakers (5 men and 5 women). There are a total of 535 utterances which consist of seven different emotions (anger, boredom, anxiety, happiness, sadness, disgust, neutral). The age group of the masses involved ranges from 21-35 years and every utterance is named according to a similar scheme.

TESS (Toronto Emotional Speech Database) [10] - This is a female-only database involving two women (young and old) with high quality audio. A total of 2800 audio files are present in it, and these portray seven emotions which include (anger, disgust, fear, happiness, pleasant surprise, sadness and neutral).

RAVDESS Emotional Speech Database [11] - Consisting of 1440 audio files from a set of 24 professional actors (12 females and 12 male), this database vocalizes two lexically matched statements in a neutral North American accent. There are a total of 8 emotions which include calm, happy, sad, angry, fearful, surprise and disgust. There are two levels of emotional intensities (normal, strong) with one additional neutral expression.

CREMA-D (Crowd-Sourced Emotional Multi-modal Actors Dataset) [12] - It contains 7442 audio files from 91 actors. The recordings were from 48 male and 43 female actors between the ages of 20 and 74 originating from different regions and hence constituting a variety of accents. The actors have spoken from a collection of 12 sentences and emotions are conveyed in one of the following six ways (anger, disgust, fear, happy, neutral and sad) and they are on four different levels namely, low, medium, high and unspecified.

SAVEE Database [13] - The recordings were taken from four male speakers and their ages ranged from 27-31 years. Emotions were classified into six categories (anger, disgust, fear, happiness, sadness and surprise). There are nearly 120 utterances per speaker and consists of 481 audio files.

When all the five databases were combined and all the emotions were categorized accordingly, it was observed that certain emotions were required to either be neglected or be put under another category, both owing to low or negligible amount of data. From among the various categories present in the datasets that are being considered, the number of labels were negligible for surprise and hence this emotion was neglected. On the other hand, due to an acoustic similarity, boredom was placed under the emotion sadness. Another reason for this was that it made more sense to position two negative emotions alongside each other. Fig.1 depicts the distribution of six classes of emotions after pre-processing.

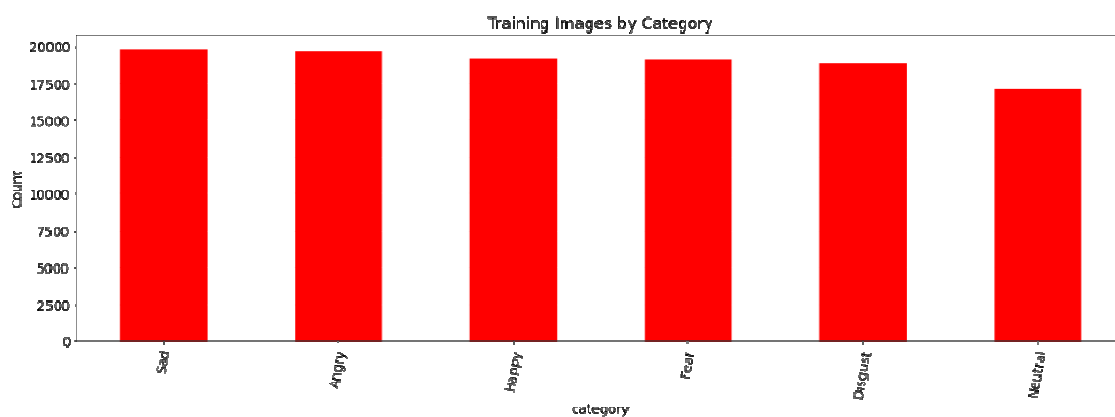


Figure 1. Distribution of Audio Files across 6 Emotions

### 3.2 Data Preprocessing-

As mentioned earlier, all the datasets were combined to form a database consisting of data pertaining to various categories of emotions. Now this was further divided into three parts namely Training Data, Validation Data and Testing Data.

A total of 11853 audio files are present.

Training Data: This comprised of a total of 60% of the data in the dataset. Around 7111 audio files fall under this category.

Validation Data: 20% of the database comprises of the validation data and this accounts for around 2371 audio files.

Testing Data: Like the validation data, this also accounts for 20% of the dataset and hence is equivalent to around 2371 audio files in quantity.

Table 1-Classes of emotions and the respective number of Mel spectrograms generated and distributed across three categories

Category	Training Data	Validation Data	Testing Data
Neutral	17104	357	356
Happy	19136	399	399
Sad	19824	414	413
Fear	19120	399	398
Angry	19680	410	410
Disgust	18896	394	394

### 3.2.1. Reading Audio from a file-

The first step involves reading and loading of the audio file which is in “.wav” format. We do this by using the “Torch Audio Transforms” function from the PyTorch library. Once the audio file is read, the signal is returned as a tensor and with a sample rate.

### 3.2.2. Channel Conversion-

While considering the audio files, it is paramount to standardize the channels since our model expects the channels to have the same dimensions. Some of the audio files are mono i.e., single channel and some are stereo i.e., two channels. In our model, we are converting the mono files to stereo by duplicating the first channel to second.

### 3.2.3. Sampling Rate Standardization-

Sampling rates of the audio files in hand is between (24000-28000) Hz so to have uniformity, we resampled these such that all sound files have the sampling rate of 16000Hz. This implies that 1 second of audio will have an array of 16000 samples all the sound files. We have created a “Resample” function wherein the first channel is resampled followed by resampling of the second channel and then both the channels are merged.

### 3.2.4. Resizing the audio files to the same length-

Audio files of the dataset are of varying lengths. To standardize these, we pad (or truncate) the signal to a fixed length. In our case, we have standardized the length to 5 seconds. If the length of the signal is more than 5s, we truncate the signal to the mentioned value and if the length is lesser than 5s, the audio file is padded with silence at the beginning and end of the signal.

Training Data constitutes for around 7111 audio files which are to be increased to an amount that could help in obtaining far more efficient results. In order to do so, we employ the method of Time Shift. This is a Data Augmentation technique done on raw audio signals. The audio signals are shifted (time-shifted) to the left or right by certain amounts in a random order by a certain percentage and values at the end are ‘wrapped around’ to the start of the transformed signal. This helps in doubling the amount of Training Data. All the above-mentioned steps are shown in Fig.2.

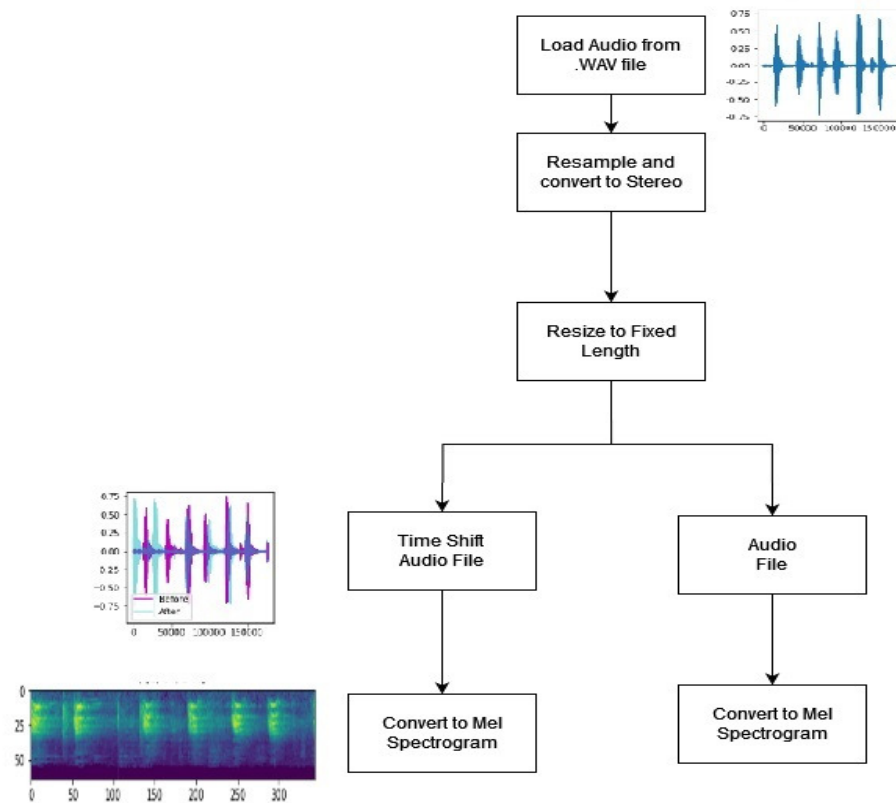


Figure 2. Stepwise Flow of Pre-processing (Part-1)

### 3.2.5. Mel Spectrogram

It is believed that the deep learning models seldom take raw audio signals as input. In order to overcome this barrier, the audio signals are converted into something called as a spectrogram. A spectrogram is defined as a concise snapshot of the audio signal and since it is an image, it would be better suited to act as an input to the CNN based models.

Spectrograms are generated from the sound signals by using Fourier Transforms. The job of the Fourier transform is to decompose the given signal into its constituent frequencies and to display the amplitude of each frequency present in the signal.

A Spectrogram chops up the duration of the sound signal into smaller time segments and then applies the Fourier Transform to each segment, to determine the frequencies contained in that segment. It then combines the Fourier Transforms for all those segments into a single plot.

In our experiment, we have made use of the “Mel Spectrogram transform” function of the python library to generate the spectrograms when audio files are supplied. The values of “n\_mels” and “n\_fft” are to be given accordingly and they are 128 and 2048 in this case. “n\_fft” corresponds to the number of samples which further corresponds to a physical duration of 93 milliseconds at a sample rate of 22050Hz. Hop Length is defined as the number of samples between successive frames for example, the columns of a spectrogram and it is always given as a positive integer. The “hop\_length” in our case is taken as 512.

Once all the audio files are converted, we now have a set of 14222 spectrograms.

To further multiply the amount of trained data, we made use of Time and Frequency Masking on the Mel Spectrograms.

In Frequency Masking, a range of consecutive frequencies are masked randomly by adding horizontal bars on the spectrogram.

Similarly, in Time Masking, we block out ranges of time in a random order in the spectrogram by using vertical bars. Augmentation in this way is done to prevent over-fitting and help the model generalize better. The masked sections are replaced with the mean value.

The Frequency Masking parameter indicates the maximum possible length of the mask and the indices are uniformly sampled from  $[0, \text{freq\_mask\_param}]$ . Time Masking Parameter also indicates the maximum possible length of the mask. Indices are from  $[0, \text{time\_mask\_param}]$ . The parameters used by us are “n\_freq\_masks” and “n\_time\_masks” respectively.

During Time Masking, the value of n\_time\_masks is given as ‘2’ to ensure complete time-masking of the dataset to produce a total of 14222 time-masked spectrograms. These add up to 28444 spectrograms in total. Similarly, giving the “n\_freq\_mask” value as ‘2’ we can mask the spectrograms with respect to the frequencies and produce another 14222 frequency-masked spectrograms. Giving the values of the both the parameters as 1 we have produced another set of spectrograms with 14222 images which were both time and frequency masked. So all in all, in the end, we had 56888 spectrograms for training. It can be observed that through these data augmentation methods, we could increase the data at hand eight-fold.

We started off with 7111 raw audio files and implemented resampling, rechanneling and padding/truncating to a fixed length of 5 seconds on these audio files. We then converted them into Mel Spectrograms and then applied Time Masking, Frequency Masking and Time-Frequency Masking and then ended up with around 28444 spectrograms. Figure 3 depicts the mentioned process.

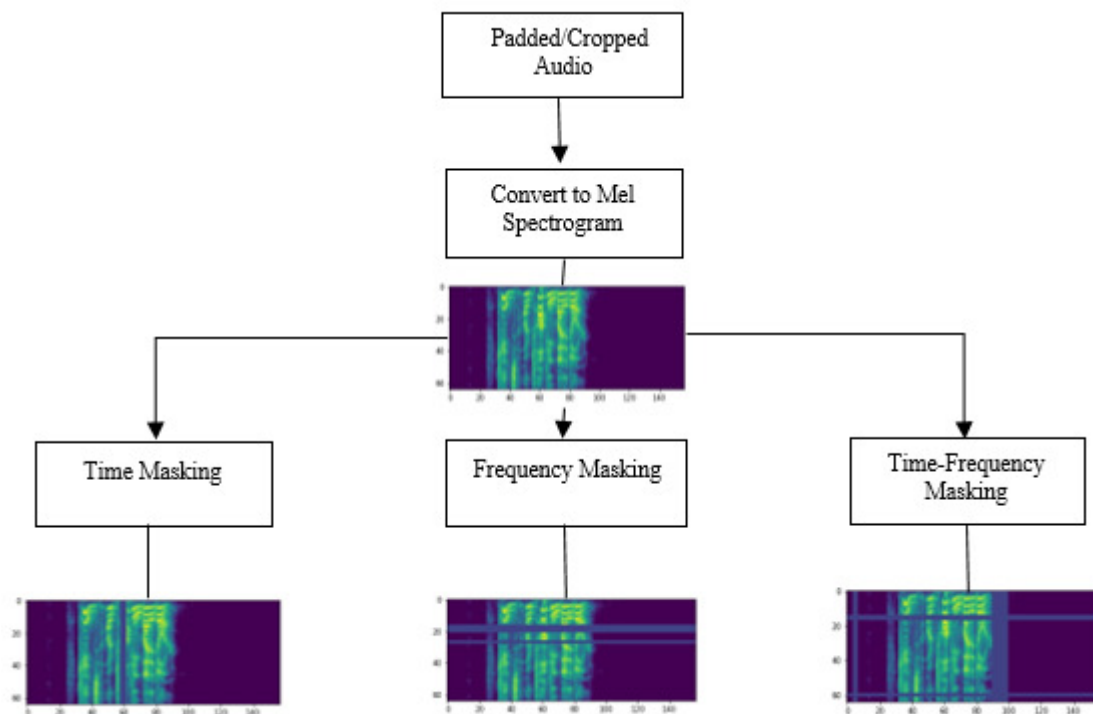


Figure 3. Stepwise Flow of Pre-processing (Part-2)

To double the data, we time-shifted the 7111 raw audio files and repeated the same preprocessing steps (Resampling, Rechanneling, Padding/Truncating) as shown in Figure 2. The differences in the Mel Spectrograms before and after time-shifting is evident from Figure 3 and Figure 4.

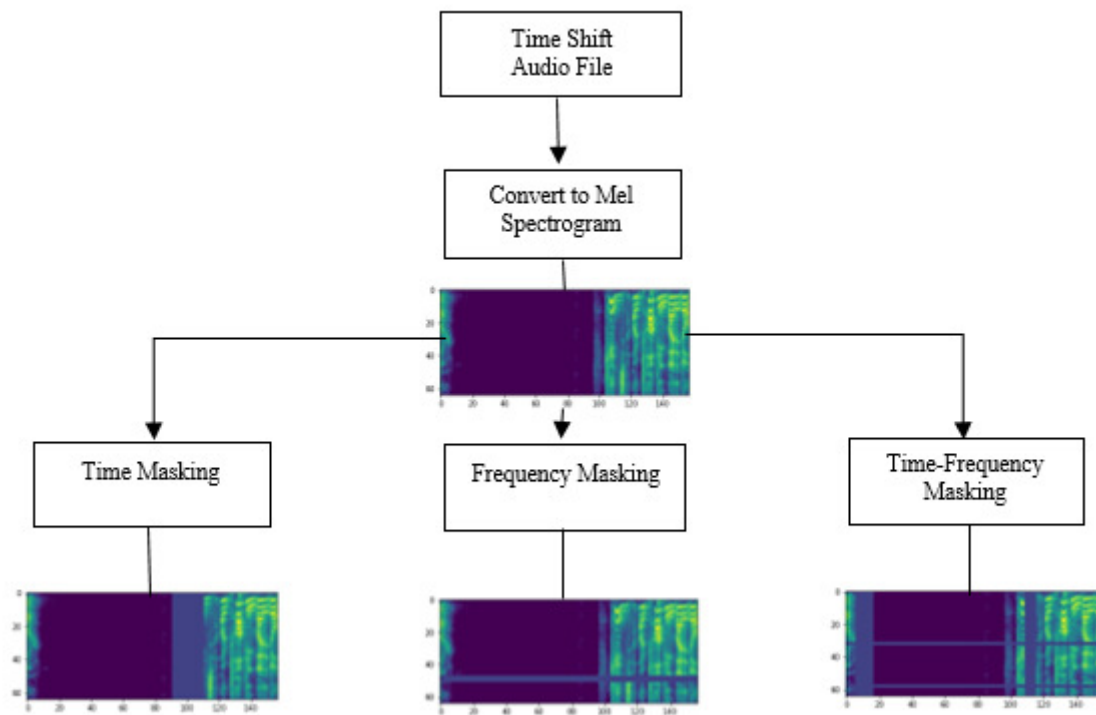


Figure 4. Stepwise Flow of Pre-processing (Part-3)

So this created 56888 spectrograms with a certain set of parameters. To further increase the data, we made changes in the parameters such as “n\_fft” and “hop\_length”. The values of “n\_fft” and “hop\_length” which were “2048” and “512” earlier were changed to “1024” and “0” respectively to help generate another 56888 spectrograms. By the end of this, we managed to collect around 113776 spectrograms.

### 3.2.6. Normalization

After generating the spectrograms, the next step is to normalize them, and we do this by using two methods Namely Z-Score Normalization and Min-Max Scaling [14]

It can be observed that a number of machine learning algorithms attempt to find trends in the data by comparing features of data points. However, we have a problem when the features are on drastically different scales. To overcome this, we make use of the normalization techniques. The goal of normalization is to make every data point have the same scale such that each feature is equally paramount.

### 3.2.7. Min-Max Normalization

This is one of the most common methods of normalizing data [14]. For every feature, the minimum value of that feature gets transformed into a 0 and the maximum value gets transformed to 1 and every other value gets transformed into a number lying between 0 and 1.

The formula for Min-Max Scaling is given below,

$$scaled\_data = [data - \min(data)] / [\max(data) - \min(data)]$$

### 3.2.8. Z-Score Normalization

There is a downside to the Min-Max scaling technique, and this could be overcome by using Z-Score Normalization [14]. Min-Max Scaling cannot handle outliers well, so Z-Score Normalization is a method of normalizing data by avoiding the outlier issue.

The formula for Z-Score Normalization is given below,

$$scaled\_data = [data - mean(data)]/std(data)$$

In our experiment, we have normalized the spectrograms using both the aforementioned normalization techniques and the scaled values lie between 0 and 255.

### 3.3 Transfer Learning-

A pretrained model is the one that has been trained previously on a dataset and contains the weights and biases that represent the features of whichever data it was trained on. Learned features are always transferable to different data. We use pretrained models to save our time. The model is previously trained and computed to learn a number of features and hence our model will likely benefit from it.

In our experiment, we initially tried using the pretrained networks because they converge faster at a good accuracy. They come with a plethora of advantages that are listed below.

These models are super simple to incorporate and can achieve solid model performance quickly. The labelled data that they demand is not too high when compared to the rest. There are a varied number of uses like transfer learning, prediction and feature extraction.

The models that we have extensively used in our experiment are DenseNet161, ResNet50 and ResNet101.

#### 3.3.1 DenseNet161-

DenseNet161-Dense Convolutional Network (DenseNet) [15] connects each layer to every other layer in a feed-forward fashion as shown in Fig.5. In general, the traditional layers have  $L$  layers with  $L$  connections one between each layer and its subsequent layer, but DenseNet has  $L(L+1)/2$  direct connections. The advantages of DenseNet include alleviation of the vanishing-gradient problem, strengthening feature propagation, encouraging feature reuse and to reduce the number of parameters.

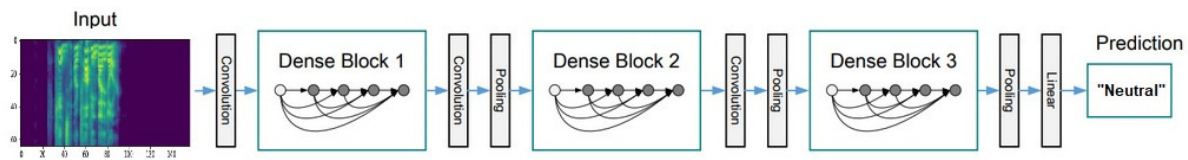


Figure 5. Depiction of connection of layers in DenseNet161

#### 3.3.2 ResNet50-

This is a convolutional neural network which is 50 layers deep [16]. It can be used to load a pretrained version of the network trained on more than a million images from the ImageNet database. This pretrained network can classify images into 1000 object categories and it has an image input size of 224-by-224. Fig.6 shows the layer-wise structure of this DNN.

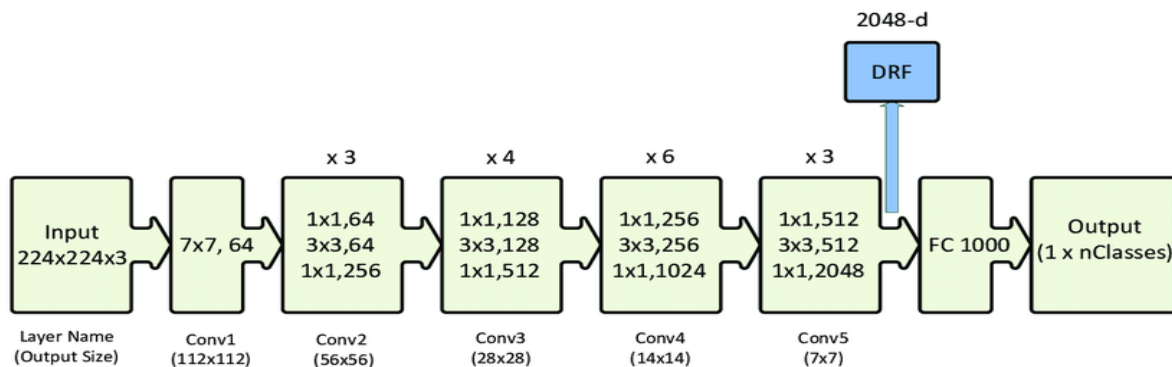




Figure 6. Depiction of structuring of layers in ResNet50

#### IV. EXPERIMENTAL STUDY

##### 4.1 Implementation-

For training our model, we utilized Stochastic Gradient Descent with a fixed learning rate of 0.01 to optimize a cross-entropy loss function also known as log-loss. The input data in the experiment was supplied to the DNN in batches of size 16 in multiple iterations. The segments contained by each batch were the same and each sample succeeded with a sample of different class.

In order to prevent over-fitting, we set the patience equal to 4. This meant that the experiment would be terminated if no progress on validation loss was made for more than 4 epochs of training. We utilized the capabilities of PyTorch [17] and sklearn frameworks to build the DNN model and we worked on accelerating the training on GPU (NVIDIA GeForce RTX 3700).

Using the trained model, we could derive the prediction probabilities for each class of each segment from validation and testing sets and then we took the maximum value from the predicted probabilities to denote the predicted class. It was paramount to deliver the final results for the whole audio files. In order to achieve this, we computed the average probability of all segments belonging to the particular file and used it to denote the final predicted class.

##### 4.2 Evaluation Parameters-

In the experiment, we have made use of a number of evaluation parameters and the description of each of these is given below.

**Confusion Matrix:** It is an “m x m” matrix [18] with actual values on one axis and the predicted values on the other. The terms associated with this matrix are true positive, true negative, false negative and false positive. There are formulae associated with each of these terms and even if the data is imbalanced, we can figure out that our model is working well or not. For this, the values of True Positive and True Negative must be high and False Positive and False Negative must be as low as possible.

Using the confusion matrix, other performance parameters can be calculated.

**Precision:** This is used to identify the amount of true positive present out of all the positive data that is present. [18]

$$Precision = \frac{TP}{TP + FP}$$

In this case, TP- True Positive    FP- False Positive

The value of precision always lies between 0 and 1.

**Recall:** This is defined as the amount of predicted positive out of total positive. [18]

The formula for recall is given below,

$$Recall = \frac{TP}{TP + FN}$$

Here, TP- True Positive and FN-False Negative

**F1 Score-** Harmonic Mean of precision and recall gives the F1 Score. This takes into consideration both False Positives (FP) and False Negatives (FN) and therefore performs well even on an imbalanced dataset. [18]

$$F1\ score = \frac{2}{\frac{1}{Precision} + \frac{1}{Recall}} = \frac{2 * (Precision * Recall)}{(Precision + Recall)}$$

#### 4.3 Findings and Discussion-

In order to perform 6-class (angry, disgust, fear, happy, neutral, sad) prediction on 113776 testing audio files of the combined database, we built a deep neural network model consisting of convolutional, pooling and fully connected layers. We trained it with raw, standardized input data cleared of silent segments using mini batches of size 16 on the GPU.

Three DNN models were used by us for the experiment. They were Densenet161[15], Resnet50[16] and Resnetxt50\_32x4d. In the first case using Densenet161, the DNN predicted 670 test files to belong to a wrong class as opposed to 1700 precise predictions. The overall accuracy for this DNN was found out to be 72%. The precision, recall, f1-score and support are shown in Table 1. If you observe carefully, the precision for identifying certain emotions like 'angry', 'fear' and 'neutral' was on the higher side around 80% whereas it was the lowest for 'happy' with only 55%.

When it came to Resnet50, the DNN predicted 718 data files to belong to a wrong class as opposed to 1652 correct predictions. And here the overall accuracy was 70%. Precision, recall, f1-score and support for this model is shown in Table-2. This DNN showed relatively higher precision for neutral and sad at around 77% and the precisions of all the other 4 emotions ranged between (65-68) %.

Resnetxt50\_32x4d predicted 770 data files to belong to a wrong class as opposed to 1600 correct predictions. The overall accuracy was 68% and the precision, recall, f1-score and support values for this model are shown in Table-3. This DNN showed higher precision for the emotion 'happy' and the lowest precision for 'angry'. The other four emotions' precision ranged between (69-76) %.

All in all, our DNN achieved a best accuracy of 72% on speech emotion recognition task on testing files.

Table 2-Evaluation Parameters for DenseNet161

	Precision	Recall	F1-Score	Support
Angry	0.83	0.68	0.75	356
Disgust	0.72	0.7	0.71	413
Fear	0.83	0.63	0.72	399
Happy	0.55	0.81	0.66	394
Neutral	0.81	0.8	0.8	410
Sad	0.7	0.68	0.69	398

Table 3-Evaluation Parameters for ResNet50

	Precision	Recall	F1-Score	Support
Angry	0.57	0.89	0.69	356

<b>Disgust</b>	0.69	0.69	0.69	413
<b>Fear</b>	0.74	0.51	0.61	399
<b>Happy</b>	0.76	0.52	0.62	394
<b>Neutral</b>	0.7	0.84	0.76	410
<b>Sad</b>	0.69	0.62	0.65	398

Table 4-Evaluation Parameters for ResNetxt50\_32x4d

	<b>Precision</b>	<b>Recall</b>	<b>F1-Score</b>	<b>Support</b>
<b>Angry</b>	0.65	0.84	0.73	356
<b>Disgust</b>	0.68	0.68	0.68	413
<b>Fear</b>	0.66	0.7	0.68	399
<b>Happy</b>	0.67	0.66	0.67	394
<b>Neutral</b>	0.78	0.76	0.77	410
<b>Sad</b>	0.77	0.56	0.65	398

## V. CONCLUSION

The main objective of this experiment was to predict the emotion of a person by taking into account a voice recording that lasted for 5 seconds. Our methodology achieved a best accuracy of 72% on the testing data. A total of five pretrained models were used by us to derive inferences and come up with to a conclusion regarding the best pretrained models.

Initially we tried to train the data using pretrained models such as AlexNet and VGG19 but this resulted in lower accuracy of prediction. On trying with other pretrained models such as DenseNet161[15], ResNet50[16] and Resnetxt50\_32x4d, we observed a spike in accuracy. On closer observation, we realized that it all depended on the number of layers of the pretrained models. Since the latter three models had higher number of layers, it resulted in a tremendous betterment in the accuracy.

The Training and Validation accuracies with respect to each of the models namely, DenseNet161, Resnet50 and Resnetxt50\_32x4d are shown in tables 5,6 and 7. Necessary inferences could be drawn from the graphs.

Additionally, we used the Adam Optimizer which manages the learning rate and Cross-entropy loss function which is then used to evaluate how different the predicted and actual data is and penalizes the model for poor predictions.

Table 5-Accuracies of DenseNet161

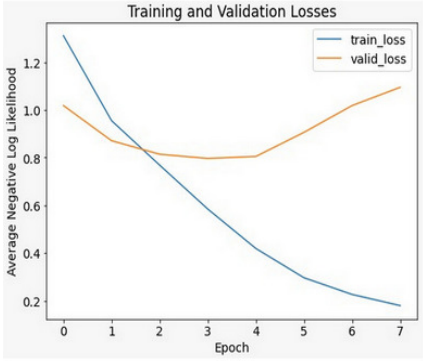
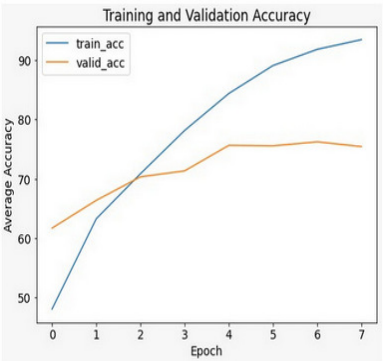
Accuracy	Loss Graph	Accuracy Graph
<div>Training Accuracy (93.46%)</div> <div>Validation Accuracy (75.43%)</div> <div>Testing Accuracy (72%)</div>		

Table 6-Accuracies of ResNet50

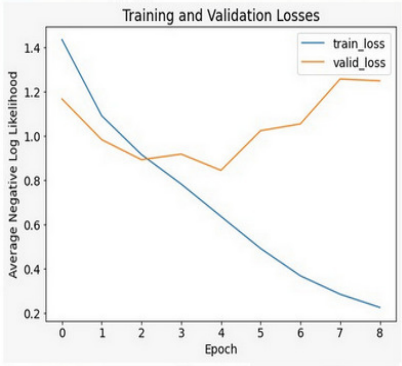
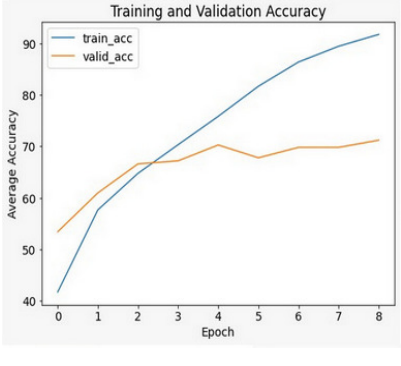
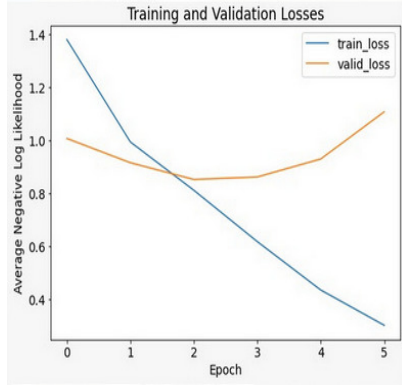
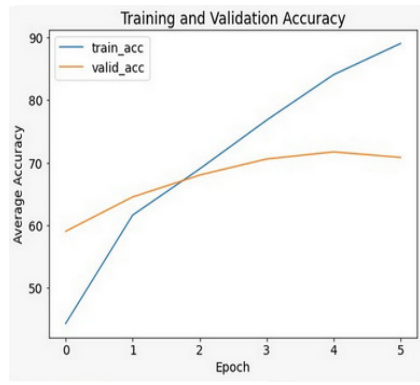
Accuracy	Loss Graph	Accuracy Graph
<div>Training Accuracy (91.74%)</div> <div>Validation Accuracy (71.18%)</div> <div>Testing Accuracy (70%)</div>		

Table 7-Accuracies of ResNetxt50\_32x4d

Accuracy	Loss Graph	Accuracy Graph
Training Accuracy (89.04%)  Validation Accuracy (70.84%)  Testing Accuracy (68%)	 <p>Training and Validation Losses</p>	 <p>Training and Validation Accuracy</p>

#### ACKNOWLEDGEMENT

The author would like to extend hearty thanks to the Jawaharlal Nehru Technological University Hyderabad for providing the platform to explore respective areas of interest through the guidance of our esteemed Professor Dr. Supreethi K.P (Department of Computer Science Engineering).

#### REFERENCES

- [1] R. A. Khalil, E. Jones, M. I. Babar, T. Jan, M. H. Zafar and T. Alhussain, "Speech Emotion Recognition Using Deep Learning Techniques: A Review," in IEEE Access, vol. 7, pp. 117327-117345, 2019, DOI: 10.1109/ACCESS.2019.2936124.
- [2] Pan, Y., Shen, P. and Shen, L., 2012. Speech emotion recognition using support vector machine. International Journal of Smart Home, 6(2), pp.101-108.
- [3] EmoDB Database by Piyush Agnihotri. (2020) <https://www.kaggle.com/piyushagni5/berlin-database-of-emotional-speech-emodb>
- [4] Harár, Pavol & Burget, Radim & Dutta, Malay. (2017). "Speech emotion recognition with deep learning". 137-140. 10.1109/SPIN.2017.8049931.
- [5] Suraj Tripathi, Abhay Kumar, Abhiram Ramesh, Chirag Singh, Promod Yenigalla (2019) Deep Learning based Emotion Recognition System Using Speech Features and Transcriptions
- [6] Lech Margaret, Stolar Melissa, Best Christopher, Bolia Robert, "Real-Time Speech Emotion Recognition Using a Pre-trained Image Classification Network: Effects of Bandwidth Reduction and Comanding" in Frontiers in Computer Science, vol. 2, ISSN=2624-9898, DOI=10.3389/fcomp.2020.00014
- [7] Selvaraj, Mahalakshmi & Bhuvana, R. & Karthik, S Padmaja. (2016). Human speech emotion recognition. 8. 311-323.
- [8] S. K. Pandey, H. S. Shekhawat and S. R. M. Prasanna, "Deep Learning Techniques for Speech Emotion Recognition: A Review," 2019 29th International Conference Radioelektronika (RADIOELEKTRONIKA), 2019, pp. 1-6, doi: 10.1109/RADIOELEK.2019.8733432.
- [9] Fayek, Haytham & Lech, Margaret & Cavedon, L. (2015). Towards real-time Speech Emotion Recognition using deep neural networks. 10.1109/ICSPCS.2015.7391796.
- [10] Toronto emotional speech set (TESS) Database by EU Jin Lok (2019) <https://www.kaggle.com/ejlok1/toronto-emotional-speech-set-tess>

- [11] RAVDESS Emotional speech audio Database by Steven R. Livingstone (2018) <https://www.kaggle.com/uwrfkaggler/ravdess-emotional-speech-audio>
- [12] CREMA-D Database by TEU Jin Lok. (2019) <https://www.kaggle.com/ejlok1/cremad>
- [13] SAVEE Database by Tarun sunkaraneni. (2019) <https://www.kaggle.com/barelydedicated/savee-database>
- [14] Normalization  
<https://www.codecademy.com/articles/normalization#:~:text=Min%2Dmax%20normalization%3A%20Guarantees%20all,with%20the%20exact%20same%20scale>
- [15] DenseNet-161(2017) - <https://www.kaggle.com/pytorch/densenet161>
- [16] Resnet50 - <https://www.mathworks.com/help/deeplearning/ref/resnet50.html>
- [17] Audio Classification using Librosa and Pytorch by Hasith Sura. (2020) <https://medium.com/@hasithsura/audio-classification-d37a82d6715>
- [18] Performance Metrics: Confusion matrix, Precision, Recall, and F1 Score by Vaibhav Jaiswal(2020)-  
<https://towardsdatascience.com/performance-metrics-confusion-matrix-precision-recall-and-f1-score-a8fe076a2262>
- [19] Effects of spectrogram pre-processing for audio classification by Lahiru Nuwan Wijayasingha. (2019) <https://medium.com/using-cnn-to-classify-audio/effects-of-spectrogram-pre-processing-for-audio-classification-a551f3da5a46>
- [20] CNNs for Audio Classification by Papia Nandi. (2021) <https://towardsdatascience.com/cnns-for-audio-classification-6244954665ab>