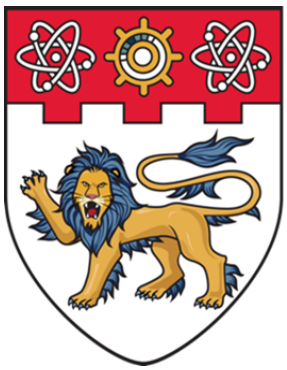


NANYANG TECHNOLOGICAL UNIVERSITY
SINGAPORE

CZ4042 Neural Networks and Deep Learning

Report for Project

2022-23 Semester 1

A Study on Speech Emotion Recognition using Neural Networks

Group Members:

Faizan (U2023863D)

Karanam Akshit (U2020311E)

Lumlertluksanachai Pongpakin (U2023344C)

1) Introduction

Speech Emotion Recognition (SER) is a fundamental task to predict the emotion label from speech data. The act of attempting to recognize human emotion and affective states from speech which is capitalised on the fact that voice often reflects underlying emotion through tone and pitch, is tough because emotions are subjective and annotating audio is challenging. Moreover in this rapidly advancing world where our daily tasks and needs are taken care of by Artificial Intelligence such as Amazon's Alexa or Google's Assistant, having physical or virtual service robots be able to understand human emotions better enables them to provide better service and accomplish tasks that range from caring for the elderly to assessing the effectiveness of your marketing campaign. We can find wide applications of speech emotion recognition in marketing, healthcare, customer satisfaction, gaming experience improvement, social media analysis, stress monitoring, and much more. [1][2][3]

We are utilising a hybrid of LSTM with attention and CNN model for the real-time analysis of emotions using a dataset from RAVDESS and TESS.

Problem Statement

Recent works mostly focus on using convolutional neural networks~(CNNs) to learn local attention maps on fixed-scale feature representation by viewing time-varied spectral features as images. However, rich emotional features at different scales and important global information are not able to be well captured due to the limits of existing CNNs for SER. In this Report, we tackle such problems by trying to implement different models and even concocting a hybrid model consisting of existing models to see if they fare better.

Objective

The objective of this project, given the problem statement, is to be able to accurately predict the emotions of a given word or sentence. Our team aims to develop and implement a model that can help to predict the emotion perceived from the text to the highest acceptable degree of accuracy.

Evaluation Criteria

Our team has decided to make use of cross entropy loss, categorical accuracy, given that the problem statement is a classification problem (classifying the speech into different sentiments), to evaluate the performance of the models designed and implemented.

Running Files

The code repository can be found using the following link : https://github.com/akshitkaranam/CZ4042-Project

2) Data Preparation

In making the data useful for applying our model architecture, implementation and the subsequent analysis, it has to be prepared for preliminary pre-processing and exploration. The given dataset have been used for this project :

1) RAVDESS : Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS) contains 1440 files: 60 trials per actor x 24 actors = 1440. The RAVDESS contains 24 professional actors (12 female, 12 male), vocalising two lexically-matched statements in a neutral North American accent. Speech emotions include neutral,calm, happy, sad, angry, fearful, surprise, and disgust expressions. Each expression is produced at two levels of emotional intensity (normal, strong), with an additional neutral expression. [4]

2) TESS : Toronto emotional speech set (TESS) Collection contains a set of 200 target words which were spoken in the carrier phrase "Say the word _____' by two actresses (aged 26 and 64 years). The recordings were made of the set portraying each of seven emotions (anger, disgust, fear, happiness, pleasant surprise, sadness, and neutral). There are 2800 stimuli in total. [5]

The following steps were implemented to prepare the given dataset:

- We decided to combine the two datasets so that we could have more data to train our model and make more accurate predictions
- RAVDESS had 8 emotions (neutral,calm, happy, sad, angry, fearful, surprise, and disgust) whereas TESS had only 7 emotions (anger, disgust, fear, happiness, pleasant surprise, sadness, and neutral). Nevertheless they were still merged.

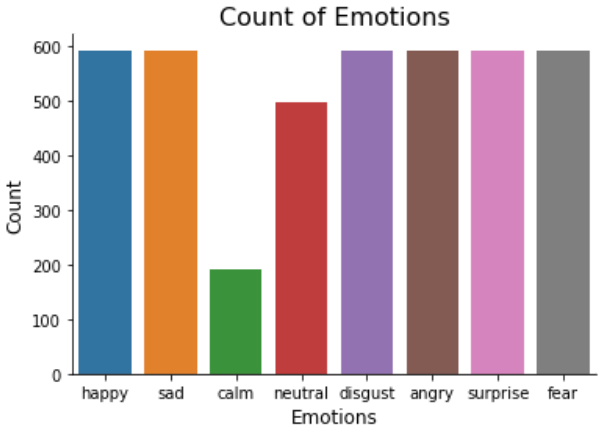
- TESS's pleasant surprise emotion was merged with RAVDESS's surprise emotion.
- The resulting dataset had 2 columns: The Emotion name and the path of the directory of the audio file stored in the folder
- The resulting dataset had 4240 data points with 2 columns.

The prepared dataset at the end could be found in the folder <FOLDER NAME>. With the data now merged and prepared for use, our team decided to explore and analyse the data. Understanding and examining the dataset aids in the construction of the model architecture and greatly helps in devising the appropriate strategies in employing the best strategies to solve the problem statement and fulfil the objective effectively.

3) Exploratory Data Analysis

Exploratory Data Analysis refers to the critical process of performing initial investigations on data so as to discover patterns,to spot anomalies,to test hypothesis and to check assumptions with the help of summary statistics and graphical representations.[6] EDA is for seeing what the data can tell us beyond the formal modelling and thereby contrasts traditional hypothesis testing [7]

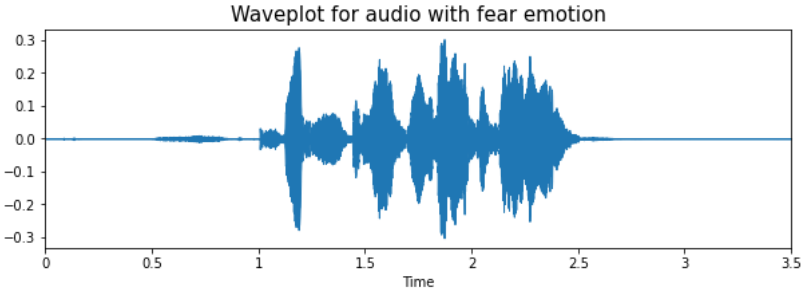
Total Emotion Spread:
The 8 emotions of RAVDESS and 7 emotions of TESS are shown below as well as their spread.



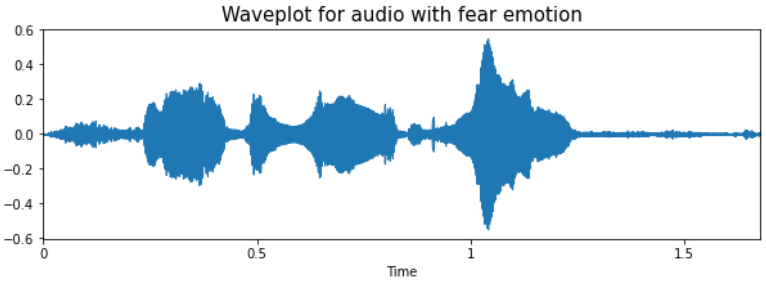
Since there is a repeat of the emotions happy,sad,neutral,disgust,angry,surprise and fear among the 2 original datasets, they are more than the emotion 'calm' which is found only in RAVDESS. Nevertheless, the distribution is quite uniform among the other emotions with each emotion constituting 592 datapoints. Neutral is the other exception with only 496 data points - RAVDESS has 96 datapoints for 'neutral' emotion and TESS has 400 data points for it.

Individual Emotion Analysis
Analysing each emotion separately for its waveplot and spectrogram yields different results. The waveplot plots the amplitude envelope of a waveform. The amplitude envelope refers to the changes in the amplitude of a sound over time, and is an influential property as it affects our perception of timbre. This is an important property of sound, because it is what allows us to effortlessly identify sounds, and uniquely distinguish them from other sounds. For example, we are easily able to identify a piano sound, and tell it apart from a trumpet sound [8]. The spectrogram is a wave-like graph which is used to represent measures like loudness, frequencies, and other signals that change over time. With the help of a spectrogram, these signals and measures are visually more understandable. A spectrogram is a two-dimensional graph in which the time component is represented mostly on the x-axis. [9]

Let's take the emotion 'fear' for an example analysis.
Waveplot for the RAVDESS data point:

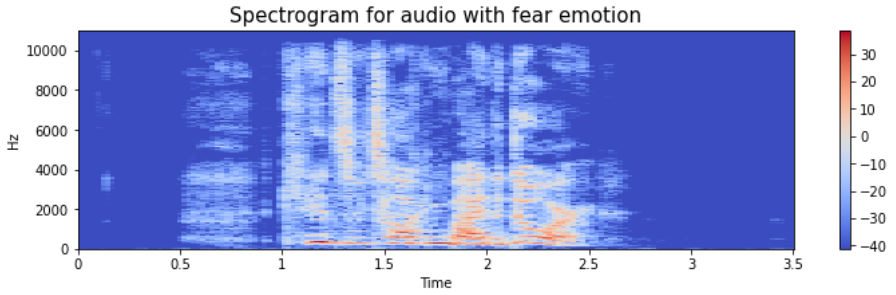


Waveplot for the TESS data point:

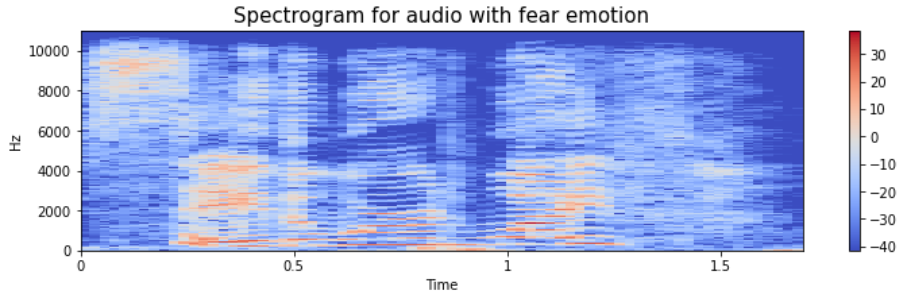


The RAVDESS data point clearly has empty stretches in front of and behind the main waveplot which could be the pause the actor makes before and after uttering the sentence in the emotion ‘fear’. Whereas for the TESS datapoint there are no such pauses and the audio clip is only of the actor uttering the sentences. This also explains the difference in the duration of the audio clip (RAVDESS - 0:03, TESS - 0:01). However, these pauses in the audio made no difference while training the model and thus we chose not to do anything about it and let it be.

Spectrogram for the RAVDESS data point:



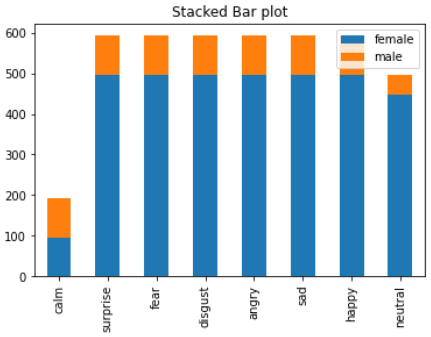
Spectrogram for the TESS data point:



For spectrograms, the colour adds a third dimension to the data point where it labels the amplitude (or energy or “loudness”) of a particular frequency at a particular time. The darker the colour, the lower the amplitudes and brighter colours up through red corresponding to progressively stronger (or louder) amplitudes[10]. You could see some red specks scattered through a sea of blue for the spectrograms for both the data points which indicate stronger amplitudes which in turn indicate high-energy which is seen as normal when expressing the emotion ‘fear’. However, as seen with the waveplot for RAVDESS, you could see an undisturbed block of blue which indicates no change in amplitude owing to the pause before and after the audioclip. As presented above, these pauses have no effect on the subsequent training of the model hence they are left as it is.

We do this for all the emotions and they display similar characteristics on account of the emotion they are affiliated to.

<u>Spread of Genders:</u>
RAVDESS has data points of audio from both the genders. However TESS utilises only female voices for the enunciations of the words. As such, there is certainly a disparity between the ratio of male and female voices.

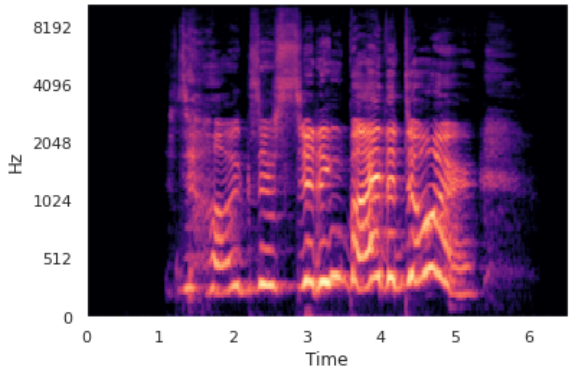


When it comes to the age group of the voice-actors, little is known from the RAVDESS dataset. However, TESS tells us that they utilised voice-actors who were 26 and 64 years old thus splitting their dataset into the categories of young and old. Since such variations will benefit when it comes to training of the model and the subsequent real-time analysis of speech emotion, they were not meddled with.

4) Feature Extraction

For feature extraction, we focused on four features of any audio file: Mel-spectrogram, Mel-frequency cepstral coefficients (**MFCCs**), zero crossing rate (**ZCR**) and root mean square (**RMS**).

Mel-Spectrogram is a spectrogram where the frequencies are converted to the mel scale which is a unit of pitch such that equal distances in pitch sound equally distant to the listener. Studies have shown that humans do not perceive frequencies on a linear scale. We are better at detecting differences in lower frequencies than higher frequencies. For example, we can easily tell the difference between 500 and 1000 Hz, but we will hardly be able to tell a difference between 10,000 and 10,500 Hz, even though the distance between the two pairs are the same. And thus a new scale was proposed and it is known as mel-scale. [11]




```
MEL spectrogram shape:  (128, 282)
```

Mel-frequency cepstral coefficients (**MFCCs**) are coefficients that collectively make up an MFC. Mel-frequency cepstrum (MFC) is a representation of the short-term power spectrum of a sound, based on a linear cosine transform of a log power spectrum on a nonlinear mel scale of frequency. MFCCs are derived from a type of cepstral representation of the audio clip (a nonlinear "spectrum-of-a-spectrum").

Zero Crossing Rate (**ZCR**) measures how many times the waveform crosses the zero axis.

Root Mean Square (**RMS**) is a commonly used normalization technique. Audio normalization is a fundamental audio processing technique that consists of applying a constant amount of gain to an audio in order to bring its amplitude to a target level.

5) Data Preprocessing

A python library, librosa, was used to work with the audio files. First, the raw signals were normalised, and then trimming was done to remove unnecessary data at the beginning and ending of the file. To ensure all signals are of the same length of 144,000, padding and further trimming was done at the end of each signal. Next, the dataset was split into parts - train,test and validation. To ensure that all three subsets had a similar proportion of each emotion type, the split was made on each emotion type, with a (80,10,10) split, and are then concatenated together.

Additive White Gaussian Noise (AWGN) was also added on the train set. AWGN tries to imitate the random processes that occur naturally. Data augmentation is a necessary step in many CNN models as it creates more variation and might allow the model to generalise better. After these steps of pre-processing, the features as mentioned in the previous sub-section.

Three datasets were prepared, one with a combination of all the 4 features (dataset 1), one containing just the mel spectrograms (dataset 2), and the other containing a combination of MFCCs, RMS and ZCR (dataset 3). Training on dataset 1 did not converge to a meaningful value, which might be due to the high inter-correlation between MFCC and the mel spectrogram. Training was conducted on datasets 2 and 3, however, the results on the training on dataset 3 were better than that of dataset 2. Hence, in this report, we focus on the training of dataset 3. For the results on dataset 2, please refer to the appendix.

6) Model Architecture

LSTM: Long Short-Term Memory (LSTM) networks are a type of recurrent neural network capable of learning order dependence in sequence prediction problems. This is a behaviour required in complex problem domains like machine translation, speech recognition, and more. LSTM has feedback connections, i.e., it is capable of processing the entire sequence of data, apart from single data points such as images. This finds application in speech recognition, machine translation, etc. LSTM is a special kind of RNN, which shows outstanding performance on a large variety of problems. [12]

LSTM with Attention: The encoder-decoder architecture suffers from the constraint that all input sequences are forced to be encoded to a fixed length internal vector. Attention is the idea of freeing the encoder-decoder architecture from the fixed-length internal representation. This is achieved by keeping the intermediate outputs from the encoder LSTM from each step of the input sequence and training the model to learn to pay selective attention to these inputs and relate them to items in the output sequence.[13]

CNN: CNN(Convolutional Neural Network) is a deep, feed-forward artificial neural network. The models are called "feed-forward" because information flows right through the model. There are no feedback connections in which outputs of the model are fed back into itself.[14]

Hybrid Model: We are using a concatenation of the LSTM with attention model and the CNN model.

In addition, the dimension of each 'X' subsets was expanded by 1 unit so that it can be inputted to a 2-dimensional CNN layer. However, as LSTMs take in only 2-dimensional input vectors, a squeeze operation is performed to remove this additional dimension before the training. Relu activation function was used wherever necessary.

LSTM

First, there are 2 LSTM layers of 64 units each which are then connected to a softmax output layer. Batch size of 32, learning rate of 0.001, RMSProp optimizer, Categorical Cross entropy loss function was used to train the model.

LSTM + Attention

First, there is a single LSTM layer of 64 units each which is connected to an attention layer. This is then connected to a softmax output layer. A custom attention layer was implemented in Keras using the Bahdanau Attention Algorithm[17]. Batch size of 256, learning rate of 0.001, RMSProp optimizer, Categorical Cross entropy loss function was used to train the model.

CNN

First there are four Conv2D layers with these sizes: [256,256,128,64], each connected to a MaxPool2D layers. A flatten layer is used to bring the shape to a 1 dimensional tensor. It is then connected to 4 fully connected layers with the sizes: [256,64,32,32]. This is then connected to a softmax output layer. Batch size of 64, learning rate of 0.01, SGD optimizer, Categorical Cross entropy loss function was used to train the model.

Hybrid

The hybrid model concatenates the embeddings of the models as described in 7.2 and 7.3. The architecture of the models in 7.2 and 7.3 remain largely unchanged. In 7.2, the softmax layer is replaced with a fully connected layer of size 256. In 7.3, after the flatten layer, there is only one fully connected layer of size 256. The concatenated embeddings are then connected to 4 fully connected layers with the sizes: [256,128,64,8]. This is then connected to a softmax output layer. Dropout layers were also implemented at each juncture. Refer to the diagram in the appendix, for an illustration of the hybrid model.

Hyperparameter tuning was done to get optimum values of learning rate, batch size, optimizer and dropout probability. The results of this are detailed in Section 8.

7) Hyperparameter Tuning

The results of each part of the tuning is shown below in the bar plots.

Batch Size : Batch Size is among the important hyperparameters in Machine Learning. It is the hyperparameter that defines the number of samples to work through before updating the internal model parameters. It can be one of the crucial steps to making sure your models hit peak performance. [15] A larger batch size generally means that the time to train the model will be significantly lower, but the accuracy might be affected. Hence, there is usually a trade off between the time taken and accuracy.
These batch sizes were used for the tuning: [32,64,128,256].

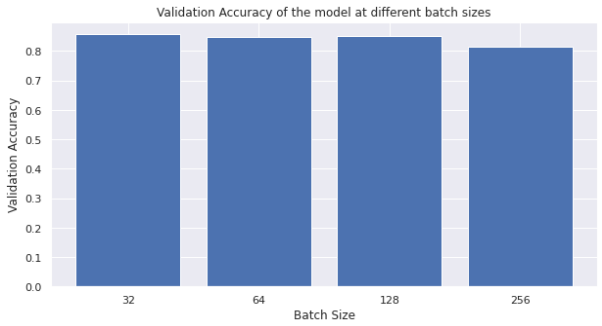
Each model was trained for 100 epochs. Batch size of 128 was chosen as it gave a similar validation accuracy as that of other small sizes while batch size of 256 showed a decrease in validation accuracy.

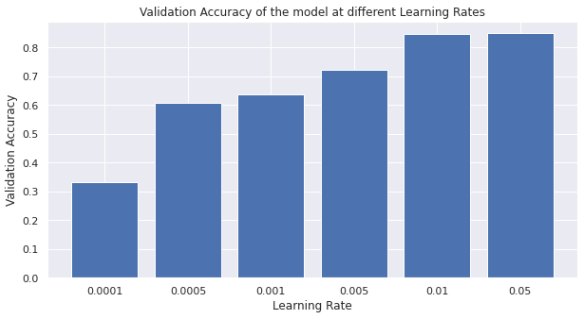
Learning Rate : The learning rate in an optimization algorithm determines the step size at each iteration while moving toward a minimum of a loss function. Metaphorically speaking, it represents the speed at which a machine learning model "learns".[16] If the learning rate is too small, the changes to the parameters are small as well, which would signify that more time is needed to reach the minimum point, hence more time is required to train the model. Time is an important issue as computational resources are usually limited. If the learning rate is too large, the local minima could be skipped and lead to fluctuating losses.
These learning rates were using for the tuning: [0.0001,0.0005,0.001,0.005,0.01,0.05]

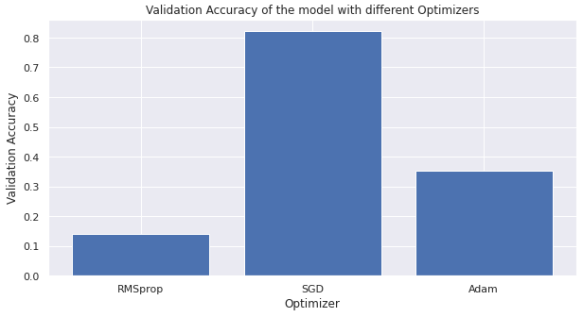
The model is trained for 100 epochs with the optimum batch size of 128 for each learning rate value. As seen from the results above, a higher learning rate does not diverge away from the minima, which suggests that it would be better to use a higher learning rate, as less computation is required to achieve similar results. We chose to select a learning rate of 0.01 as it brings about good convergence at a reasonably fast pace.

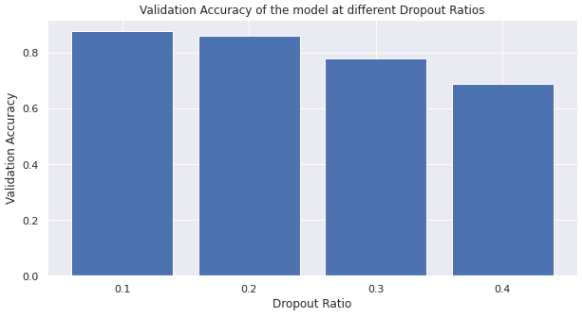
Optimizer: An optimizer is one of the two arguments required for training a neural network. We will be putting to test optimizers such as Adam, SGD, RMSprop.
Each model was trained for 100 epochs with a learning rate of 0.01 and batch size of 128. Referring to the diagram above, the SGD optimizer was best able to converge to a reasonable accuracy.

Dropout Ratio: Dropouts are used to reduce overfitting, where certain outputs of nodes are dropped out at random. These dropout ratios were used for tuning: [0.1,0.2,0.3,0.4]



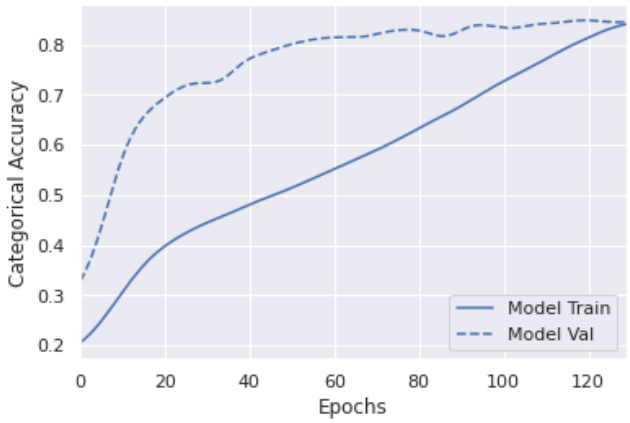


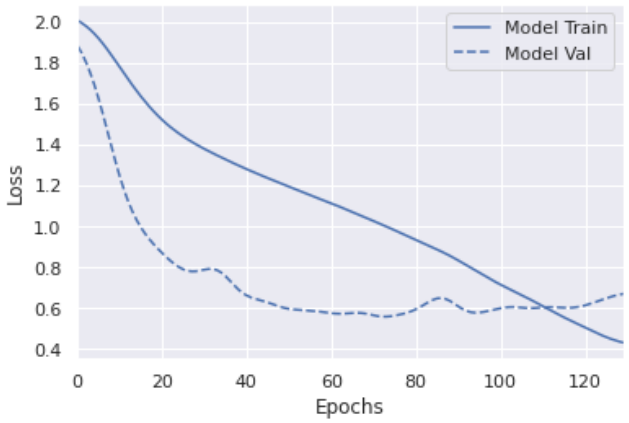




8) Model Testing

The final model was trained for 140 epochs, using tuned hyperparameters as mentioned in the previous section. An accuracy of 86.28% was achieved on the test set. The training and validation loss over the 140 epochs are shown in the graphs below.





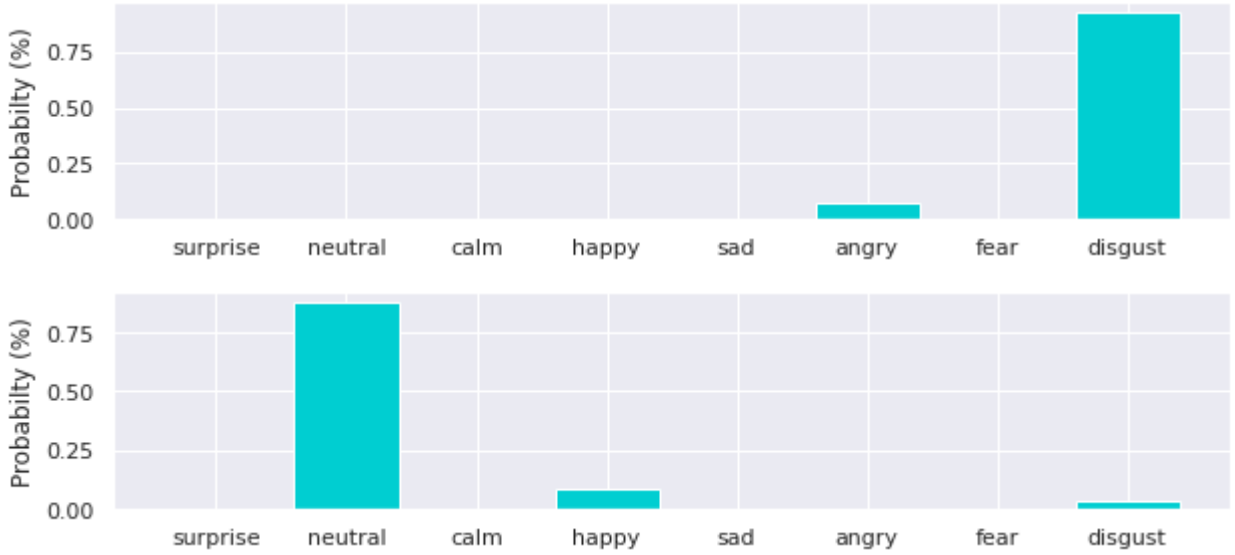
9) Real-Time Analysis

The model, an 86.28% accuracy LSTM with attention-CNN hybrid model based deep learning network, had learned how to classify the correct emotion expressed in a long-time sequence of speech features extracted from the audio signal.

This system will record audio input, create a temporary .wav file containing the audio signals recorded, preprocess it, and present the distribution of emotions found in speech. The sequence length chosen for the task is ~3 seconds. The process is cyclic and continues to accrue as long as there is continuous speech. The process can be ended manually by the user by selecting a button to end the recording session. An emotion bar plot which showcases the percentage of the emotions displayed in the audio snippet is displayed for every cycle. At the end of a session, a summary is presented, holding the mean values of all emotions recognized during the session.

Using a function called get_audio we get the audio input directly from the user in the browser. The UI utilises a simple button which prompts the user to start recording and stop the recording when they are done. Once the function gets an audio file it preprocesses it. The preprocessing of the audio files starts with the reduction of noise, the division of the said audio file corresponding to the same length of the input to our hybrid model and extracting of features (MFCC,RMS and ZCR) from the audio file. It then returns the concatenated array of the 3 extracted features back.

With the array, we then use our hybrid model to predict the emotions and display the compatible emotions that we derived initially from the EDA of the datasets. Thus we print the emotions displayed during the audio snippets and show the percentage of the probability of each emotion as a bar graph.



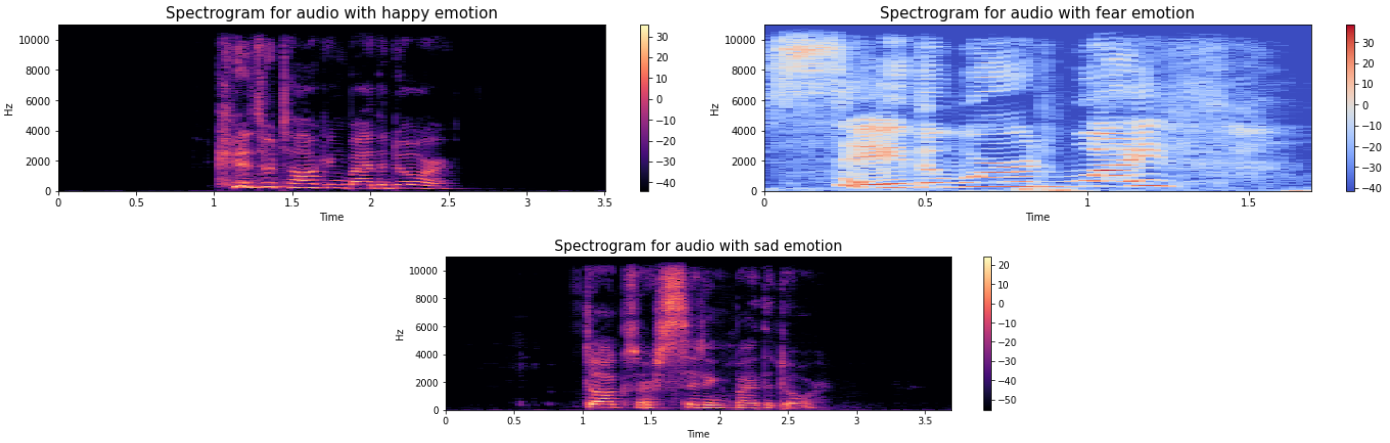
It also displays the emotion that was displayed the maximum percentage during the audio snippet which the model predicted.

```
max emotion: disgust
```

As such we have displayed the real-time emotion analysis fed by the user to the program effectively using our hybrid model trained previously.

10) Result Discussion

Having seen the real-time analysis done on the audio which was recorded by a user, let's discuss what influences the prediction of the emotion by the model. Having seen the EDA of the dataset (RAVDESS+TESS), the spectrogram helps us to understand that the audio clip(s) from the professional voice-actors does not mimic the day-to-day speech of a common person. Trying to mimic the emotions that was displayed by the voice actors for the dataset proves to be futile.



Let's take a look at the emotions : 'happy' (RAVDESS), 'sad' (RAVDESS), 'fear' (TESS) .
The 'happy' (RAVDESS)'s third dimension displays the amplitude of the audio through its colour. Not accounting for the black bars at either ends which are the pauses, we can see that the colours are quite dark and are spread throughout with mild variations to slightly deep orange. The same can be seen in the 'sad'(RAVDESS) emotion. Whereas for 'fear' (TESS), we can see specks of white and faint orange scattered throughout which signals that the amplitude of the audio is quite higher. Now, we know that the strength of the amplitude corresponds to the energy of the audio wave which generally conveys the volume of the audio clip. With the above graphs for reference, the spectrogram of 'happy'(RAVDESS) conveys that the energy of the audio clip is low which might not make sense for the common person when they are displaying the emotion of happiness. Though, the spectrogram for the 'sad' (RAVDESS) emotion makes sense, again sadness could be displayed with high energy/volume. Due to such nuances, the model which trains on such a dataset yields results that could often be not corresponding to the exact emotion displayed. Happiness conveyed with high energy could be accidentally referred to as fear or vice-versa. Our model does display the percentages of the probability of each emotion that it has predicted from the user's audio clip. However, the probability can be misled by such subtle variations.

11) Improvement

To conclude the project, we would like to elaborate on the areas that we could have improved on in order to achieve a better result.

1) **Need for more data -** From the previous paragraph where we explained the subtle difficulties the model might experience, it would have been a better option to train the model with a wider variety of datasets instead of confining it to using only two. Owing to our limited computational capabilities we had to proceed with using 2 datasets for model training.
2) **Using a multi-faceted model** - We could have explored the usages of transformers or other models better suited to analysis of emotions on top of the existing LSTM with attention and CNN hybrid that we used.

12) Conclusion

Overall, this was a very exciting project to work on. Seeing the analysis of the emotions in real-time was definitely worth the excruciating hours spent on working and refining the dataset and the subsequent models. This project has definitely taught us that neural networks and deep-learning can be used in a myriad of ways to achieve interesting results for real-world problems.

Acknowledgements

[1] https://blog.dataiku.com/speech-emotion-recognition-deep-learning
[2] https://data-flair.training/blogs/python-mini-project-speech-emotion-recognition/
[3] https://arxiv.org/abs/2204.05571#:~:text=Speech%20Emotion%20Recognition%20(SER)%20is,varied%20spectral%20features%20as%20images.
[4] https://www.kaggle.com/datasets/uwrfkaggler/ravdess-emotional-speech-audio
[5] https://tspace.library.utoronto.ca/handle/1807/24487
[6] https://towardsdatascience.com/exploratory-data-analysis-8fc1cb20fd15
[7] https://en.wikipedia.org/wiki/Exploratory_data_analysis
[8] https://maplelab.net/overview/amplitude-envelope/
[9] https://java2blog.com/spectrogram-in-python/#:~:text=A%20spectrogram%20is%20a%20twoin%20Python%20using%20different%20dictionaries.
[10] https://pnsn.org/spectrograms/what-is-a-spectrogram#:~:text=A%20spectrogram%20is%20a%20visual,energy%20levels%20vary%20over%20time.
[11] https://medium.com/analytics-vidhya/understanding-the-mel-spectrogram-fca2afa2ce53
[12] https://intellipaat.com/blog/what-is-lstm/
[13] https://machinelearningmastery.com/attention-long-short-term-memory-recurrent-neural-networks/
[14] https://www.datacamp.com/tutorial/convolutional-neural-networks-python
[15] https://medium.com/geekculture/how-does-batch-size-impact-your-model-learning-2dd34d9fb1fa
[16] https://en.wikipedia.org/wiki/Learning_rate#:~:text=In%20machine%20learning%20and%20statistics,minimum%20of%20a%20loss%20function.
[17] https://arxiv.org/abs/1508.04025