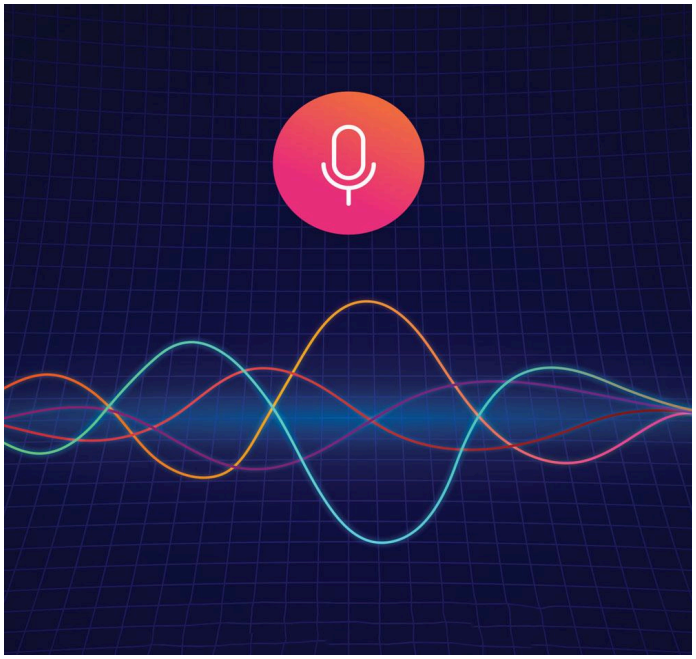
Speech Audio Emotion Recognition:

Classifying Audio Sound Clips into Emotion Categories Using Convolutional Recurrent Neural Networks



Bella Davies, Roy Chen

DS340 Spring 2024 - Prof. Gold

29 April 2024

Table Of Contents

Introduction

This project aims to recognize and classify emotions from speech audio by collecting speech audio samples to train a neural network. The goal is to determine the emotional state of any given speaker for any given sentence spoken from the emotions of anger, calm, disgust, fear, joy, sadness, surprise, and neutral. Exploring speech emotion recognition to develop a successful model that can recognize emotions expressed in spoken language is important as it can be used in many different applications and integrated into many technologies we use today. For example, such a model could be used to identify and display the overall emotion of voicemails and voice messages. Another application of this model could be to include a version of movie subtitles for people with hearing impairments, showing the emotion of the character speaking on screen. These implementations of the model would provide insight into spoken language without a human needing to listen to and analyze the emotions expressed. Furthermore, speech emotion recognition models can assist in the development and improvement of virtual mental health assistants, as well as call center analytics in general. Creating reliable speech emotion recognition models serve to enhance human-computer interaction, and would allow systems to adapt their responses based on the emotional tone of the users to foster more insightful and engaging interactions.

The audio files used to train the model were obtained from the Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS) and the Surrey Audio-Visual Expressed Emotion (SAVEE) dataset. RAVDESS contains 1440 audio files from 24 actors, 12 of which are male and 12 female. The actors in this dataset are speaking two different sentences in a North American accent with the emotions of anger, calm, disgust, fear, joy, sadness, surprise, and neutral. SAVEE contains 480 audio files from four native English male speakers, aged 27 to 31,

speaking 15 different sentences with all the same emotions contained in RAVDESS except calm. These datasets were then combined to provide more variation in the actors and sound clips. Together the two datasets contain 1920 audio files from 28 different actors speaking 17 different sentences, with 252 audio files labeled with emotions of happy, angry, sad, fear, disgust, and surprise, 216 audio files labeled 'neutral', and 192 labeled 'calm'. The first five rows of the combined dataset are shown in *Appendix 1*, and distribution of emotions in the combined dataset can be seen in *Appendix 2*.

Methodology

Waveplot and Spectogram

Before preparing the combined dataset for modeling, the waveplots and spectograms are examined for the 8 different categories of emotions. A waveplot shows the amplitude of an audio signal over time where time is on the x-axis and amplitude is on the y-axis. On the other hand, a spectogram shows the frequency of an audio signal over time, where time is on the x-axis, frequency on the y-axis, and the color representing the magnitude of the frequency. To visualize how the waveplot and spectogram vary for different emotions in the dataset used for this project, the waveplot and spectogram are generated on the first observation for each emotion in the dataset. The emotions that show the highest variance visually in the audio measurement data for the first sentence spoken in the dataset were happy and sad. The waveplots and spectograms for the emotions of happy and sad are shown in *Appendix 3* and *Appendix 4*. The difference between the waveplots and spectogram between the two emotions indicate that audio files of happy speech generally have a higher amplitude and frequency than audio files of sad speech.

Data Augmentation

After combining the datasets, there are 1920 total audio files. While this seems to be plenty, to ensure that the model has enough training data as well as enough variation in the audio clips, data augmentation is conducted to create variation in the data and increase the size of the dataset before extracting the features to train the model. The data preparation stages and the baseline models for this project are based on the source code from this Speech Emotion Recognition Kaggle notebook. Using librosa.load(path), the audio files are loaded into an array containing audio measurements and the sampling rate which represents the number of audio measurements captured per second.

While the source code performs the augmentations of adding noise as well as combining stretching and pitching, this project instead tries separating these for three separate augmentations instead of two on the loaded audio measurement data. The three augmentations used for this project are adding noise, stretching, and changing the pitch. The effects of these augmentations are displayed on the waveplots in *Appendix 5-8.* Random noise is added to each audio file by computing a noise amplitude that is a fraction of the maximum amplitude of the sound clip, then multiplying this noise amplitude by a randomly generated number between 0 and 1, and adding the result to the data. Time stretching is also performed on the audio data, which stretches the audio using the rate parameter, set to a default value of 0.8. This rate means that the audio is stretched to 80% of its original duration while preserving its pitch. The final augmentation performed is pitch shifting, where the pitch is shifted up by 7 semitones using the n_steps parameter. After performing these three data augmentations there are a total of four different versions of audio measurement data for each audio file originally present in the

combined dataset, making a total of 7680 observations. These observations of original and augmented sound clips make up the total dataset from which features will be extracted.

Feature Extraction

Next, the following features were extracted from the total dataset: zero crossing rate, chroma short-time fourier transform, mel-frequency ceptral coefficients, root mean square value, and mel spectogram. The zero crossing rate, or ZCR, measures the number of times the audio signal passes through zero amplitude in a specified time frame. It helps identify characteristics about the rhythmic content, sudden changes in the sound, like percussive sounds, and distinguishing between voice and silence. The chroma short-time fourier transform, or chroma STFT, computes a chromatogram from an audio signal which represents the intensity of various pitches, and is useful in music analysis for capturing the tone of the sound. Mel-frequency cepstral coefficients, or MFCCs, summarize the power, texture, and quality of the sound, and come from the ceptral representation of the audio clip, which mimics the human auditory system and the human ear's response. The root mean square (RMS) value measures the audio signal's amplitude, or energy level, and helps to assess the volume or strength of a sound signal as well as distinguishing between voiced and unvoiced sounds. The mel spectogram maps the frequencies of the audio signal onto the mel scale, which is a scale of pitches that are equal in distance. The resulting spectogram shows how the energy is distributed across the mel-scaled frequency bands over time, and it helps model music and speech textures.

The feature extraction process effectively transforms the sound clips into numerical representations of the audio signals, such that the values of the measurements can be inputed to a machine learning model. The dataset of features has 7680 rows, 161 columns, and labels for each observation, and can be seen in *Appendix 9*. Each feature or column plays a different role in the

speech audio analysis, enhancing the model's ability to understand and classify audio data into emotions in ways aligned with human auditory perception.

Data Preparation

After extracting features to transform the audio clips into numerical measurements of the audio signal, the target Y variable, or emotion, is one-hot encoded since this is a multiclass classification problem. Next, the dataset must be split into training and test sets. Splitting the dataset into training and testing sets is a fundamental practice in machine learning, so that the test set can be used to evaluate the performance of the model. The training dataset is the larger portion of the data (80%) that is used to train the model. The test set is the smaller portion used to test the model's performance on new data (20%). The purpose of this split is to prevent overfitting, where a model might perform exceptionally well on the training data but poorly on test data. After splitting, X_train has 5760 rows and 161 columns and X_test has 1920 rows and 161 columns. Y_train and y_test have the same number of respective rows as X_train and X_test, but with 8 columns for the labels of the different classes of emotions.

Sometimes in machine learning there is an additional validation set that is split off and used to evaluate the model during the process of fine tuning the model, so that the test dataset used to evaluate the final model has not been seen by the model before during training. In the context of this project, the "testing" dataset will act as the validation set that will be used to evaluate the model as improvements are made and parameters are changed. The original dataset is not split into training, validation, and test sets since there are originally only 2,000 audio clips, and including the further split with all three would reduce the training data further to 60%, and this project aims to ensure the model will have enough training data so it uses 80% to train and the other 20% to fine tune the model. After fine tuning the model on the validation set, the model

will be tested on new, different sound clips that do not come from either of the original two datasets. The reasoning behind this is that it is more useful to evaluate the final model on sound clips that come from a different context than the original datasets which are voice actors in a controlled environment and speaking only 17 total different statements with assigned emotions to act. This type of speech audio is not necessarily a real-world example of speech dialogue. Therefore this project will test whether the model can perform well on unseen data and speech from any speaker, statement, and context, in order to correctly identify emotions from speech audio in practical applications. This will ensure that the model is not only well-trained but also robust and generalizable to new, real-world data.

After splitting the data into training and test sets, the data must be normalized to prevent the model from valuing certain features over others during the learning process due to their larger scale. Using sklearn's standard scaler, all features in X_train and X_test are scaled to a mean of 0 and a standard deviation of 1. Next, the data must be resized by adjusting the dimensions to fit the requirements of a model where input data features must meet specific dimensional criteria. Since the models which this project will explore, like CNNs (Convolutional Neural Networks), require a fixed input size to efficiently process the data in batches, resizing the data is necessary in order to ensure data compatibility with the model, facilitate better performance, and avoid errors during model training. In the case of a 1-Dimensional CNN, the data should have the following dimensions: (# samples, # observations in each sample, # of features in each observation). Therefore the data is expanded to add a dimension on the third axis of X_train and X_test, with the resulting shapes of (5760,161,1) and (1920,161,1), making the data compatible for modeling.

Modeling

This project will explore two different types of machine learning models, Convolutional Neural Networks (CNNs) and Convolutional Recurrent Neural Networks (CRNNs). For both models there are variations added to the model architecture to evaluate the performance and the effect of changing model parameters, as well as attempts to improve the accuracies of the models for further reliability. To evaluate each model, an accuracy and loss plot are generated to show the training and testing performance throughout the epochs of the model's learning. Additionally, predictions are generated for X_test, as well as a confusion matrix displaying the predicted labels and the actual labels. Lastly, a classification report is generated to show the precision, recall, and F1 scores for each of the 8 emotion classes as well as their overall averages.

1. CNN with 1 Convolutional Layer

The first model explored is a Convolutional Neural Network with a 1 dimensional convolutional layer with 128 filters of size 5. The input shape is (X_train.shape[1], 1), where X_train.shape[1] is the number of observations (161) and 1 represents the number of features in each observation. After the convolutional layer, the Rectified Linear Unit (ReLU) activation function is added to the model, introducing non-linearity and helping the model to learn complex patterns. Next, a dropout layer is added with a dropout rate of 0.2, which helps to prevent overfitting by dropping a random sample of the input data during training. A flatten layer then flattens the data into a 1-dimensional vector, preparing the data to be inputted to a fully connected dense layer with 8 units for each prediction class. The dense layer helps the model learn high-level features and has a softmax activation function which converts the output into probabilities, making it suitable for multi-class classification problems such as this one. Finally,

the model is compiled with the Adam optimizer and categorical cross-entropy loss, which is suitable for multi-class classification. This model gave a test accuracy score and f1-score of 62.76%, and will act as the baseline model for this project to improve upon.

2. CNN with 4 Convolutional Layers

The next model explored is a CNN with 4 convolutional layers. In this model, one-dimensional max pooling layers are applied after each convolutional layer to downsample or reduce the dimensions of the features. Flatten layers and dense layers with ReLU and softmax activation functions are also used here after the convolutional layers. There are two dropout layers with dropout rates of 0.2 and 0.3 are applied after specific layers to help prevent overfitting. The model is also compiled with the Adam optimizer and categorical cross-entropy cross entropy function. The model is trained on the training data with a learning rate reduction, which adjusts and reduces the learning rate as the loss plateaus. This second baseline CNN model resulted in a higher test accuracy of 65.63%, improving from the first baseline model.

3. CRNN with 3 Convolutional Layers and 4 Recurrent Layers

To further improve the performance, the next model attempted is a Convolutional Recurrent Neural Network (CRNN). A CRNN combines both convolutional and recurrent layers to capture more patterns and temporal features like energy and frequency in the audio data. To obtain a baseline CRNN model, generative AI was used with the prompt of generating a simple CRNN for speech audio emotion recognition. The code was then debugged and altered to The one dimensional convolutional layers use 64 convolutional filters of size 3 with ReLU activation. Max pooling is added again here to reduce the dimensions of the features. Before adding recurrent layers, the output of the convolutional layers need to be prepared for input to the

recurrent layers. To do this, a flatten layer is used to convert the data to a one-dimensional array. The flatten layer is used inside a Time Distributed wrapper, which applies a layer to each time step of the input. Together, the combination of TimeDistributed(Flatten()) converts the 2-dimensional feature map into a one-dimensional vector ready for input to recurrent layers. The recurrent layers use LSTM units with 64 units per layer. Stacking LSTM layers further helps the model to capture temporal patterns in the sequential data. Dropout layers with a dropout rate of 0.2 are added to help prevent overfitting. A dense layer with softmax activation is used to produce class probabilities, and the model is compiled with the Adam optimizer and categorical cross-entropy loss function. The model is trained and a learning rate reduction callback adjusts and reduces the learning rate as the training loss plateaus. The performance of the CRNN with 3 convolutional layers and 4 recurrent layers shows further improvement from the CNN with 4 convolutional layers. The test accuracy and f1-score of the CRNN is 67.29%. As this is a jump up from our initial two baseline models, this project will attempt to make improvements to this CRNN model by varying the model architecture.

4. Variations to the CRNN Model

To attempt to improve the model, the following variations were made to the CRNN model: reducing dropout rate, changing the filter size of the convolutional filters, adding an additional convolutional layer, using multiple activation functions, and using gated recurrent units. The results of these variations are summarized in *Appendix 10.* When reducing the dropout rate, a rate of 0.1 was tried instead of 0.2. The reasoning behind this is that there is limited training data, and using a smaller dropout would allow the model to learn from more of the training data during the learning process. This resulted in a test accuracy and f1 score of 67.35%.

As this is only a 0.06% improvement and not significant it was not used in other following variations since 0.2 is a more standard dropout rate. Next, the convolutional filter size was changed from 3 to 5, to attempt to capture more complex patterns in the data. This resulted in an improvement to the model's performance with a test accuracy and f1 score of 70.1%. Since this is an improvement, a filter size of 5 will be kept in other following variations. Next, an additional convolutional layer is added to the model to see if this would help the model capture even more complex patterns. Adding a layer resulted in a higher jump in the test accuracy and f1 scores, at 74.43%, and the additional layer will be kept in further variations. Another variation to the CRNN that was tried was using different activation functions in the convolutional layers to allow for more diverse representations of the input data and make the model more flexible, robust, and generalizable. The second layer was changed to the exponential linear unit (ELU) activation function, which allows for negative values as compared to ReLU and can be more robust to noisy input data, and the third layer was changed to the hyperbolic tangent (tanh) activation function, which also allows for negative values. The effect of trying different activation functions resulted in a further improved test accuracy and f1 score of 76.2%, so the different activation functions will be kept in the model for the final variation. Lastly, this project explores the effect of using gated recurrent units (GRUs) as opposed to LSTM units, as they are simpler and more computationally efficient. Moreover, they can be more effective for smaller datasets such as the one used in this project. The effect of using GRUs instead of LSTMs resulted in the highest test accuracy and f1 scores across all models, at 76.5%, so this is the final selected model as it is the best performing. The final model summary can be seen in *Appendix 11*, the loss and accuracy plots are shown in *Appendix 12*, and the confusion matrix is displayed in *Appendix 13*.

Results

To evaluate the model, this project uses sound clips from the trailer Pixar's 2015 movie, "Inside Out". Since the model architecture was changed and improved according to the same test dataset, the test dataset from the original data acted as a validation set, as discussed earlier in the Data Peparation section of this paper. To provide new, unseen audio data to evaluate the model in a real-world context, three audio clips for each of the 8 emotion classes are extracted from the trailer for the movie. Since the characters speaking in the movie are acting as specific emotions, the characters themselves are the emotion labels for their own dialogue. The three emotions in this classification problem that are not present in the movie, neutral, calm, and surprise, are included as well using a human auditory evaluation of the dialogue to label these emotions. Three of each emotion are used to provide a few examples of each to allow for more information gathered from the test dataset during feature extraction. These 24 audio files undergo the same data preparation process as the original training data. Three data augmentations are performed, adding noise, time stretching, and changing the pitch. After augmenting there are 4 versions of each audio file including the original version, resulting in 96 audio files. The same features of zero crossing rate, chroma short-time fourier transform, mel-frequency cepstral coefficients, root mean square value, and mel spectogram are extracted from the 96 audio files. The labels are one hot encoded, then the features are scaled to a mean of 0 and standard deviation of 1 and expanded to add a third dimension to make the data compatible for the CRNN model.

The performance of the "Inside Out" test audio data on the model is evaluated in the same way by generating predictions, a confusion matrix, and calculating the test accuracy score. The test accuracy of the test data was around 25%, and the confusion matrix is shown in *Appendix 14.* Examining the confusion matrix, it can be seen that the model most accurately

predicted the emotions of anger, disgust, and sadness. This makes sense since these emotions are typically stronger in the ways they are expressed, as well as being more distinguishable to both humans and machines. Since the model seems to perform better with these stronger emotions, the accuracy of strong emotions was also calculated at 42%, while the accuracy of all other emotions was only around 15%. Examining the confusion matrix further, it can be seen that the model sometimes mixed up fear/anger, as well as happy/surprise, and sad/neutral. These emotions can often sound similar in the ways they are expressed, considering volume, frequency, and tone. If a human might have trouble distinguishing between specific emotions, it is reasonable that a machine would also have trouble. Another observation made from the confusion matrix of the test data is that there were very few predictions for calm, neutral, and surprise. This is likely due to the fact that there are no characters for these emotions in the movie (the movie has Joy, Sadness, Anger, Disgust, and Fear), so the sound clips for the other three classes were less clear on the specific emotion and were labeled based on individual human perception.

Furthermore, since emotions are subjective, human natural language and dialogue may not always have exactly one clear emotion label that is agreed upon by everyone. However, the training data did have clear labeled emotions because the audio clips came from speech actors with assigned statements and emotions in a controlled environment that is not representative of natural language speech. Therefore accuracy may not be the best evaluation metric since emotions can not always be measured numerically or boiled down to one clearly labeled class. These reasons provide a few possible explanations for why the model performed well with the clearly labeled training data from a controlled environment, but did not perform as well with real world, unlabeled natural language speech dialogue.

Conclusion

In conclusion, the final CRNN model had a 76% validation accuracy but 25% test accuracy. From the considerations discussed above, there are several possible reasons for the low test accuracy, including stronger and milder emotions, self-labeled test data, and real-world complexity of the expression and perception of human emotions. Throughout the process of building this project, there were several learning takeaways, including how to perform data augmentation and its benefits for providing variation in the training data, feature extraction for audio data, creating a convolutional neural network and convolutional recurrent neural network, and changing the model architecture and model parameters to improve overall performance.

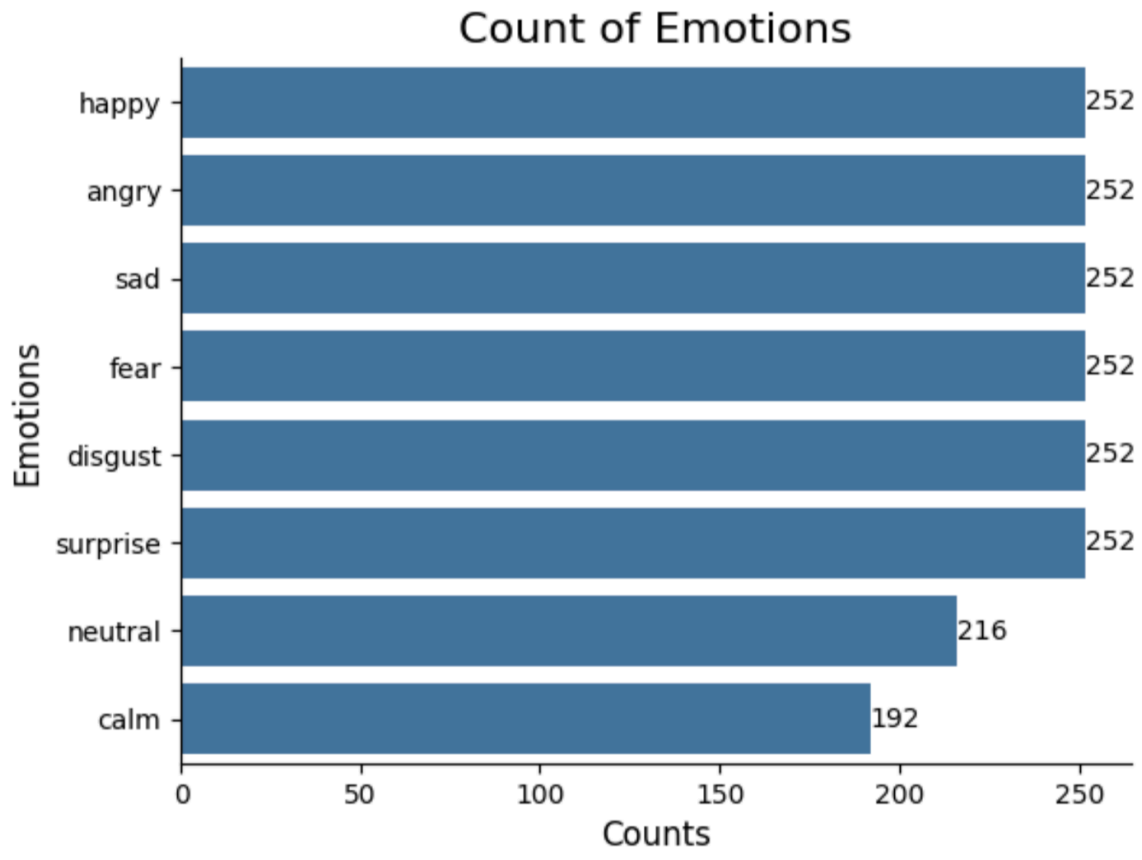
There are a few ways in which this project could be further improved. Using more training data would provide more for the model to learn from, as this project has somewhat limited training data with only around 200 audio clips per emotion class. Furthermore, the model could be tested on more varied test data from the actual movie, rather than a movie trailer, to provide more natural dialogue without background music. In terms of achieving a higher accuracy in the future, further data augmentations could be performed on the original data, or additional features could be extracted. One could also experiment with transformer architectures like BERT, including attention mechanisms, and tring ensemble methods to further boost learning. Overall, this project provided experience for working with audio data as a new form of multimodal data, as opposed to working with tabular data which has been most commonly done in past classes, and was overall a huge learning experience creating something new.

Appendix

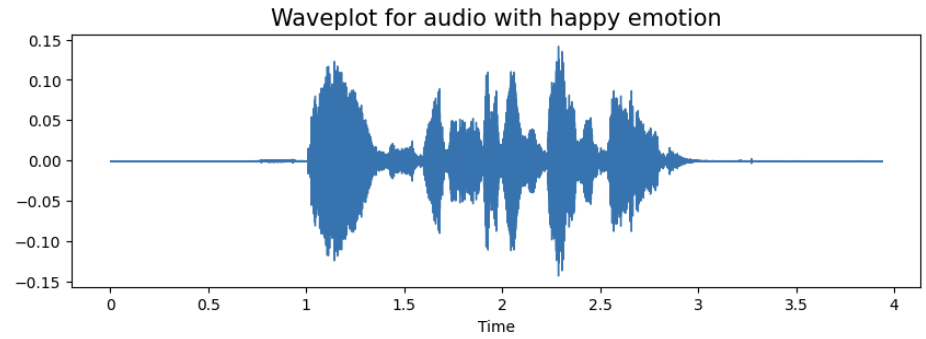
Appendix 1: Combined Dataset

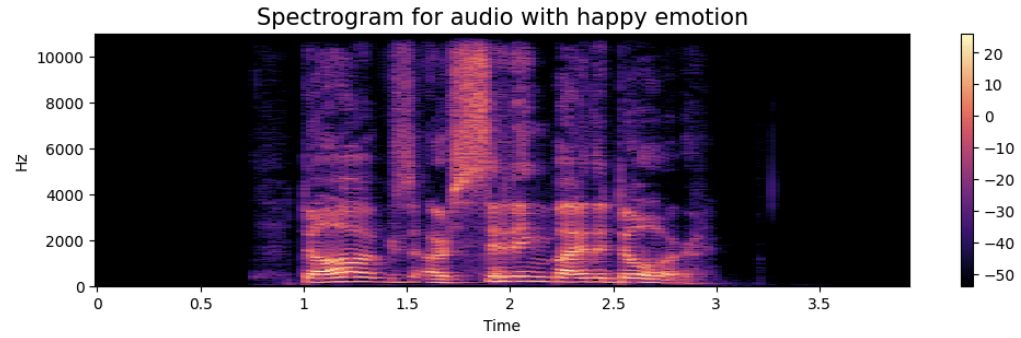
index	Emotions	Path
0	neutral	/content/drive/MyDrive/DS340Project/RAVDESS/Actor_01/03-01-01-01-01-02-01.wav
1	happy	/content/drive/MyDrive/DS340Project/RAVDESS/Actor_01/03-01-03-01-02-02-01.wav
2	calm	/content/drive/MyDrive/DS340Project/RAVDESS/Actor_01/03-01-02-01-01-01-01.wav
3	angry	/content/drive/MyDrive/DS340Project/RAVDESS/Actor_01/03-01-05-02-02-02-01.wav
4	calm	/content/drive/MyDrive/DS340Project/RAVDESS/Actor_01/03-01-02-01-01-02-01.wav

Appendix 2: Distribution of Emotions in Combined Dataset

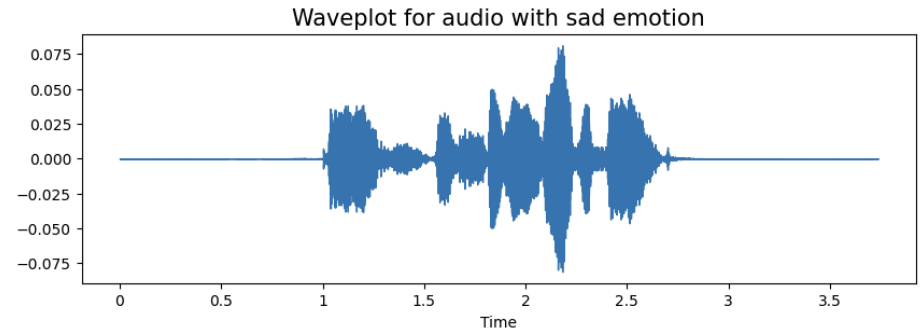


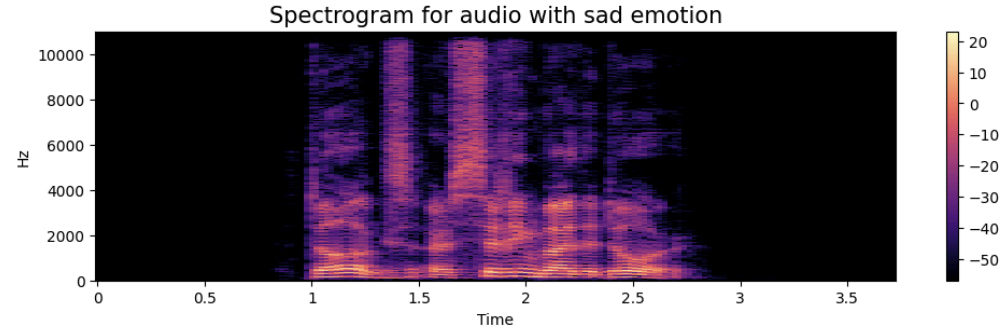
Appendix 3: Waveplot and Spectogram of Happy



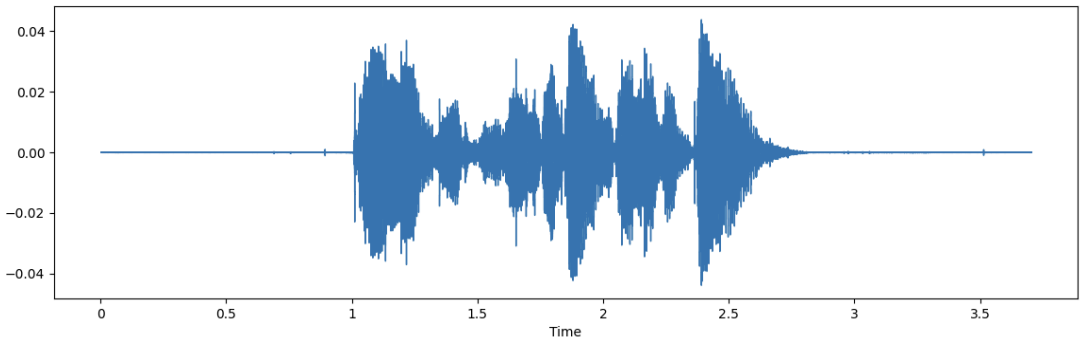


Appendix 4: Waveplot and Spectogram of Sad

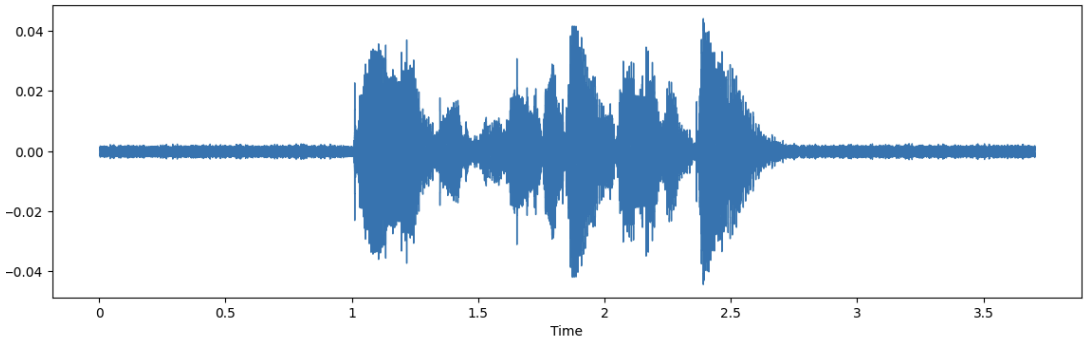




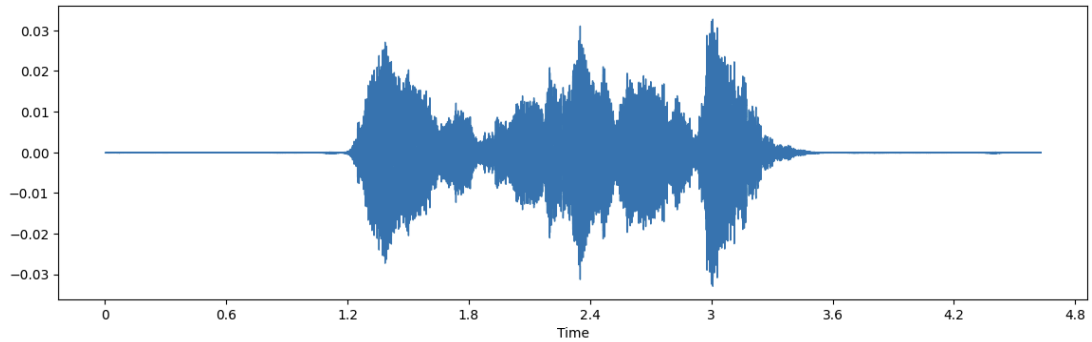
Appendix 5: Sample Audio Waveplot



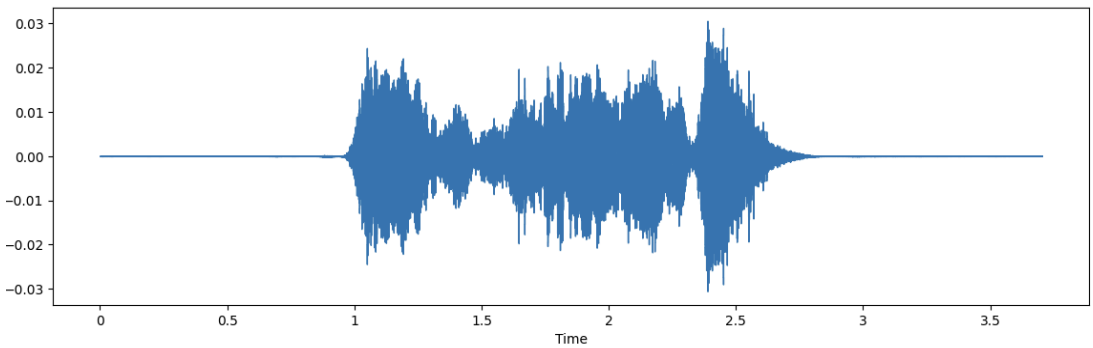
Appendix 6: Noise Injection Waveplot



Appendix 7: Time Stretching Waveplot



Appendix 8: Changing Pitch Waveplot



Appendix 9: Dataset After Data Augmentation and Feature Extraction

	0	1	2	3	4	5	6	7	8	9	...	153	154	155	156	157	158	159	160	161	labels
0	0.293566	0.673896	0.722096	0.723508	0.682302	0.680533	0.675352	0.628977	0.679179	0.707283	...	6.984505e-06	7.034950e-06	6.654923e-06	6.979548e-06	1.214236e-05	9.640183e-06	1.096403e-05	5.543237e-06	4.254084e-07	neutral
1	0.318283	0.767527	0.817022	0.799512	0.785292	0.794199	0.746606	0.651910	0.686053	0.726052	...	9.014994e-05	8.813220e-05	8.358031e-05	8.959403e-05	9.375046e-05	9.444011e-05	9.445891e-05	8.785902e-05	8.004236e-05	neutral
2	0.168298	0.694371	0.727215	0.688030	0.664346	0.675902	0.634768	0.637774	0.691974	0.722612	...	2.797620e-06	3.027207e-06	3.510560e-06	3.349115e-06	5.726472e-06	4.637588e-06	5.238408e-06	2.391436e-06	1.455581e-07	neutral
3	0.192044	0.675267	0.634700	0.681160	0.702156	0.731349	0.665670	0.640723	0.608663	0.655598	...	1.595107e-05	1.223287e-05	7.110164e-06	7.660465e-06	3.098759e-06	1.970621e-06	2.038793e-06	7.761399e-07	5.283976e-08	neutral
4	0.259879	0.621764	0.669827	0.692539	0.652645	0.571166	0.595168	0.586239	0.608535	0.637656	...	6.779312e-05	4.632183e-05	5.351165e-05	5.744937e-05	5.894652e-05	5.425128e-05	5.536501e-05	2.759516e-05	2.388180e-06	happy

Appendix 10: Model Results Table

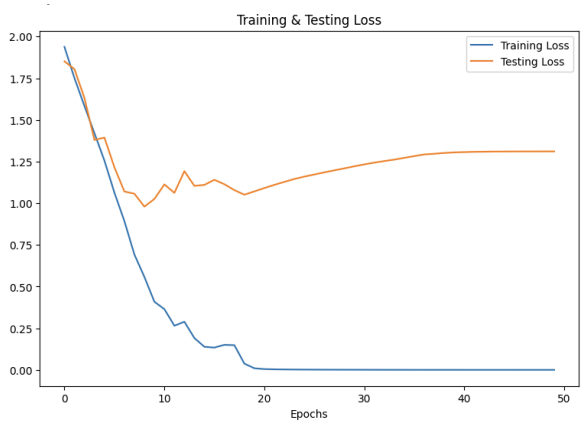
Model	Test Accuracy
CNN with 1 Hidden Layer	62.76%
CNN with 4 Hidden Layers	65.63%
CRNN	67.29%
● Reducing dropout	67.34%
● Changing layer size	70.10%
● Adding a layer	74.43%
● Activation functions	76.20%
● Gated Recurrent Units	76.51%

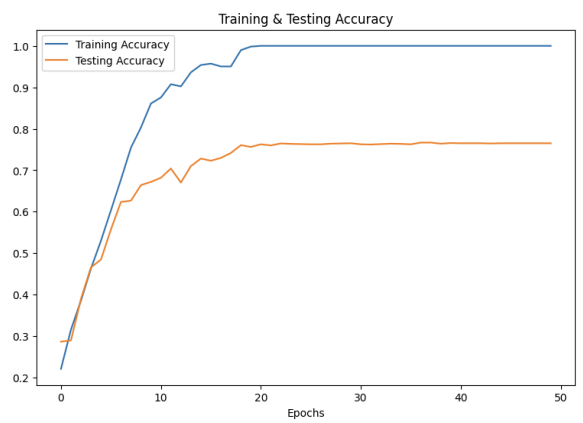
Appendix 11: Final Model Summary

Layer (type)	Output Shape	Param #
conv1d (Conv1D)	(None, 161, 64)	384
max_pooling1d (MaxPooling1D)	(None, 80, 64)	0
conv1d_1 (Conv1D)	(None, 80, 128)	24704
max_pooling1d_1 (MaxPooling1D)	(None, 40, 128)	0
conv1d_2 (Conv1D)	(None, 40, 256)	98560
max_pooling1d_2 (MaxPooling1D)	(None, 20, 256)	0
conv1d_3 (Conv1D)	(None, 20, 512)	393728
max_pooling1d_3 (MaxPooling1D)	(None, 10, 512)	0
time_distributed (TimeDistributed)	(None, 10, 512)	0
gru (GRU)	(None, 10, 64)	110976
dropout (Dropout)	(None, 10, 64)	0
gru_1 (GRU)	(None, 10, 64)	24960
dropout_1 (Dropout)	(None, 10, 64)	0
gru_2 (GRU)	(None, 10, 64)	24960
dropout_2 (Dropout)	(None, 10, 64)	0
gru_3 (GRU)	(None, 64)	24960
dropout_3 (Dropout)	(None, 64)	0
dense (Dense)	(None, 8)	520

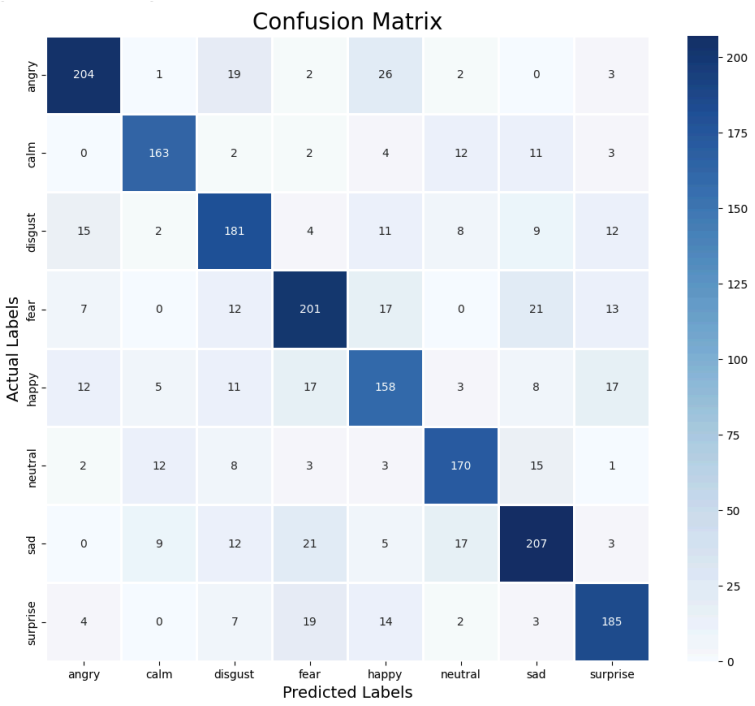
Total params: 703752 (2.68 MB)
Trainable params: 703752 (2.68 MB)
Non-trainable params: 0 (0.00 Byte)

Appendix 12: Final Model Loss and Accuracy Plots





Appendix 13: Final Model Confusion Matrix



Appendix 14: "Inside Out" Testing Confusion Matrix