Speech Emotion Recognition

Using MLP Classifier

By:

Divesh Gadhvi

Parth Borkar

Tanay Dangaich

Koutilya Pande

Abstract

Effective communication is essential for social connection, and emotions play a vital role in this process. As humans, we primarily use speech and language to communicate with each other, and the way we express our emotions through speech can vary depending on the intensity and type of emotion we are experiencing. Recognizing emotions in speech is a challenging but essential task in human-machine interaction. Our project aims to create an intelligent system that can recognize emotions in speech using a convolutional neural network (MLP classifier) and recommend a Spotify playlist based on the identified emotion. This system will utilize various modules for emotion recognition, classifier training, and integration with the Spotify API.

Introduction

Human-computer interaction (HCI) is crucial in the rapidly evolving world of artificial intelligence (AI). As virtual assistants Siri and Alexa and physical service robots become more prevalent in society, they must be able to understand and respond appropriately to human emotions. This includes tasks such as evaluating the success of a marketing campaign and providing care for the elderly. A deeper awareness of human emotions leads to better service delivery and a greater understanding of people's needs.

While it may be easy for humans to understand and recognize emotions, it can be more challenging for machines. This is where machine learning comes in. By training algorithms to recognize emotions, machines can better understand and respond to the needs of humans. Emotion detection from voice is a specific application aiming to identify a person's emotions from an input voice sample.

The emotion detection process involves extracting relevant features from audio data and applying different classifiers to identify expressed emotions. Emotions can be characterized by various acoustical properties such as energy, pitch, rhythm, and loudness. To extract these properties from audio files, we can use feature extraction techniques to remove unwanted noise and convert the audio data into digital form. For example, the 'librosa' module can be used to extract features such as the Mel frequency, short-term power spectrum (Mfcc), and pitch (Chroma) of the audio signal. Once these features are extracted, we can use a variety of classifiers to match them to the corresponding emotions.

We employed the artificial neural networks (ANN) algorithm in this project. ANN uses a variety of inputs to determine the optimum output based on the specified output. Here, we are utilizing a RAVDESS dataset with 1400 audio files. Each audio is distinct from the others and contains a different kind of emotion. When we add live audio or a human voice into the system, the algorithm will correctly predict the emotion because it has been trained to recognize the particular feeling that a voice is having.

Data Exploration

For this project, we are using the RAVDESS dataset, which is the audio-visual database of emotional speech.

Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS):

Speech audio-only files (16bit, 48kHz .wav) from the RAVDESS. The RAVDESS contains 1440 files: 60 trials per actor x 24 actors = 1440. The RAVDESS contains 24 professional actors (12 female, 12 male), vocalizing two lexically-matched statements in a neutral North American accent. Speech emotions include calm, happy, sad, angry, fearful, surprised, and disgusted expressions. Each expression is produced at two levels of emotional intensity (normal, strong), with an additional neutral expression.

Each of the 1440 files has a unique filename. The filename consists of a 7-part numerical identifier (e.g., 03-01-06-01-02-01-12.wav). These identifiers define the stimulus characteristics:

Filename identifiers

- Modality (01 = full-AV, 02 = video-only, 03 = audio-only).
- Vocal channel (01 = speech, 02 = song).
- Emotion (01 = neutral, 02 = calm, 03 = happy, 04 = sad, 05 = angry, 06 = fearful, 07 = disgust, 08 = surprised).
- Emotional intensity (01 = normal, 02 = strong). NOTE: There is no strong intensity for the 'neutral' emotion.
- Statement (01 = "Kids are talking by the door", 02 = "Dogs are sitting by the door").
- Repetition (01 = 1st repetition, 02 = 2nd repetition).
- Actor (01 to 24. Odd-numbered actors are male, even-numbered actors are female).

In [3]:
```
#Emotions in the RAVDESS dataset
emotions={
  '01':'neutral',
  '02':'calm',
  '03':'happy',
  '04':'sad',
  '05':'angry',
  '06':'fearful',
  '07':'disgust',
  '08':'surprised'
}

#Emotions to observe
observed_emotions=['disgust', 'happy', 'angry', 'sad', 'fearful', 'neutral', 'surprised']
```

There are 8 emotions in the RAVDESS dataset and we are using 7 emotions that are in the observed emotion.

Proposed Methodology

System implementation turns models into functional systems or new applications using newly developed designs. This project is a critical stage involving converting the dataset, extracting relevant features, testing, and training the system. These steps are necessary to ensure that the system functions correctly and accurately identifies speech emotions.

A. Proposed System

We are identifying emotions from a speech in this experiment. For this project, we utilized an MLP Classifier, a sound file library to read the sound file, and the 'librosa' library to extract features from the sound file. Therefore, this will enable us to recognize human emotions more accurately.

B. Feature Extraction

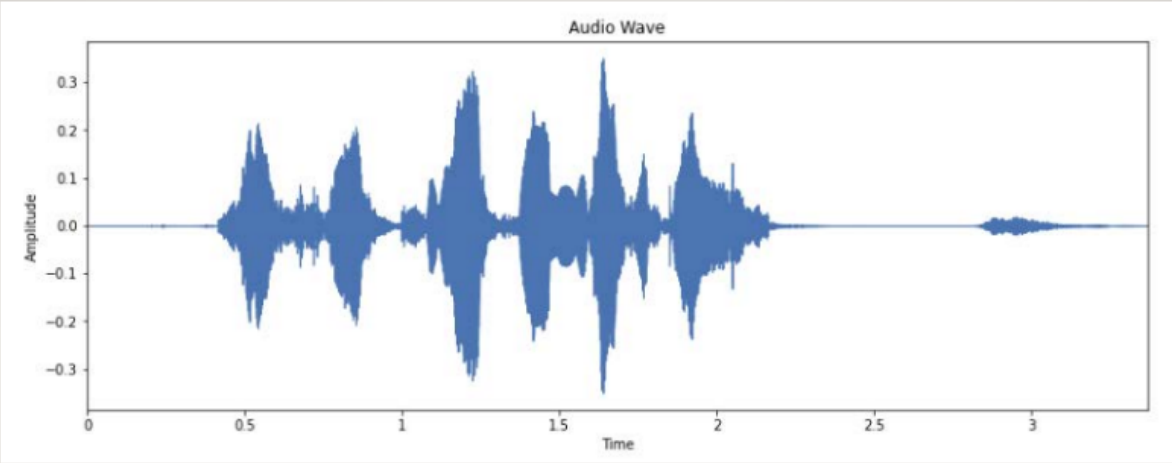

Fig: Waveform

The above plot describes the change in amplitude (loudness) of a signal over the time domain.

The plot above shows the changes in the amplitude or loudness of a signal over time. In the case of audio waves, variations in amplitude and frequency can provide insights into the emotions being expressed. Each audio wave is made up of multiple single-frequency signals, which can be extracted using a mathematical technique called the Fourier Transform. This process converts a time-domain signal into a frequency-domain signal, allowing us to analyze how the amplitude and frequency vary over time. This process of extracting individual, single-frequency signals from an audio file is called spectrum analysis.

```python
#Extract features (mfcc, chroma, mel) from a sound file
def extract_feature(file_name, mfcc, chroma, mel):
    with soundfile.SoundFile(file_name) as sound_file:
        X = sound_file.read(dtype="float32")
        sample_rate=sound_file.samplerate
        if chroma:
            stft=np.abs(librosa.stft(X))
            result=np.array([])
        if mfcc:
            mfccs=np.mean(librosa.feature.mfcc(y=X, sr=sample_rate, n_mfcc=40).T, axis=0)
            result=np.hstack((result, mfccs))
            #print(result)
        if chroma:
            chroma=np.mean(librosa.feature.chroma_stft(S=stft, sr=sample_rate).T,axis=0)
            result=np.hstack((result, chroma))
            #print(result)
        if mel:
            mel=np.mean(librosa.feature.melspectrogram(X, sr=sample_rate).T,axis=0)
            result=np.hstack((result, mel))
            #print(result)
    return result
```

Fig: Function to extract feature

This function extracts the Mfcc, Mel, and Chroma features from a sound file. It takes four parameters: the file name and three boolean parameters representing the features to be extracted. The three features are explained below:

- **Mel**: The Mel scale (short for "melody") is used to represent a sound's perceived pitch. It is based on the way that humans perceive pitch, which is not directly proportional to the frequency of a sound. The Mel scale is designed to be more closely aligned with how humans perceive pitch than the linear frequency scale, which is commonly used in speech and music analysis.
- To convert from linear frequency to Mel scale, a non-linear function is applied to the frequency. This function is based on how the human ear responds to different frequencies, with a greater sensitivity to frequencies in the middle of the range and a decreased sensitivity to frequencies at the high and low ends.
- It shows how the frequency and amplitude evolve over time, with the x-axis representing time, the y-axis representing frequency, and the color representing amplitude. A **spectrogram** is a visual representation of the signal strength or 'loudness' of a signal over time at various frequencies present in the waveform.
- The Mel scale is commonly used in speech recognition and music classification tasks, as it is thought to be more closely aligned with how humans perceive pitch than the linear frequency scale.

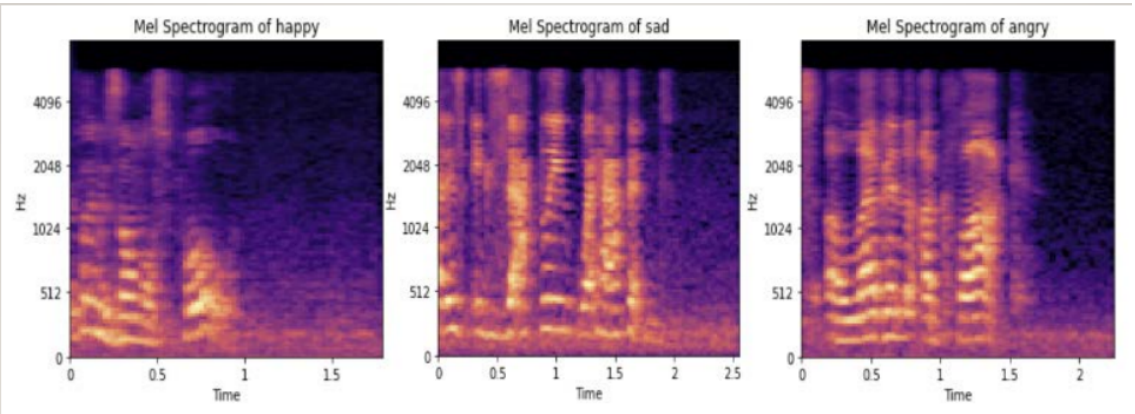

Fig: Mel Spectrogram

- **MFCC:** Mel Frequency Cepstral Coefficients (MFCCs) are a set of features used to represent the spectral characteristics of a sound.
- MFCCs are derived from the Mel-scaled spectrogram of a sound. The spectrogram is transformed using the cepstral transform, which decomposes the spectrum into a set of coefficients representing the sound's spectral shape. These coefficients are the MFCCs.
- MFCCs help represents the spectral characteristics of a sound because they capture the overall shape of the spectrum and are relatively insensitive to changes in the absolute level of the sound. This makes them useful for tasks such as speech recognition, where the level of the sound may vary significantly, but the spectral characteristics are still crucial for identifying the words being spoken.

- **Chroma**: Chroma refers to pitch classes in music. It represents the pitch content of a sound, indicating which pitches are present and how strong they are.
- In music, the chroma of a sound is typically represented using a 12-dimensional space, with each dimension corresponding to one of the 12 pitch classes (C, C#, D, D#, E, F, F#, G, G#, A, A#, B). The value of each dimension indicates the strength of the corresponding pitch in the sound.
- Chroma features are commonly used in music classification tasks, as they provide a compact representation of the pitch content of a sound. They are also used in tasks such as music transcription, where the goal is to identify the pitches in a sound.
- Chroma features can be extracted from an audio signal using various techniques, including the short-time Fourier transform, constant-Q transform, and wavelet transform. These techniques decompose the signal into its constituent frequency components and assign each component to the appropriate pitch class based on its frequency.

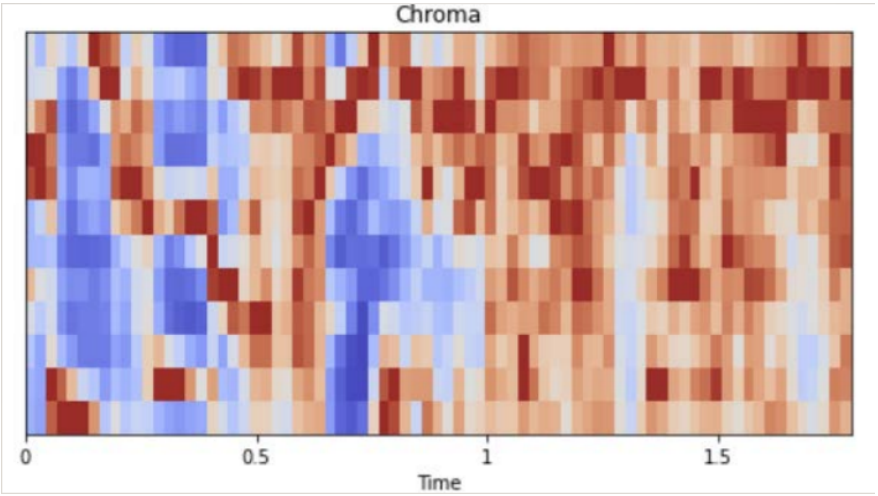

Fig: Chroma

For each of the three, if it exists then a call will be made to the corresponding function from librosa. The mean value will be noted and the result along with the feature value and stored in a file.

C. Testing and Training data

The relative size of the test set is a parameter that is taken into account when loading the data. Both X and Y are void.

```
#Split the dataset
x_train,x_test,y_train,y_test=load_data(test_size=0.25)
```

Fig: Test train split

If the emotion being analyzed is present in the list of observed emotions, the system will check for it using lists and functions. The features and emotions will be assigned to X and Y, respectively. The function for testing and training will then be called, with 25% of the audio being used for training and 75% being used for testing. The MLP Classifier will be used for classification.

D. Multilayer Perceptron Classifier (MLP) is a feedforward artificial neural network used for classification tasks. It is composed of multiple layers of artificial neurons or perceptrons. MLPs are trained using a variant of the backpropagation algorithm, which involves adjusting the weights of the connections between neurons based on the error between the predicted output and the accurate output of the network.

One key feature of the MLP is that it can learn nonlinear relationships between the input and output data, which makes it a powerful tool for many classification tasks. This is because each hidden layer can apply a nonlinear activation function to the input data, allowing the network to capture complex patterns in the data that would not be possible with a linear model.

The structure of an MLP consists of an input layer, one or more hidden layers, and an output layer. The input layer receives the raw input data and passes it through to the hidden layers, where it is transformed and processed using the activation functions. The output layer then produces the final prediction based on the processed input data.

In terms of training and evaluation, it is common to split the data into training and testing sets. The training set is used to fit the parameters of the MLP, while the testing set is used to evaluate the model's performance. This helps ensure that the model is generalizable and not overfitting the training data. We have performed train_test_split and we have given 75% as training data and 25% as testing data for training and testing the model using MLP Classifier, where all these features come under independent data and emotions come under dependent data.

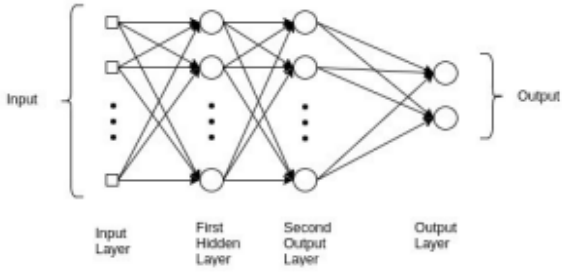

Figure1. Multilayered Perceptron.

E. Music Recommendation

In order to recommend music playlists based on emotion, we first utilized the output emotion from our previous model as input for the playlist recommendation system. We installed the 'spotipy' library, which provides a Python interface to the Spotify API, and used it to authenticate our API access and create a Spotify object. This object was then used to search for tracks and create playlists based on the identified emotion. The user is prompted to enter their user ID, playlist ID, and a list of track URIs as arguments, which will be used to add the tracks to the playlist previously created.

Conclusion and Results

In our project, we first explored the effect of different parameters such as model dimension, number of epochs, and partition ratio between training and test datasets on classification accuracy through experimentation. By altering these parameters, we could assess our model's performance in different scenarios. When classifying among ten classes, a lighter CNN design with 75% of training data and 25% of test data produced better results than a more profound CNN architecture. This model's accuracy was found to be between 50%-70%. The considerable variation in the accuracy is because when we add more emotions, our accuracy for the model goes down as emotions are very subjective and difficult to predict for a machine. When we reduced our number of predicted outputs, our accuracy went to approximately 70%.

The deeper CNN model performed exceptionally well when categorizing data into two classes. This is because more coaching samples were available for categorizing data into two classes, whereas the training dataset was split into ten sections when an analogous model was chosen to classify among ten classes.

Once the emotion was predicted, our program used the Spotify API and the 'spotipy' library to recommend music playlists based on the predicted emotion.

References

- SPEECH BASED EMOTION RECOGNITION
- Multilayer Perceptron - an overview | ScienceDirect Topics
- Get Available Devices | Spotify for Developers
- GitHub - aj-naik/Emotion-Music-Recommendation: Flask web app for recommending music based on your facial expressions using FER 2013 dataset and Spotify api