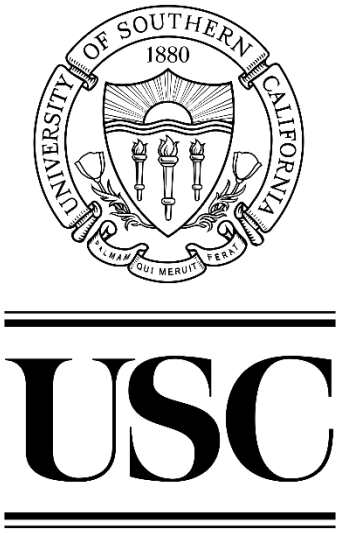
EE 599 DEEP LEARNING PROJECT

"MULTIMODAL SENTIMENT ANALYSIS"



SUBMITTED BY:

Prajakta V Karandikar, Anjana Niranjan, Po Yu Yang

Contents

INTRODUCTION

Sentiment analysis is a field of study that deals with analyzing sentences for various purposes. It can be used to understand emotions like anger, happiness and so on behind a sentence, it can be used to understand customer reviews and so on. The use of AI in analyzing sentiments is increasing and the variety of applications is also growing.

In this project, we have worked on analyzing sentences to classify them into positive, negative and neutral classes. This has been done by analyzing the audio and the text of each sentence. The performance of analysis on text, audio and a combination of both have been presented.

DATASET

The dataset of we have used is the MELD dataset [1]. This contains video clippings dialogues from a TV show. These clipping have been labeled in two ways – one according to sentiment, and one according to the emotion, and this information is given as a csv file. It contains 9900 data points for training, 1100 points for validation and 2610 points for testing. The details have been explained below.

Class labels according to "Sentiment" – the clippings have been divided into:

1. Positive
2. Negative
3. Neutral

Class labels according to "Emotion" – the clippings have been divided into:

1. Sadness
2. Surprise
3. Neutral
4. Joy
5. Anger
6. Disgust
7. Fear

The data extraction for our included the following steps:

1. For text data, information and the corresponding labels were taken from the supporting CSV file.
2. For audio data, information was extracted from the video clippings using a python module, *moviepy,* and stored in .mp3 format. The labels for each of these audio files was again taken from the supporting CSV file given in the main dataset

This data was preprocessed to extract text and audio features to prepare a file dataset for training our models, as explained in Section 3.

INITIAL WORK – DATA PREPARATION

Text features

For analyzing text data, it must be converted into numerical sequence that can be fed to the model. This is done with some preprocessing using the NLTK library for python. The steps followed are:

1. We tokenize every line and split the sentence into separated word.
2. We remove all the unnecessary word, such as stop words, and anything that is not an English word, or any word length that is not longer than 1.
3. We lemmatize every word back to its basic format.

After this process, we use the following as our features:

1. Length of the remaining words in sentence. We also use
2. pos_tag(part-of-speech tagging) to calculate the frequency of different part of speech in every sentence such as
    a. verb,
    b. noun,
    c. adjective and
    d. adverb,

    so that our model can identify the same word but used in different way.

We make them become some of our new defined features.

We then use Tokenizer to convert text into numerical sequence in order to map it to the embedding matrix. For embedding matrix, we use the Stanford GloVe dataset to construct it. [3][4]

The process can be summarized as below:

Fig. Text Feature Extraction

Audio features

The features we are trying to capture from the audio data are based on the voice modulation. The way something is said, the tone and the pitch is being considered. We have used the python library, *Librosa,* to extract the following features [2]:

1. MFCC (Mel-Frequency Cepstral Coefficients):
   MFCC is also called as 'Most-frequently considered coefficients'. Any sound produced by humans is mainly determined by how their vocal tract is shaped (including tongue, teeth, etc). In order to correctly represent human generated sound, it is essential to accurately determine the shape of their vocal tract. The human vocal tract can be represented by an envelope of the time power spectrum of the speech signal. MFCC (which is nothing but the coefficients that make up the Mel-frequency cepstrum) is that feature which accurately represents this envelope. [5][6]

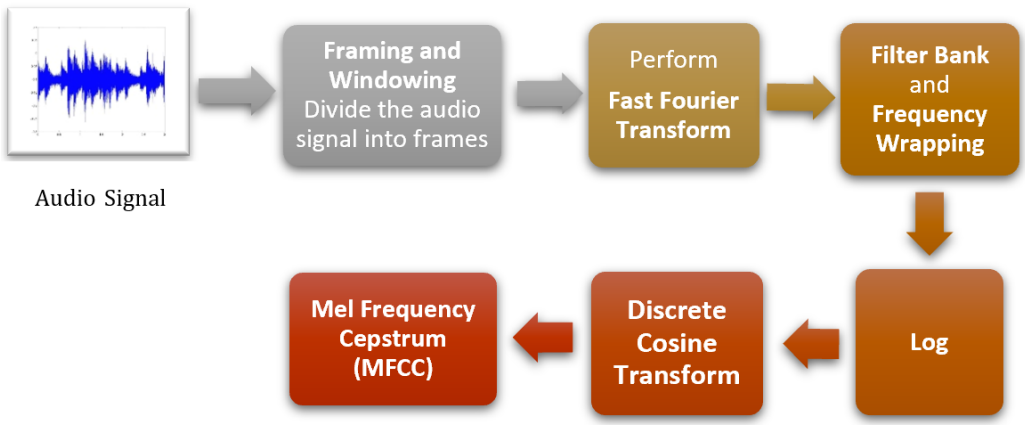


Fig. MFCC (Audio) Feature Extraction

2. Chroma:

Chroma feature also known as chromagram or "pitch class profiles" as they are closely associated to the twelve different classes of pitch. It basically depicts a distribution of the signal's energy across a predefined set of pitch classes (which is called chroma). An important property of chroma features is that they capture harmonic and melodic characteristics of music/audio. The two main chroma features are listed below:

a) Chroma vector: It is representation of the spectral energy consisting of 12 elements. The twelve pitch classes of equal temper in western music are represented by the bins (called semitone spacing).

b) Chroma deviation: It is the standard deviation of the 12 chroma coefficients. [7]

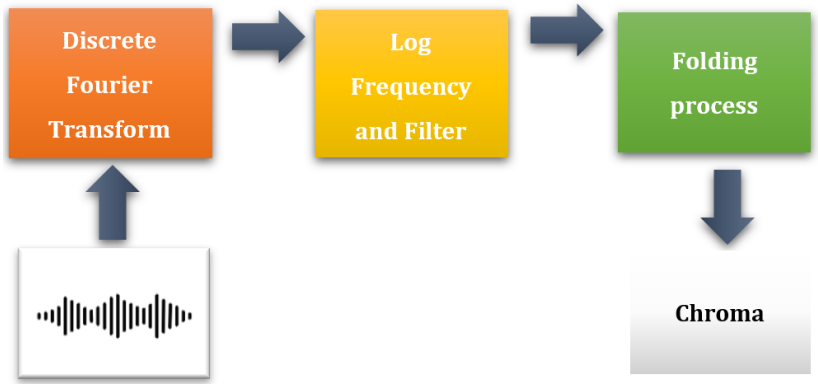


Fig. Chroma (Audio) Feature Extraction

3. Tonnetz:
   The Tonal Centroids (or Tonnetz) contain harmonic content of a given audio signal. It is a pitch space defined by the relationships between the pitches in music in just intonation. [8][9]

4. Mel spectrogram:
   In the case of audio signals like music and speech, the frequency content of signal varies over time and hence they are known as non-periodic signals. Thus, spectrums are computed by performing Fast Fourier Transform on various segments of the audio signal that are windowed. We get a spectrogram by computing this FFT on windowed signal segments that overlap.
   So, a spectrogram is basically a bunch of FFTs stacked on top of each other. It is a way of pictorially representing the loudness, or amplitude factor of an audio signal as it varies at different frequencies over time.
   The mel scale, proposed by Stevens, Volkmann, and Newmann in 1937, is a unit of pitch such that distances in pitch that are equal sounded equally distant to the listener. Therefore, a mel spectrogram is a spectrogram where the frequencies are converted to the mel scale. [10]

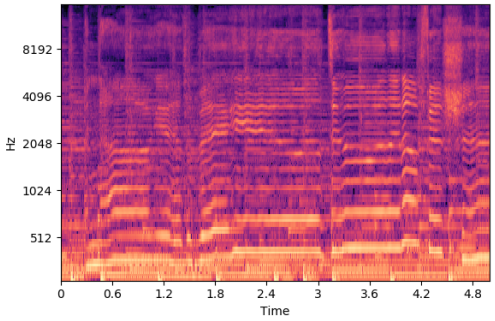


Fig. Typical Mel Spectrogram

5. Contrast:
   There are sub-bands into which each spectrogram frame is divided. For each of these sub-bands, the energy contrast is estimated by comparing the mean energy in the top (peak energy) to that of the bottom (valley energy). If the contrast value is high, it generally means clear, narrow-band signals, while for low contrast values it refers to broad-band noise.
   Thus, the spectral peak, spectral valley, and their difference in each sub-band of frequency is what octave-based spectral contrast considers. The harmonic components in music/audio data is represented by spectral peaks while non-harmonic components, or noises, often appear at spectral valleys. Thus, spectral contrast can be used to give an idea of the relative distribution of harmonic and non-harmonic components in a spectrum. [11][12][13]

The final features extracted can be summarized as shown in the figure below. So, for every audio file, a total of 193 features are extracted and stored as arrays.

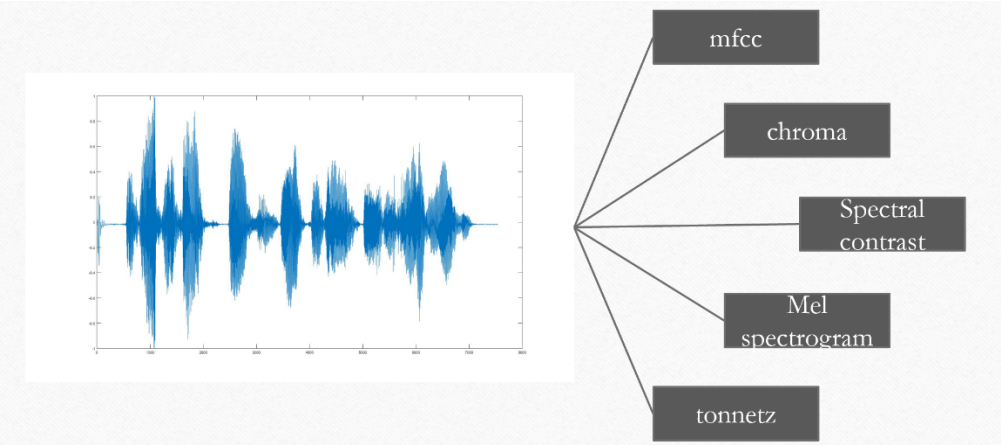


Fig. Extracted features from Audio

MULTIMODAL DATA

Modality

In today's world, with the invention of so many applications and ways of communication, it has become easier for us to express ourselves and convey our messages to other people. And thus, we can now share our thoughts which can be in the form of audio messages, or even videos (which are as good as communicating face to face with someone) and of course, in textual form. All these ways of sharing messages play a key role especially when it comes to Sentiment Analysis, as now we can input a dialogue that can be either an audio file, a video or text. Each of these forms provide information in different ways which is then used for analyzing the sentiment or emotion.

Modality is basically how we see things, a way of experiencing things. In terms of human perspective, it can be referred to as our sense of vision, touch, or smell, etc. In Deep Learning sense, modality is nothing but the various forms of data that we can get and hence use, like video or images- visual form, text or statements, or audio, vocal statements- sound form. And multimodal means relating these multiple modalities and using them in some way.

Multimodal analysis

This project uses two modalities, text and audio.

With features extracted from text as an input to the neural network model, we work with features that are based on textual phrases, sentences, verbs, adjectives, and so on. It is essential to note here that when work only with text data, we do not consider how a dialogue is being said, what is going on behind the scene, the feelings. All that is obtained is purely based on the grammatical aspects of the sentence/utterance and not the feel aspect.

With audio data as an input and usage of features that are extracted from audio, we get additional information such as the tone when a particular utterance is made, the voice modulation, pitch factor, rate of utterance (speed) and also some other para lingual aspects of speech. These factors provide additional

details about the emotion or sentiment behind the dialogue by stressing more on how that utterance was made. However, it does not include the grammatical aspect that the text data provides which can provide insights as to what is being said and can help in some way for the analysis of sentiment.

Therefore, in order to increase the information extracted by using just one modality, we make use of both text and audio data to predict the emotion behind an utterance giving rise to multimodal analysis, which is expected to achieve better results and performance compared to ones obtained using just text or audio data. [13][14][15]

ARCHITECTURES

All the models are built on Keras, with TensorFlow backend. The layers that have been used for each of the model is explained below.

Text sentiment analysis

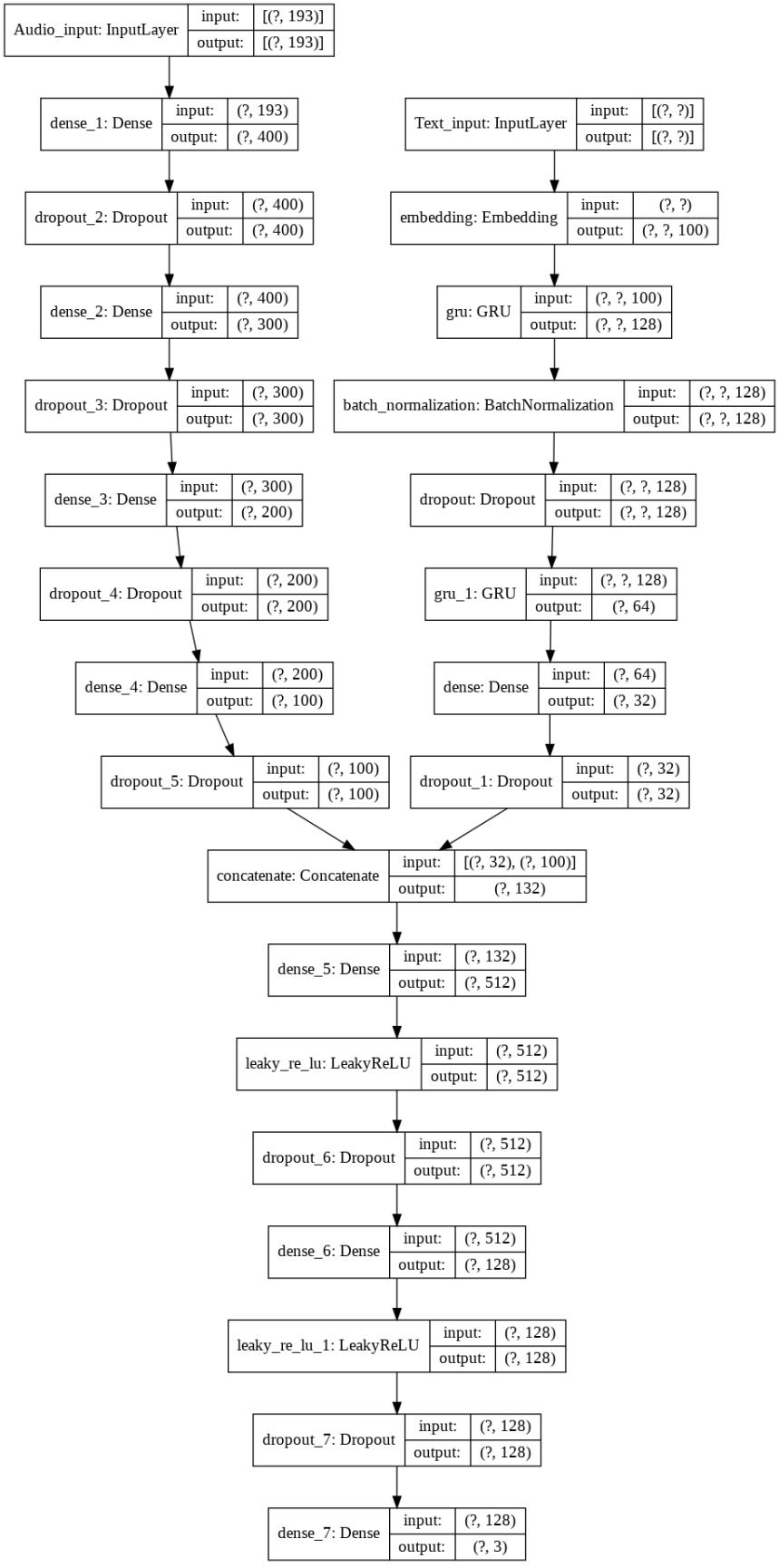
The model we used for text data are 2 GRU layers, one with 256 nodes, one with 32 nodes, also a Batch normalization layer and dropout layers with 0.2. The total number of parameters was roughly 1,304,336, because of the embedding layer, so some of them are not trainable, only 342,060 of parameters are being trained.

Audio sentiment analysis

The dataset has been prepared in a way such that each file has just one emotion or sentiment associated with it. Considering this, the model that was built for the audio analysis has Dense layers and dropout layers. We have used 4 dense layers of 512, 256, 128 and 64 nodes each, with dropouts of 0.2.

Multimodal sentiment analysis

In multimodal analysis, since we are trying to feed both the text and the audio inputs to the model, we have considered two sub-models to take the text and audio inputs and concatenated them to Dense layers further on. The architecture is as shown in the figure below.


Audio_input: InputLayer | input: [(?, 193)] | output: [(?, 193)]
dense_1: Dense | input: (?, 193) | output: (?, 400)
dropout_2: Dropout | input: (?, 400) | output: (?, 400)
dense_2: Dense | input: (?, 400) | output: (?, 300)
dropout_3: Dropout | input: (?, 300) | output: (?, 300)
dense_3: Dense | input: (?, 300) | output: (?, 200)
dropout_4: Dropout | input: (?, 200) | output: (?, 200)
dense_4: Dense | input: (?, 200) | output: (?, 100)
dropout_5: Dropout | input: (?, 100) | output: (?, 100)
Text_input: InputLayer | input: [(?, ?)] | output: [(?, ?)]
embedding: Embedding | input: (?, ?) | output: (?, ?, 100)
gru: GRU | input: (?, ?, 100) | output: (?, ?, 128)
batch_normalization: BatchNormalization | input: (?, ?, 128) | output: (?, ?, 128)
dropout: Dropout | input: (?, ?, 128) | output: (?, ?, 128)
gru_1: GRU | input: (?, ?, 128) | output: (?, 64)
dense: Dense | input: (?, 64) | output: (?, 32)
dropout_1: Dropout | input: (?, 32) | output: (?, 32)
concatenate: Concatenate | input: [(?, 32), (?, 100)] | output: (?, 132)
dense_5: Dense | input: (?, 132) | output: (?, 512)
leaky_re_lu: LeakyReLU | input: (?, 512) | output: (?, 512)
dropout_6: Dropout | input: (?, 512) | output: (?, 512)
dense_6: Dense | input: (?, 512) | output: (?, 128)
leaky_re_lu_1: LeakyReLU | input: (?, 128) | output: (?, 128)
dropout_7: Dropout | input: (?, 128) | output: (?, 128)
dense_7: Dense | input: (?, 128) | output: (?, 3)

RESULTS

Text sentiment analysis

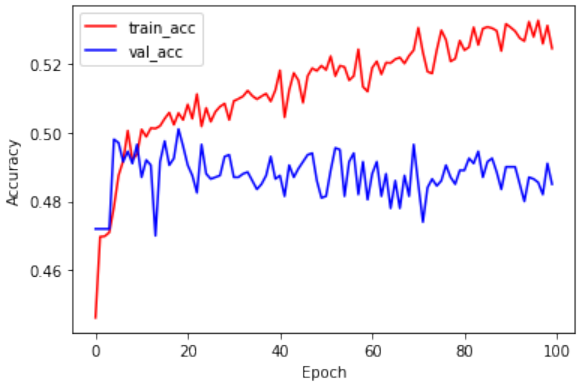
On training the architecture explained above for the text input, we were able to obtain a test accuracy of 47%. [16]

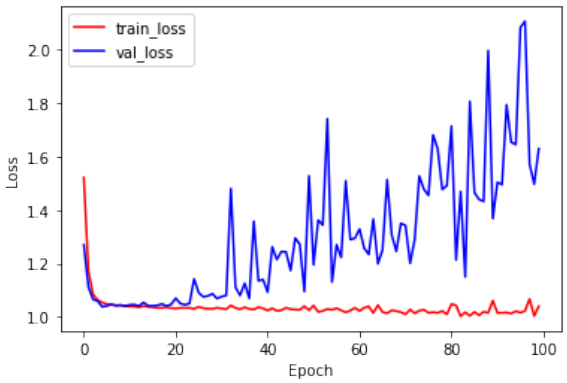
Audio sentiment analysis

On training the architecture explained above for the audio input, we were able to obtain a test accuracy of 48.2%. [17]

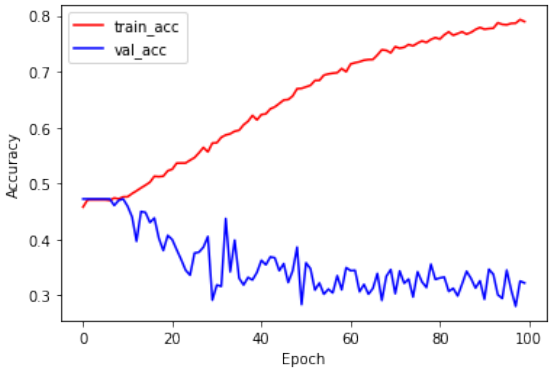
Multimodal sentiment analysis

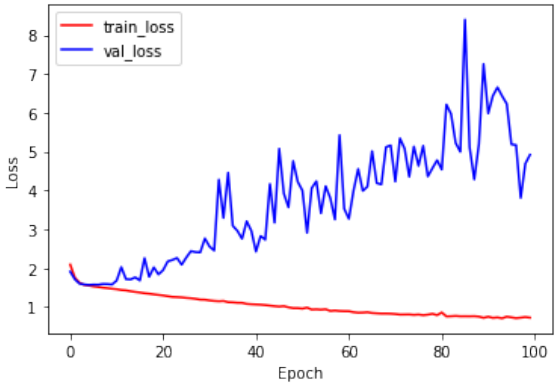
On training the architecture explained above for multimodal input, we were able to obtain a test accuracy of 51.5% for input with 3 classes. [1]





On training the same model for input with 7 classes, we were able to get a test accuracy of 48.2%.





OBSERVATIONS

As seen from the results, there has been a slight improvement in the test accuracy when we used multimodal input as compared to text or audio inputs separately.

The test accuracy for the 7-class input has not been very high. One of the factors affecting this is that we have used the same data for training and the number of training inputs for each class has reduced.

LEARNING OUTCOME

From this experiment, we have observed that there can be better prediction using multimodal inputs, but the feature selection is very important and affects the learning.

Through this project, the importance of dataset preparation became clearer to us. Over 40% of the time was spent on preparing the train and test sets, which included data conversion to usable formats, defining the features we want to train with, and extracting these features. The entire process from defining the model to the testing and obtaining accuracies will be based on this step and data preparation.

Our dataset consisted of data points as .mp3 files, in which each file had one emotion/sentiment associated with it. This meant that we could not show a change of emotion within one training input and hence using of RNNs would not make a difference. So, if we had data that could show such emotion transitions, we could have used RNNs to improve.

To improve this, the data can be modeled in a different way so as to be able to use RNNs on audio and the overall model, to improve accuracy.

REFERENCES

[1] https://github.com/SenticNet/MELD

[2] https://librosa.github.io/librosa/

[3] https://www.datacamp.com/community/tutorials/text-analytics-beginners-nltk

[4][6] https://towardsdatascience.com/how-i-understood-what-features-to-consider-while-training-audio-files-eedfb6e9002b

[5] https://www.researchgate.net/figure/Flowchart-of-MFCC-extraction-procedure_fig3_257135369

[7] https://librosa.github.io/librosa/generated/librosa.feature.tonnetz.html

[8] http://www.nyu.edu/classes/bello/MIR_files/tonality.pdf

[9] https://medium.com/analytics-vidhya/understanding-the-mel-spectrogram-fca2afa2ce53

[10] http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.583.7201&rep=rep1&type=pdf

[11] https://librosa.github.io/librosa/generated/librosa.feature.spectral_contrast.html

[12] https://musicinformationretrieval.com/spectral_features.html

[13] https://arxiv.org/pdf/1810.04635.pdf

[14] https://towardsdatascience.com/multimodal-deep-learning-ce7d1d994f4

[15]https://www.researchgate.net/publication/240954177_Multimodal_approaches_for_emotion_recognition_A_survey

[16] https://github.com/ManishShettyM/NLP-Offensive-Text-Detection?fbclid=IwAR1dA-wZu4Z6I5RSUSFTmt_PD6OfvzcjV5SphLCFk14JP8_EviKhRrhLj-c

[17] https://github.com/shaharpit809/Audio-Sentiment-Analysis

[18] https://medium.com/dair-ai/state-of-the-art-multimodal-sentiment-classification-in-videos-1daa8a481c5a