Project No: SCSE20-0777

Aiding Therapy Using Speech Emotion Recognition

CZ4079 Final Year Project

Submitted by: Koh En Rong

Matriculation No: U1822302K

Supervisor: Associate Professor Dr Qian Kemao

School of Computer Science and Engineering

A final year project report presented to the Nanyang Technological University in partial fulfilment of the requirements of the degree of Bachelor of Engineering

Abstract

In the past 20 years, mental health has come to light within society. The stigma surrounding mental illness is declining thanks to the increasing awareness and encouragement through social media and digital platforms. The growth in psychologists and therapists can also be seen in recent years.

Not only did the mental health industry has an increase in patients and counsellors, the advancement of technology integrating with this field is visible in the present day of mental health care. It brought a significant impact on helping people in need during this period of time. Research on artificial intelligence also improved the quality of therapy, bringing it closer to people who are struggling and taking over virtually. Nonetheless, the applications need to be carefully designed and balanced against their limitations, depending on different mental illnesses.

While different kinds of AI have been assisting in the mental health field, such as therapy chatbots and virtual therapists, a lack of recognizing human emotions can be commonly seen in AI systems, especially through speech. Speech Emotion Recognition became a research topic in a wide range of applications and became a challenge in speech processing.

In this project, an AI Speech Emotion Recognition system is experimented with using Deep Learning techniques to alternative traditional methods like Support Vector Machine or Hidden Markov Model. We will explore the use of a Convolutional Neural network, a type of Deep Learning method, to train and predict human emotions. We will also examine the different types of time-frequency features in audio signal processing and how they help in classifying human emotion. A SER system with a visual modality will also be developed to test on real-time prediction.

Acknowledgement

I would like to express my very great appreciation to my supervisor A/P Qian KeMao for giving me an opportunity to research an idea of my own. His trust, encouragement and patience throughout the duration of the project have given me the endurance to continue on. Many thank for his suggestions and feedback to improve on my projects. I would also like to extend my sincere thanks to Research Associate He Xiaoyu for his constructive feedback on the project during meetings. I would also like to express my gratitude to the SCSE GPU Cluster (Teaching) Team for allowing me to conduct my experiments on their GPU server. Finally, I would like to specially thank my family for the support and comments they gave whenever I faced any challenges.

Table of Contents

1 Introduction

1.1.Background

In the study of clinical therapy, due to the complexity of the field and multi-factoriality in mental health diagnosis, professional counsellors and therapists are required to have a broad scope of perspective and point of view. Not only is the knowledge of the field important in counselling, core skills such as active listening, social and communication skills are also needed for one to treat patients effectively.

Even the most competent counsellor with adequate knowledge and skills has the possibility of making mistakes. Studies have shown that people with bipolar disorder are delayed for an average of 13.2 years before they are diagnosed, most are misdiagnosed with depression[1]. Other typical case studies that were published in The Lancet showed that only 47.3% of the patient cases were correctly identified as depression by general practitioners. Of the wrongly diagnosed cases, the rate of false positives outweighs the false negatives by at least 50%[2].

The possible reasons for misdiagnosis of mental disorders can be complex and multiple. Patients contribute mainly to its cause as they may falsely report their symptoms due to shame or embarrassment. On the other hand, practitioners can also be the factor by failing to recognize nonverbal communication such as body language, facial expressions and tone of voice. These unconscious behaviours may better portray the patient's emotional state and offer valuable information that a patient may be reluctant or incapable to put into words[3].

Biasness also plays an extensive part in misdiagnosing mental disorders. Like all humans, therapists have their values, assumptions, and beliefs. They may unconsciously have certain biases and underlying presumptions about a patient. These stereotypes can affect the diagnosis and limit the effectiveness of the treatment of a patient[4]. Not only that, a study on gender role conflict shows that the power dynamic of the practitioner and the patient can also contribute to clinical bias[5].

1.2.Motivation

Researchers have been finding various ways to aid in the diagnosis and treatment of mental disorders. Numerous successful methods have integrated artificial intelligence into therapy sessions, such as AI Therapist and mental health chatbots, using emotion recognition through speech and body language[6, 7]. These help to minimize the errors and biases happening. However, the tone of voice is one area that can be further researched by incorporating artificial intelligence, also known as Speech Emotion Recognition SER.

1.3.Objective

This project aims to determine whether the tone of a person can be correctly mapped to the seven basic types of emotions: Happiness, Sadness, Fear, Disgust, Anger, Surprised, and Neutral. This will be done by extracting features from the sample audio of different emotions and training using various machine learning models. The insights from the training will return a suitable model that can predict the subject's emotions via speech.

2 Literature Review

2.1. Speech Emotion Recognition

The most basic natural way to express ourselves is through speech, and we depend on it as a way of communication in our daily life. Together with other forms of communication such as body language and text messages, they can be easily misunderstood. What makes us better understand each other would be the emotion behind it.

It is known that eventually, we will extend this understanding to computers and have artificial intelligence (AI) working on it. Thanks to smart devices such as Siri and other voice assistants, we can use it to start an app or reply to messages using voice commands. The part that's still lacking in research would be speech emotion recognition (SER), which allows AI to detect our emotions and react to them.

It has been more than two decades since the start of SER research[8], and it has been applied to different areas such as call centres[9], mobile services[10], and psychological assessment[11]. Even though it has numerous applications, emotion detection is still a challenging task since emotions are subjective. There is no common agreement on measuring or categorizing them, but they are evaluated by individuals' knowledge and can be easily misinterpreted.

In this SER architecture, different methodologies are used to process and classify speech signals to extract emotions in each audio data. As shown in Figure 1, the overview of SER architecture can be split into six different areas, and there are various approaches and combinations to achieve emotion classification. The areas are surveyed from the left to right direction, with the emotions being embedded in the data on the left and are extracted on the right.

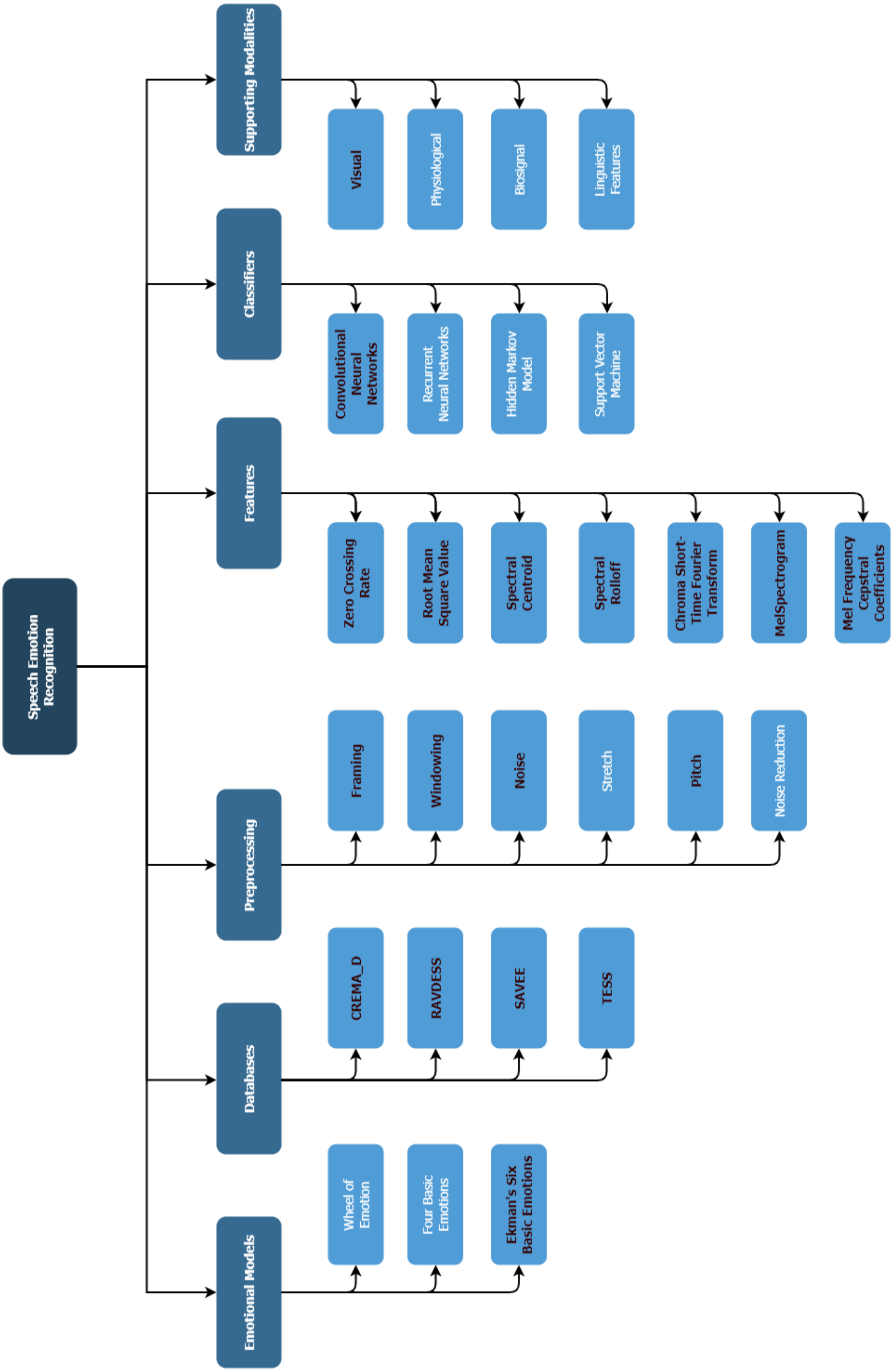


Figure 1. SER Architecture.png

To date, there are over 30 thousand different kinds of emotions and feelings, combining various countries and cultures[12]. Most commonly used are the discrete and dimensional models. Therefore a discrete model is chosen to classify the emotions and source the dataset. After selecting the audio data, they will require preprocessing before their emotional features can be extracted. It is crucial to choose which features to be extracted as they reduce the original audio to its most important characteristics. All the extracted features are then passed through a classification model to be trained. Classical classifiers such as Support Vector Machines[13] and Hidden Markov Model[14] have been used with a high recognition rate of up to 90%. The classifier can also be reinforced by adding supporting modalities, for instance, visual or linguistic, to aid in classification.

2.2.Emotions

The definition of "Emotion" by the Cambridge Dictionary is, a strong feeling such as love or anger, or strong feelings in general. There are in total 34,000 emotions a human can experience, many of which are complex and challenging to understand. Unlike basic emotions, complex emotions vary across people and cultures. It can be in the form of grief, jealousy or regret, or made up of two or more basic emotions [15, 16]. Many theories of basic emotions exist, which includes but aren't limited to:

- **The wheel of emotion**

  Developed by Robert Plutchik, the emotions are categorized into eight primary emotions and grouped into polar opposites, Joy to Sadness, Anger to Fear, Trust to Disgust and Surprise to Anticipation[15].

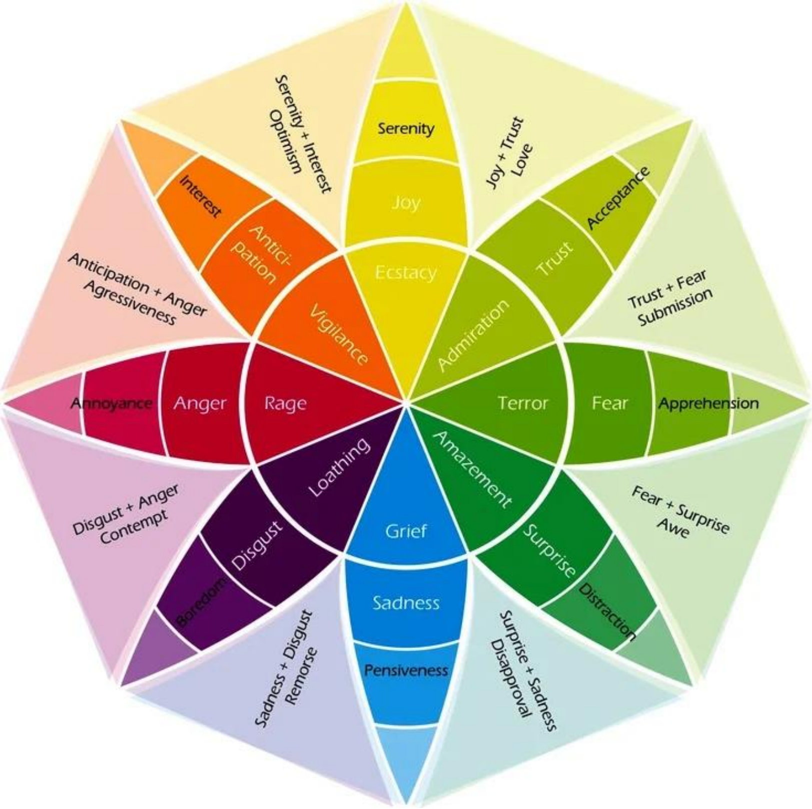


Figure 2. Wheel of emotion.png[15]

- **Four Basic Emotions**

  Based on newer research by the Institute of Neuroscience and Psychology at the University of Glasgow, some facial signals are very similar, suggesting that biologically the distinction between anger and disgust and between surprise and fear are the same. Therefore, this will end with four basic emotions: Happy, Sad, Fear/Surprise, and Anger/Disgust[17].

- **Ekman's Six Basic Emotions**

In the 1970s, Paul Ekman, a well-known psychologist, identified six basic emotions to be universally experienced in all human cultures. That includes Happiness, Sadness, Fear, Disgust, Surprise, and Anger[18]. In this project, we will be using mainly the concept of Pual Ekman's basic emotions.

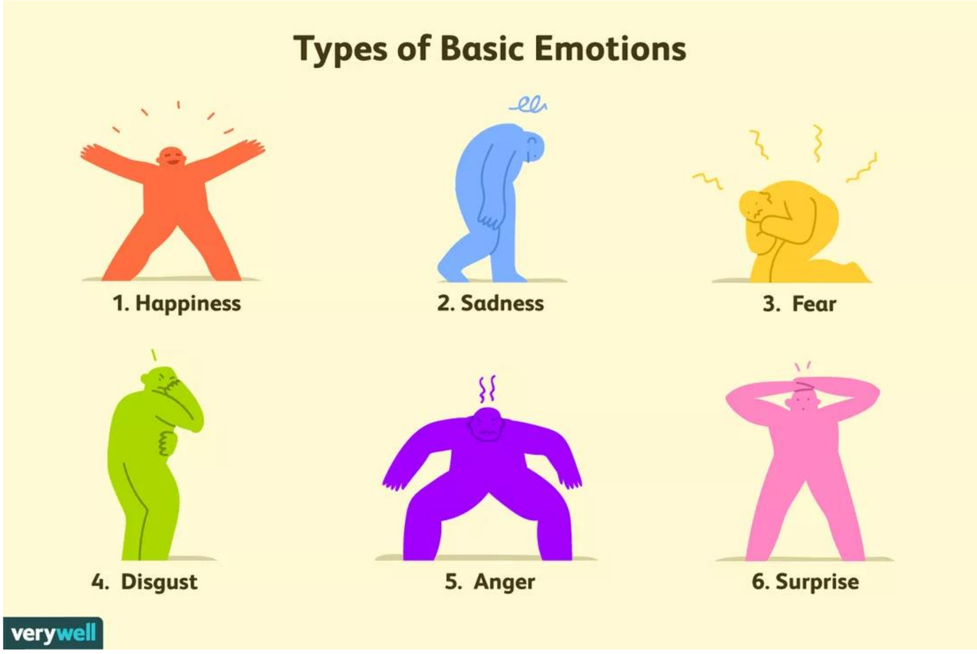


Figure 3. 6 basic emotions.png[18]

2.2.1. Happiness

Happiness is often seen as a pleasant emotional state. It is categorized by feelings of joy, satisfaction and contentment. While this is considered one of human's basic emotions, an individual's culture can affect what creates happiness, such as having a high paying job or owning a house. People also believed that happiness could increase longevity, physical and mental health. The tone of voice is usually upbeat and have a pleasant way of speaking.

2.2.2. Sadness

Sadness is often defined as a transient emotional state. It is categorized by feelings of disappointment, hopelessness and grief. Human expresses it in numerous ways including crying, quietness or even withdrawal. Prolong and severe experiences of sadness can decrease mental health and can turn into depression. The tone of voice is usually soft, gloomy and can sound depressing.

2.2.3. Fear

Fear is a powerful emotion that plays a significant part in survival. When facing danger or experiencing fear, one will have an instinctive reaction known as the fight or flight response. This emotion has a different trigger point for everyone and is an emotional response to immediate danger or can be developed to anticipate potential threats. It can also be decreased by the idea of exposure therapy, which is to expose to things that induce fear gradually in a controlled and safe manner. The tone of voice is usually shaky, tremulous or even stuttered.

2.2.4. Disgust

Another basic emotion that can originate from numerous things, including unpleasant taste, sight or smell, is believed to be evolved as a reaction to foods that is unsafe for consumption. Foul-smelling or tasting food, blood, poor hygiene, or rot can also trigger a disgust response that tells the body to avoid. The tone of voice usually sounds repulsive, repelling or disappointed.

2.2.5. Anger

Anger can notably be a powerful emotion that is categorized by feelings of frustration, hostility or agitation. Like fear, anger can have an impact on our body's fight or flight response. When a danger provokes the feeling of anger, one may tend to fend off the threat and protect oneself. It can be a positive emotion that helps with clarity and motivation or be a problem when excessive and uncontrollable. The tone of voice is usually gruffly or yelling.

2.2.6. Surprise

Surprise is usually a brief startling response following something unexpected. It can be a positive, negative or neutral emotion that can trigger the fight or flight response. When startled, one may have a burst of adrenaline that helps the body to have instinctive reactions. Researchers found that memories of surprising or unusual occasions tend to stand out more than others, as humans tend to notice unexpected events disproportionately. The tone of voice is usually yelling, gasping or screaming.

2.2.7. Neutral

In this project, we will also be using what we call neutral emotion. A Neutral emotion is believed to be an emotional state of neither pleasure nor pain. It covers a range in the middle of the spectrum of feelings. Some researchers have considered it's impossible to feel neutral as humans always feel something or emotions are an affective state, and neutral is neither positive nor negative[19]. However, other beliefs debate that neutral emotion exists and is defined as feeling indifferent, nothing in particular, or lack of preference. An article called "What about neutral feelings?" said that neutral feelings belong to a peaceful and sublime kind of pleasure[20]. In general, humans have a standard concept of their own, what neutral feels like. As the tone of a neutral state can be objective, it is known to be calming, relaxing, or gentle.

2.3. Database

A huge part of the classification process depends on an accurate and reliable dataset. The quality of the data can directly affect the recognition process. Low quality or wrongly labelled databases may cause false predictions. Therefore precisely labelled dataset is needed to assure a high recognition rate.

Speech emotions databases can be divided into three types[21]:

- Acted (Simulated) Emotional Speech Database
- Elicited (Induced) Emotional Speech Database
- Natural Emotional Speech Database

Simulated corpora are recorded by professional actors in a sound-proof studio. They are requested to articulate specific words or sentences in different emotions. This is considered an easier and reliable method of data collection as one can control the amount and the range of emotion collected depending on the purpose of the prediction. However, these audios collected through simulated are typically more intense and exaggerated, unlikely to convey real-life emotions perfectly. These might decrease the recognition rate for real-life predictions.

On the other hand, Elicited corpora are obtained by putting speakers through a simulated artificial emotional situation. They participate in an emotional discussion with the anchor, where the anchor makes different relevant circumstances through the discussion to inspire different feelings from the subject, without the speaker's knowledge. These audio data may be more natural than the simulated corpora, but the subjects might not express themselves adequately knowing they were recorded.

Lastly, natural corpora are collected from talk shows, call centre recordings or any emotional conversations in public places. Unlike simulated corpora, the emotions are mildly expressed. These databases are challenging to build as finding a wide range of emotions is not only labouring, but the annotation of the data is highly subjective and debatable. They are also challenging to obtain due to ethical and legal issues when processing and distributing them.

After deciding which kind of database to create, other variables should be considered depending on the research objective, such as age, gender or language. Most of the databases are adult speakers for general usage, while the database of children or certain languages also does exist. Table 1. below shows the databases used in this project. They are built with various English speakers recording specific words or sentences in seven or more different emotions. All of the databases chosen are supported by the theory of Ekman's Six Basic Emotions.

Table 1: Databases used in this speech emotion recognition project.

Database	Language	Size	Emotion	Type	Modalities
Crowd-sourced Emotional Multimodel Actors Dataset (CREMA-D)	English	6 Emotions X 91 Actors (48 male, 43 female) X 4 levels	Anger, Disgust, Fear, Happy, Sad, Neutral	Acted	Audio/Visual
Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS)	English	8 Emotions (5 for songs) X 24 Actors (12 male, 12 female) X 2 levels	Calm, Happy, Sad, Angry, Fear, Surprise, Disgust, neutral	Acted	Audio/Visual
Surrey Audio-Visual Expressed Emotion (SAVEE)	English	7 Emotions X 4 Actors (male)	Anger, Disgust, Fear, Happy, Sad, Surprise, Neutral	Acted	Audio/Visual
Toronto Emotional Speech Set (TESS)	English	7 Emotions X 2 Actors (female)	Anger, Disgust, Fear, Happy, Surprise, Sad, Neutral	Acted	Audio

2.3.1. Crowd-sourced Emotional Multimodel Actors Dataset (CREMA-D)[22]

CREMA-D is a dataset containing 7442 original clips from 91 actors, 48 male and 43 female actors aged between 20 and 74 years old. They come from various races and ethnicities, including African America, Asians, Caucasians, Hispanics etc. Actors are requested to spoke from a selection of 12 sentences using six different emotions (Anger, Disgust, Fear, Happy, Sad and Neutral) and four different emotion levels (Low, Medium, High, and Unspecified). All of the data are captured in an audio and visual format

Due to the large amount of data needed to rate, a total of 2443 participants rated the emotion and its level based on the combined audio and visual, visual alone, and the audio alone. Each participant rated 90 unique clips, 30 visual, 30 audio, and 30 audio-visual, resulting in 95% of the clips having more than seven ratings.

The file naming conventions have four parts in each audio file. An actor id is the first identifier, a four-digit number at the start of the file. An underscore then separates each subsequent identifier.

The second identifier will be the selection of 12 sentences chosen by the actors. Each sentence is paired with a three-letter acronym.

- It's eleven o'clock (IEO).
- That is exactly what happened (TIE).
- I'm on my way to the meeting (IOM).
- I wonder what this is about (IWW).
- The airplane is almost full (TAI).
- Maybe tomorrow it will be cold (MTI).
- I would like a new alarm clock (IWL)
- I think I have a doctor's appointment (ITH).
- Don't forget a jacket (DFA).
- I think I've seen this before (ITS).
- The surface is slick (TSI).
- We'll stop in a couple of minutes (WSI).

The third part of the identifier represents the different types of emotions, each pairing with a three-letter code.

- Anger (ANG)
- Disgust (DIS)
- Fear (FEA)
- Happy (HAP)
- Neutral (NEU)
- Sad (SAD)

Lastly, the fourth part of the identifier represents the different emotional levels, each pairing with a two-letter code.

- Low (LO)
- Medium (MD)
- High (HI)

- Unspecified (XX)

Some examples of the files include:

- 1001_IEO_DIS_HI.mp3
- 1005_DFA_SAD_XX.mp3
- 1006_TAI_FEA_XX.mp3
- 1008_MTI_NEU_XX.mp3
- 1008_IEO_HAP_LO.mp3
- 1010_MTI_ANG_XX.mp3

2.3.2. Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS)[23]

RAVDESS was created in need of dynamic facial and vocal expressions. The database consists of 24 professional actors (12 male, 12 female), vocalizing statements and songs in a neutral North American accent. The speech includes seven different emotions (Happy, Sad, Calm, Angry, Fearful, Surprise and Disgust), and the songs contain five different emotions (Happy, Sad, Calm, Angry, Fearful). Each expression is recorded at two levels of intensity with the addition of a neutral expression. This database contains a total of 7356 recordings, including audio, visual, and audio-visual format. Each recording was rated ten times on emotional intensity, validity and genuineness, rated by 247 individuals. In this project, only 1440 audio files are used for the recognition.

The file naming conventions for this database is separated into seven parts numerical identifier. The identifiers are ordered in: Modality-Channel-Emotion-Intensity-Statement-Repetition-Actor.mp4. Each identifier code is shown in Table 2. below. For example, "03-01-06-01-02-01-12.mp4" would refer to:

- Audio-only (03)
- Speech (01)
- Fearful (06)
- Normal intensity (01)
- Statement "Dogs are sitting by the door"
- 1st Repetition (01)
- 12th Actor, Female (12)

Table 2. RAVDESS file naming convention

Identifier	Coding description
Modality	01 = Audio-Visual, 02 = Visual-only, 03 = Audio-only
Channel	01 = Speech, 02 =Song
Emotion	01 = Neutral, 02 = Calm, 03 = Happy, 04 = Sad, 05 = Angry, 06 = Fearful, 07 = Disgust, 08 = Surprised
Intensity	01 = Normal, 02 = Strong
Statement	01 = "Kids are talking by the door.", 02 = "Dogs are sitting by the door."
Repetition	01 = First repetition, 02 = Second repetition
Actor	01-24 actors, old numbers are male actors, even numbers are female actors

2.3.3. Surrey Audio-Visual Expressed Emotion (SAVEE)[24]

SAVEE database was taped by four native English male speakers, identified as DC, JE, JK and KL. They are postgraduate students and researchers at the University of Surrey ageing from 27 to 31 years old. The recordings include six emotion categories of Anger, Disgust, Fear, Happy, Sad and Surprised. This database also added Neutral as the seventh category. Each emotion consists of 3 common, 2 emotion-specific and 10 generic sentences, having 15 different sentences per emotion. An addition of 3 common and 12 emotion-specific sentences was recorded as neutral to give 30 neutral sentences, making a total of 120 utterances per speaker.

The file naming convention for SAVEE is formed by three parts identifier, (Speaker)_(Emotion)(Sentence no.).wav. Table 3. shows each speaker detail with their unique two-letter speaker ID. The accent of four speakers includes southern English (JE and JK), Welsh (DC), and Scottish (KL), aged between 27 and 31 years old. The current SAVEE database is limited to male-only speakers.

Table 3. Detail of speakers recorded for SAVEE

Speaker ID	Age	Sex	Accent
KL	27	Male	Scottish
JE	29	Male	English
Jk	31	Male	English
DC	31	Male	Welsh

The acronym of the emotion sets the second identifier as the following:

- Anger (A)
- Disgust (D)
- Fearful (F)
- Happy (H)
- Sad (Sa)
- Surprise (Su)

The last identifier is a numeric code that uniquely labels the sentences. With 10 other generic sentences, the 3 common, 12 emotion-specific, and 2 neutral sentences are as follow:

Common – "She had your dark suit in greasy wash water all year."
Common – "Don't ask me to carry an oily rag like that."
Common – "Will you tell me why?"
A – "Who authorized the unlimited expense account?"
A – "Destroy every file related to my audits."
D – "Please take this dirty table cloth to the cleaners for me."
D – "The small boy put the worm on the hook."
F – "Call an ambulance for medical assistance."
F – "Tornado's often destroy acres of farmland."
H – "Those musicians harmonize marvellously."
H – "The Eastern coast is a place for pure pleasure and excitement."

Sa – "The prospect of cutting back spending is an unpleasant one for any governor."
Sa – "The diagnosis was discouraging; however, he was not overly worried."
Su – "The carpet cleaners shampooed our oriental rug."
Su – "His shoulder felt as if it were broken."
N – "The best way to learn is to solve extra problems."
N – "Calcium makes bones and teeth strong."

An example of an audio file would be "JE_n04.wav".

2.3.4. Toronto Emotional Speech Set (TESS)[25]

The TESS database was recorded by the University of Toronto, with 200 target words spoken by two actresses recruited from the Toronto area. Each audio carries a phrase of "Say the word __", portraying each of seven emotions (Anger, Disgust, Fear, Happy, Surprise, Sad, Neutral). Both actresses, aged 26 and 64 years old, speak English as their mother tongue and have musical training. This results in a total of 2800 utterances in the database.

The file naming convention of the TESS database is separated into three parts identifier with an underscore, Actress_word_emotion.wav. The first identifier states each audio file's younger (YAF) actress and the older (OAF) actress. After the actress identifier code, it is followed with the word spoken in the phase by each actress. The last identifier stated the emotion expressed by the actress.

An example of an audio file would be "OAF_witch_angry.wav".

2.4. Preprocessing

Preprocessing is the step after collecting data that will be used for the training of the SER classifier. Different preprocessing techniques help with the normalization and feature extraction of the audio data. This will significantly improve the recognition of the classifier during the training phase. This step also adds the augmentation of the audio to generalize the data and help the classifier recognize possible real-life speech with techniques such as adding noise, increasing or decreasing pitch and tone, and stretching the audio.

2.4.1. Framing

Signal framing is a process of segmenting continuous speech signals into a fixed segment length. As sound is a non-stationary signal, we are unable to do spectral-domain analysis on sound. Hence, apply such analysis techniques to sound, framing is needed. By framing the signals into a sufficiently short period of 20 to 30 milliseconds, we can consider each frame in a quasi-stationary state, depending on how fast the spectral content signal ranges. In this state, the signal frames can be approximated while retaining the emotional information, allowing the local features to be obtained. However, this introduces a problem when applying Fourier transform (From time to frequency domain), spectral leakage.

2.4.2. Windowing

Windowing is a necessary method to solve the spectral leakage problem. Spectral leakage happens when there is non-integer number of periods in frames during framing, which is very likely. The endpoints of these frames are discontinuous. These discontinuities will appear as high-frequency components that are not presented in the original signal.

Applying the windowing method, the endpoints of each frame are smoothened out to minimize the leakage spike. The most frequently used windowing function is the Hann window below, where w(k) is the formula, k is each sample, s(k) is each frame and $s_w(k)$ is the output frame:

$$w(k) = 0.5 \cdot (1 - \cos(\frac{2\pi k}{K-1})), k = 1...K$$

$$s_w(k) = s(k) \cdot w(k), k = 1...K$$

Figure 4. below shows how the signal will look like after applying the Hann window function.

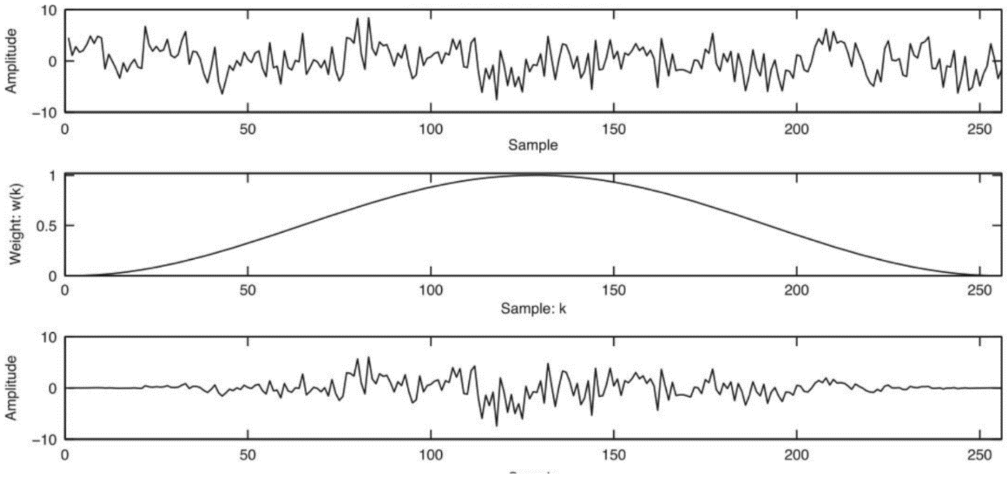


Figure 4. Hann Windowing method. The 1st signal wave is the original signal, while the middle signal wave is the Hann Windowing function. The last signal wave is the output after applying the function.

After applying the windowing function to the frames, the signals will have another issue: losing extractable features at the endpoints of the frames. This can be easily solved by overlapping the frames when applying the framing method to minimize signal loss. The amount of samples overlapped is also known as hop length.

2.4.3. Noise, Noise Reduction, Stretch, Pitch

Since the accuracy of the classifier is largely dependent on the amount and diversity of the dataset during training, often than not, the dataset is usually insufficient. This will result in the classifier being unable to recognize and learn from the data. Although collecting more data will solve this problem, the time and effort to collect data is extensive work. Another way to counter the problem of limited data is to apply different transformations to the given dataset, synthesizing new data. This will also help to generalize the classifier during training.

A method to simulate background noise or muffled audio is by adding white noise to the data. Noise injection is simply adding random value, up to a threshold, into the data by using NumPy. This will result in a signal with mild static noise.

Depending on the objective and data, the clarity of the audio might be critical for training. Noise reduction is a method to minimize the background noise by applying a Wiener filter on the audio. It subtracts an estimation of the noise spectrum from the original, minimizing the expected distortion between clean and estimated signals.

To mimic the various speed of human speech, the time series of audio data can be sped up or stretched by a fixed rate. This will also affect the pitch of audio data. In terms of musical notes, doubling the speed will raise the pitch of each note by an octave.

2.5.Features

In machine learning, feature selections are essential in characterizing different classes in classification models. A good set of features fed for training will increase the recognition rate of the SER system. Different speech features represent different information. Unfortunately, there are no certain features or number of features for precise classification but to test each of them.

There are various levels of abstraction in categorizing audio features[26] :

- High Level – These features are understood and enjoyed by listeners. It includes instrumentation, key, chords, rhythm, melody, genre, harmony, mood, etc.

- Mid Level – These are the features we can perceive, such as pitch, beat, fluctuation patterns, MFCC, STFT. These are an aggregation of low-level features.

- Low Level – These are statistical features that are extracted from audio data that make sense to the machine but not humans. Examples include Zero-Crossing Rate, Spectral Centroid, Spectral Rolloff, Spectral Flux, Root Mean Square Value, Amplitude Envelope, Energy, etc.

There are also different signal domains in audio features, which includes :

- Time Domain – Extracted from waveform from raw audio, such as Zero-Crossing Rate, Root Mean Square Value, Amplitude Envelope, Energy, etc.

- Frequency Domain – These focus on the frequency components of the signal, which are generally converted from the time domain using Fourier Transform. Examples are Spectral Centroid, Spectral Rolloff, Spectral Flux, Band Energy Ratio, etc.

- Time-Frequency Representation – These features combine both time and frequency parts of the signals, which can be obtained by applying Short-Time Fourier Transform (STFT) on the time-domain waveform. Examples are Spectrogram, Mel-spectrogram, Constant-Q Transform, etc.

In this project, features that will be extracted for training include Zero Crossing Rate (ZCR), Root Mean Square Value (RMS), Spectral Centroid, Spectral Rolloff, Mel-Frequency Cepstral Coefficients and Chroma Short-Time Fourier Transform.

2.5.1. Zero-Crossing Rate

Zero-Crossing Rate (ZCR) is a simple method of measuring the number of times a waveform crosses the horizontal time axis. It is the rate at which the signal change from positive to zero to negative, and vice versa. It is widely used in speech recognition, monophonic pitch estimation and music information retrieval, defined as

$$zcr = \frac{1}{T-1}\sum_{t=1}^{T-1} 1_{\mathbb{R}_{<0}}(s_t s_{t-1})$$

where s is a signal of length T and $1_{R<0}$ is an indicator function. In some cases, only the positive-going or negative-going crossings are counted. ZCR can be used as a pitch detection algorithm for monophonic tonal signals or be used for Voice Activity Detection, detecting if human speech is in audio data.

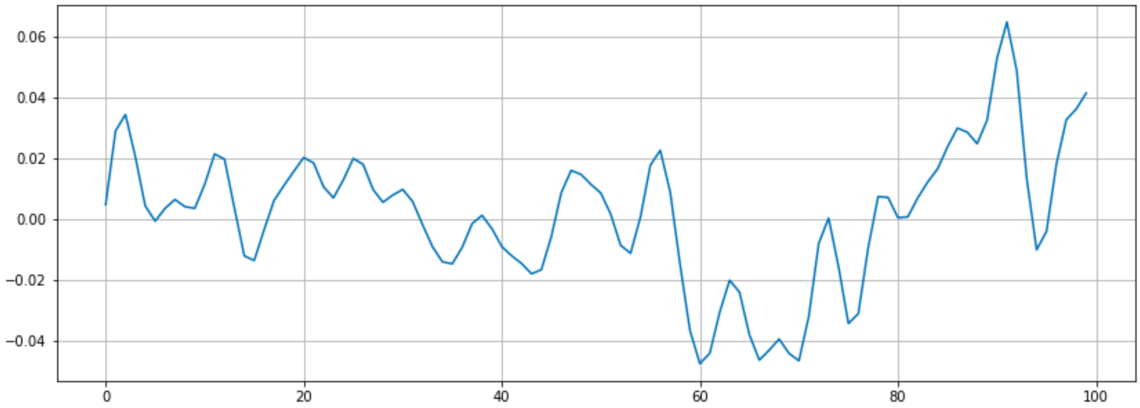


Figure 5. An example of Zero Crossing Rate, which has a value of 16.

2.5.2. Root Mean Square Value

Root Mean Square Value or Root Mean Square Energy (RMS) is often used to measure perceived loudness. It is defined as the average power output over a period of time. The higher the RMS value, the louder the perceived loudness would be. It is calculated by

$$RMS_t = \sqrt{\frac{1}{K}\cdot\sum_{k=t\cdot K}^{(t+1)\cdot K-1} s(k)^2}$$

where *s(k)* is the amplitude of k^{th} sample in the time domain, and k is the number of samples in each frame. RMS value is used mainly for audio segmentation, beat detection or music genre classification tasks.

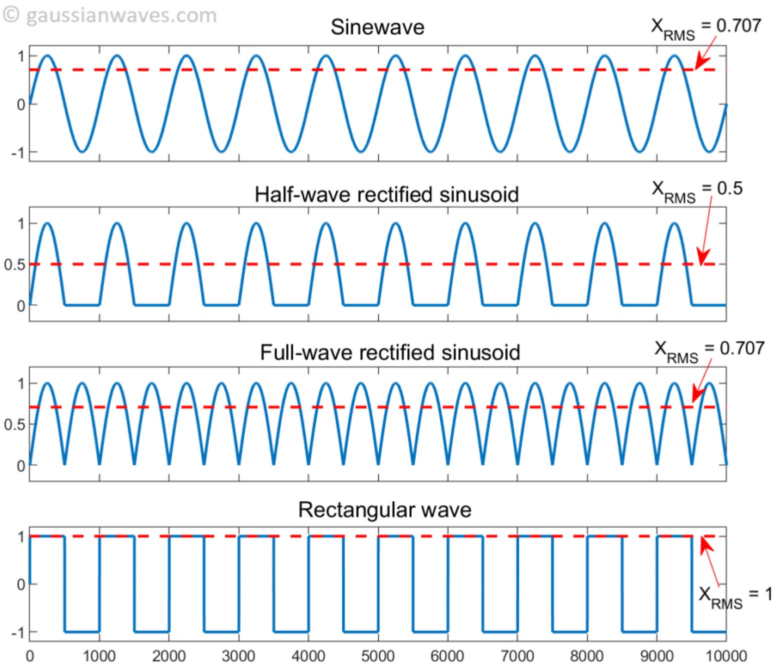


Figure 6. Examples of RMS values of well-known signals

2.5.3. Spectral Centroid

Spectral Centroid is a measurement to characterize a spectrum. It is an indication of which frequency the energy of a spectrum is centred upon or where the centre of mass of the spectrum is located. It maps into a prominent timbral feature called "brightness of sound", measuring sound sharpness in terms of strength in frequency energy. It is calculated by

$$f_c = \frac{\sum_k S(k)f(k)}{\sum_k S(k)}$$

where *s(k)* is the spectral magnitude at frequency bin *k*, and *f(K)* is the frequency at bin *k*. Spectral Centroid is widely used as an automatic measure of musical timbre.

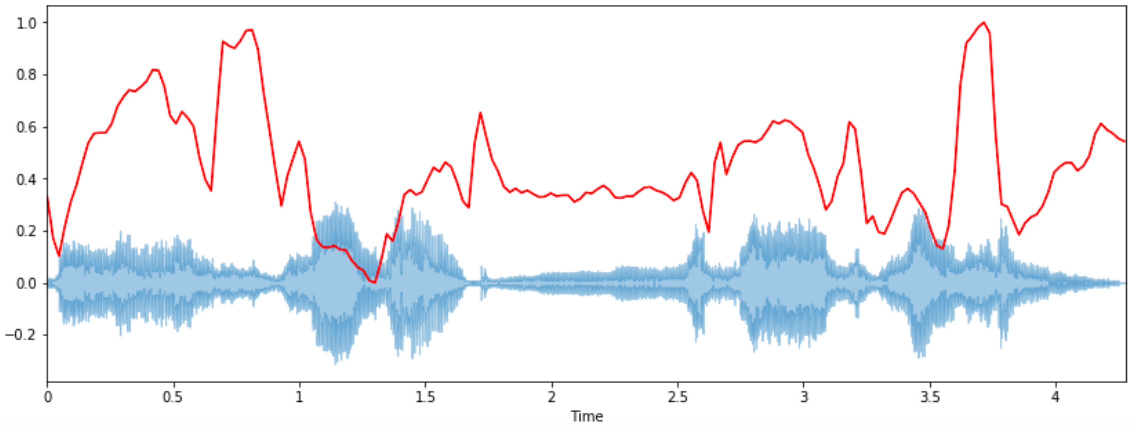


Figure 7. An example of Spectral Centroid, in red.

2.5.4. Spectral Rolloff

Measuring the shape of the signal, Spectral Rolloff represents the frequency at which high-frequency declines to zero. It is characterized as the action of a specific type of filter that is outlined to roll off the frequencies outside a specific range. The Spectral Rolloff point is the fraction of bins in the power spectrum at which 85% of the power is at lower frequencies and is used to calculate the maximum and minimum by setting up a value close to 1 and 0.

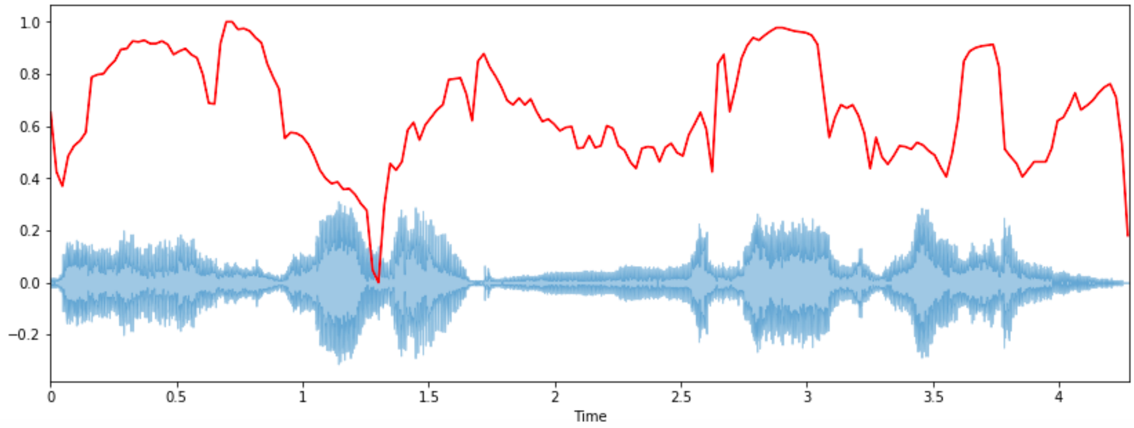


Figure 8. An example of Spectral Rolloff, in red.

2.5.5. MelSpectrogram

A spectrogram is computed by using Short-Time Fourier Transform, a method of fast Fourier Transform on several overlapping windowed segments of the signal. This is a way to visualize the signal's loudness, which is amplitude, as it varies over time at different frequencies. The y-axis of a spectrogram is converted to a log scale, while the colour dimension is converted to decibels. This is because humans are only able to perceive a small and concentrated range of frequencies and amplitude.

Since humans are unable to perceive frequencies on a linear scale, which means higher frequencies are more challenging for us to differentiate, a unit of pitch is used to allow equal distanced pitch sound equally apart to the listener. This unit is also known as a mel scale.

By converting the spectrogram using a mel scale, a MelSpectrogram is formed.

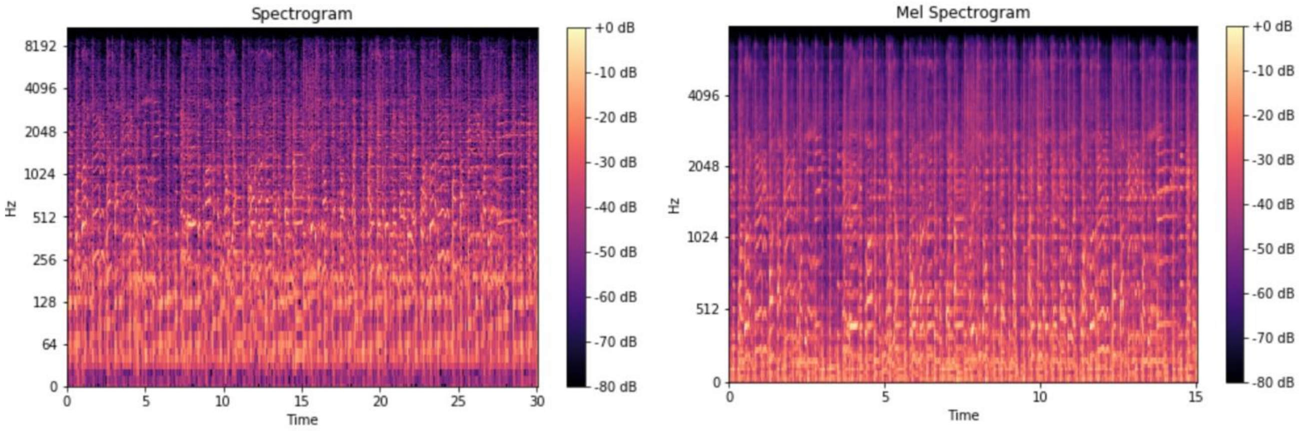


Figure 9. An example of a Spectrogram on the left and Mel Spectrogram on the right, after applying the mel scale[27]

2.5.6. Mel-Frequency Cepstral Coefficients

One of the important features in speech recognition and music similarity, Mel-Frequency Cepstral Coefficients (MFCC) represents the envelope of the power spectrum of an audio frame. It is the information of the rate of change in spectral bands given by its cepstrum. A cepstrum is a log of a spectrum of the time signal. The result is neither in the time or frequency domain but in a quefrency domain. MFCC concisely describes the spectral envelope's overall shape using a small set of features, usually in 10 or 20. This is derived by mapping the Fourier transformed signal onto the Mel scale using cosine or triangle overlapping windows. Taking the logs of the powers at each of the Mel frequencies and discrete cosine transform of the Mel log powers will give the amplitude of a spectrum. This amplitude list is MFCC.

MFCCs are useful features for deep learning models, commonly used in speech recognition systems and music information retrieval applications such as genre classification, audio similarity, etc.

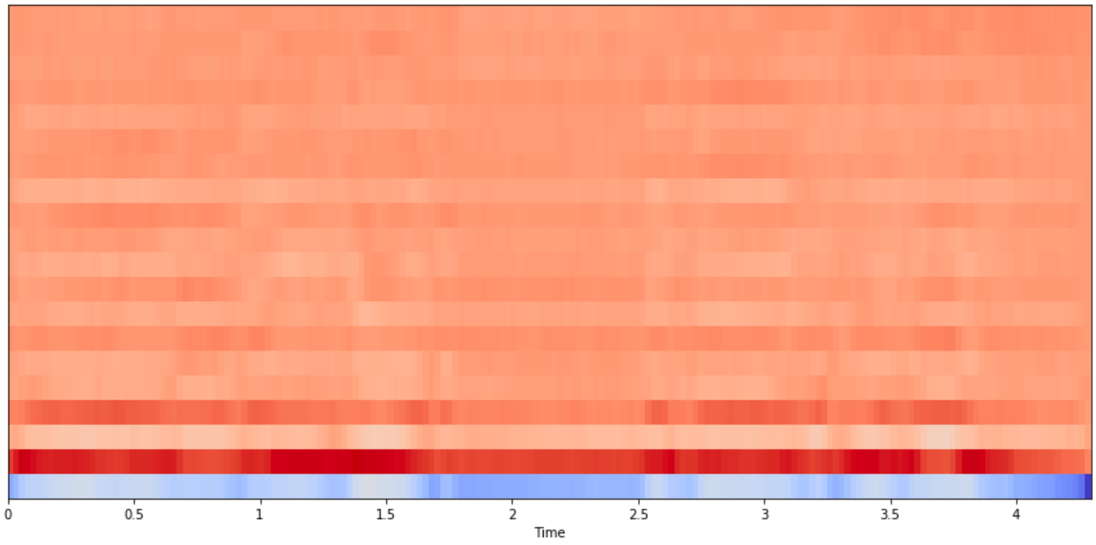


Figure 10. An example of MFCC across time.

2.5.7. Chroma Short-Time Fourier Transform

Chroma Short-Time Fourier Transform (STFT) is a type of chromagram that represents the pitches in audio data. It consists of 12 element feature vectors corresponding to 12 different pitch classes, indicating how much energy of each pitch class is represented in the signal. It is computed from the power spectrogram of the audio data, and it describes a similarity measure between music pieces.

In pitch classes, a pitch can be separated into two parts: tone height and chroma. The tone height refers to the octave number of the pitch, while the chroma represents the respective pitch spelling attribute. Each pitch class is identified as the set of all pitches that shares the same chroma. This feature aggregates all spectral information related to a given pitch class into a single coefficient.

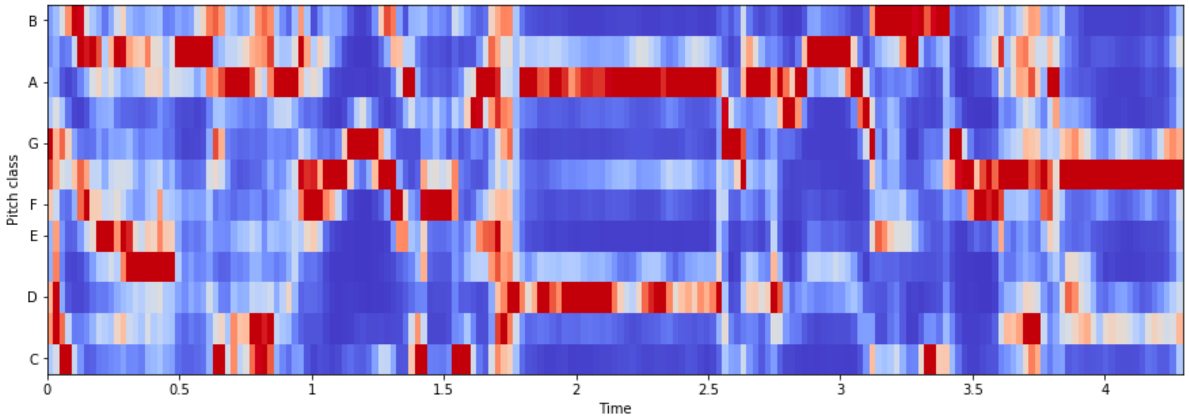


Figure 11. An example of Chroma-SFTF, with pitch class against time.

2.6.Classifiers

Many different machine learning algorithms are used in the SER system, including traditional classifiers and deep learning algorithms. Just like the features, there is no generally accepted machine learning algorithm to be used for SER. Depending on the system's objective and method of preprocessing the data, different classifiers can be selected for the prediction.

Common traditional classifiers include Hidden Markov Model, Support Vector Machines, Artificial Neural Networks, and Gaussian Mixture Model. Other methods can also be based on Decision Trees, K-Means, K-Nearest Neighbour or Naive Bayes Classifiers. Ensembling method could be implemented in SER models to obtain a better result.

Deep learning classifiers are primarily based on Artificial Neural Networks, therefore referred to as deep neural networks. The term "deep" refers to the hidden layers in the neural network, which can reach up to hundreds of layers in a model, while a traditional neural network consists of only two or three hidden layers. Over the years, deep learning algorithms have surpassed the traditional machine learning algorithms in terms of performance in many problems, including SER. Majority of the deep learning algorithms used in the SER domain consist of Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), Long-Short Term Memory (LSTM) and Gated Recurrent Units (GRU), etc.

2.6.1. Hidden Markov Model

One of the most commonly used traditional methods for speech recognition, the Hidden Markov Model (HMM) has been successfully recognizing emotions well. HMM uses the Markov Property, which refers to how the current state at time *t* depends on the previous state at time *t-1*. The output of each current state is observed from the emission probability of the previous state. The term "hidden" implies that the process of those states is unobservable. Modern general-purpose speech emotion recognition systems use HMM because speech signal is viewed as a piecewise stationary signal that can be approximated as a stationary process.

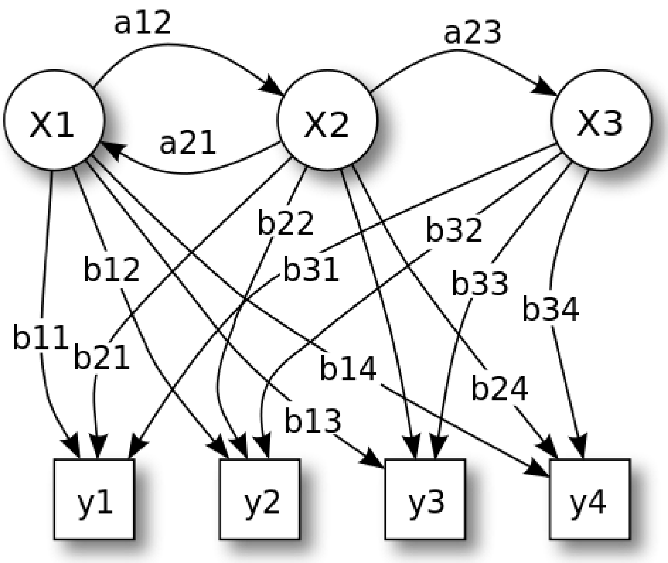


Figure 12. An example of probabilistic parameters of HMM: X – state,y – possible observations, a – state transition probabilities, b – output probabilities[28]

2.6.2. Support Vector Machine

Support Vector Machine (SVM) is another traditional supervised learning algorithm that uses the concept of finding an optimal hyperplane for linearly separable patterns given the training data. This hyperplane has the maximum margin between data points of the classes. However, it is often that the classes are not linearly separable in a space. Therefore the original finite-dimensional space can be mapped into a higher-dimensional space by using a kernel function.

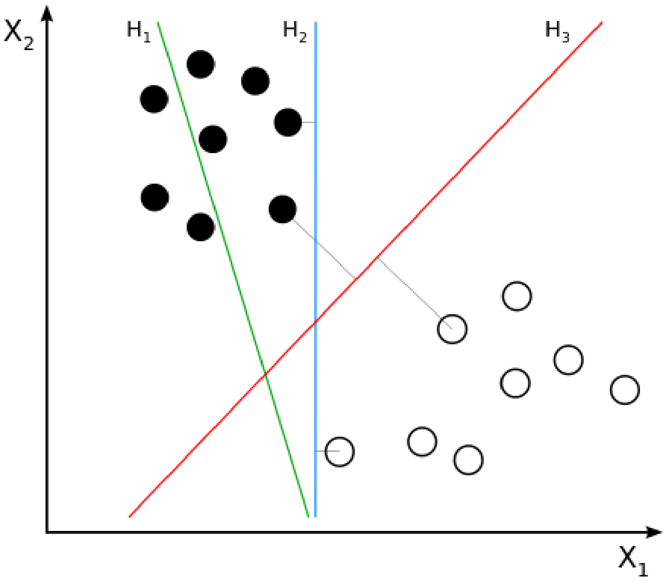


Figure 13. An example of a SVM in a graph. H1 margin does not separate the classes, H2 does with a small margin, H3 separates with a maximal margin[28]

2.6.3. Recurrent Neural Networks

Recurrent Neural Network is a class of artificial neural networks that are specialized in processing sequential data. Derived from feedforward neural networks, RNN uses internal state memory to remember the received input data to precisely predict what's next. This is especially useful for types of sequential data such as time series, video, text and speech.

RNN works by taking in sequential data and feed into a unit. This unit will produce an output of the data information and is fed into the next unit together with the next input. This retaining information and the input will then produce the following output data information, again to be fed to the next unit with the following input. This recurring step will form a loop until the last data point is processed. The retaining of important information in each step works very well in recognizing data patterns but suffers from a short term memory, also known as vanishing gradient. This is due to the nature of backpropagation algorithm used to train and optimize neural networks. The gradients at the end, which are used to adjust the neural network's weights, will exponentially shrink as it back propagates through each step. This will cause the earlier steps not to learn anything at all. This means there is a possibility that earlier data information is not considered during predictions and might struggle in understanding the full context.

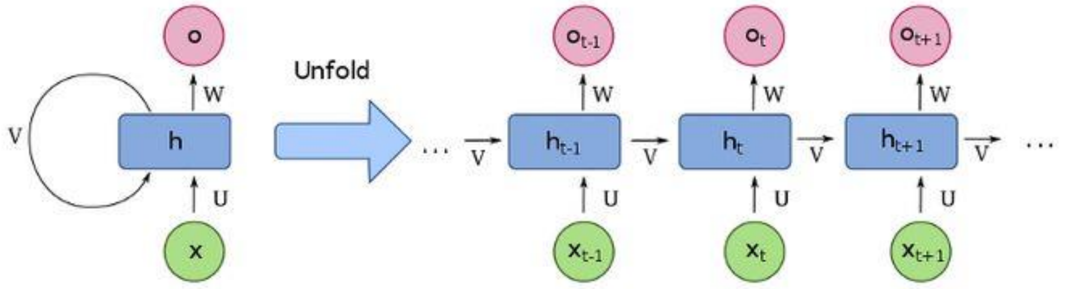


Figure 14. Basic Recurrent Neural Network unfolded[28]

Two special methods built on RNN were created to mitigate the vanishing gradient problem: Long-Short Term Memory (LSTM) and Gated Recurrent Units (GRU). Both functions essentially work just like RNN with additional mechanism call gates capable of learning long-term dependencies. These gates can learn what information to add or remove to the hidden state.

Different from RNN, LSTM introduced a long term and short term memory state. These two cell states consist of three gates, which are "forget gate", "input gate", and "output gate". The "forget gate" controls if the data is to be discarded from memory by applying a sigmoid function, while the "input gate" controls if the data is to be added to the memory by applying the sigmoid and tangent function. The "output gate" will then produce the output information by multiplying the sigmoid function of the input data and the tangent function of the "input gate" memory. This process will be used in each recurring step for each data fed and remember the important data points for the sequential data.

GRU is considered a modified version of LSTM where it combines both long term and short term memory into one state. This state consists of two gates, which are "update gate" and "reset gate". The function of "update gate" is to retain an amount of pass memory, while the function of "reset gate" is to forget an amount of pass memory. Both gates use the input data with different weights depending on how much data to reset or update and are fed to a sigmoid function before using it to produce a hidden state and the next output. This led to an overall more efficient computation wise compared to LSTM.

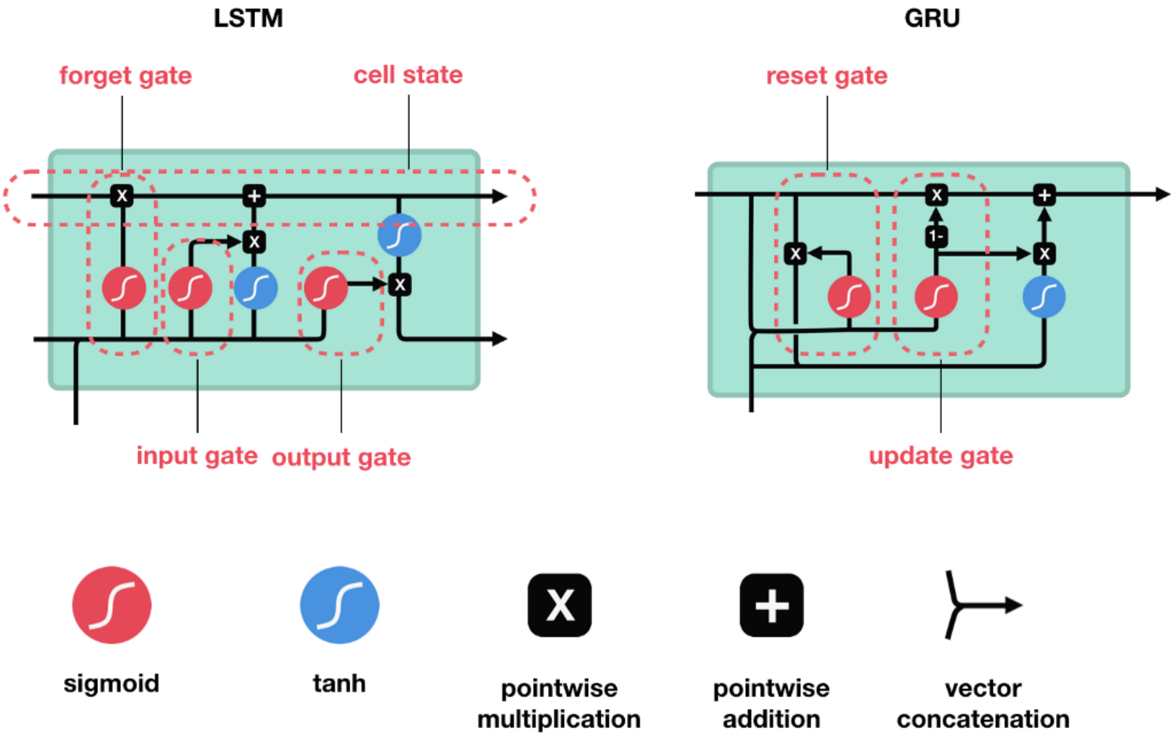


Figure 15. Long-Short Term Memory and Gated Recurrent Unit[29]

2.6.4. Convolutional Neural Network

Another class of artificial neural networks in deep learning, Convolutional Neural Network (CNN) is commonly used to analyze visual imagery. It uses convolution in place of general matrix multiplication in its architecture. It has applications in computer vision, recommender systems [30], image classification, image segmentation, natural language processing[31], and financial time series[32], etc.

Like a feedforward neural network, a basic CNN architecture consists of an input layer, multiple hidden layers and an output layer. The difference is, CNN has hidden layers that perform convolutions, typically a dot product of the input matrix. Each hidden layer contains an arbitrary number of neurons with activation functions such as ReLU, Sigmoid or Softmax. These layers can be stacked with other layers, such as pooling and fully connected layers, before going through an output layer for classification. There are also important parameters that can be added to the architecture, allowing better performance in the model, which are the dropout and normalization layers.

As shown in Figure 16, the CNN architecture is split into two main parts. The first part of which a convolution tool is used to separate and identify various features in the data is called Feature Extraction. The next part consists of fully connected layers that utilize the extraction process's output and predict the class in the output layer.

In this project, different CNN architectures will be experimented with to find the optimal accuracy for prediction.

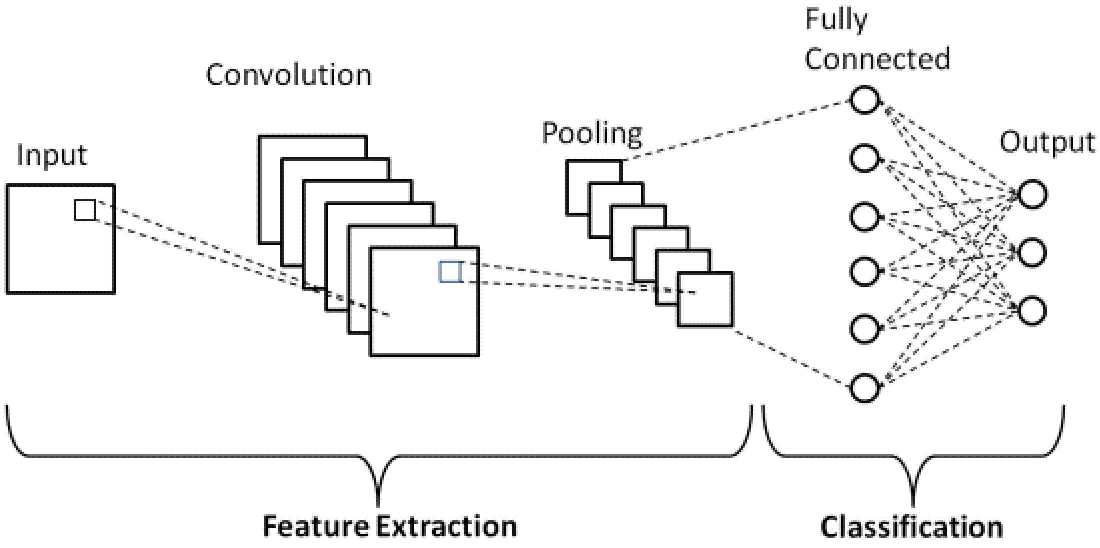


Figure 16. An outline of CNN architecture.[33]

2.6.4.1. Convolutional Layer

This layer is the primary layer that helps to extract various features from the data. In this layer, the mathematical operation of convolution is performed between the matrix of input data and a fixed filter size of MxM. Using the dot product, this filter is slid across the input matrix to produce an output matrix size not larger than the original input matrix. This output matrix is called the Feature map, which gives information about the input data. Afterwhich, the feature map can be fed into another convolutional layer or other layers to learn more data features.

2.6.4.2. Pooling Layer

Commonly, this layer is followed by a convolutional layer, which aims to decrease the size of the convolved feature map to reduce computational costs. Depending on the pooling method and size, the connection is decreased between layers and independently operates on each feature map. The pooling layer usually serves as the bridge between the convolutional layer and the fully connected layer.

- MaxPooling – An operation that calculates the maximum value in each patch of the feature map. The results are downsampled, which highlights the most present feature in that patch.

- AveragePooling – An operation that calculates the average value in each patch of the feature map. The results are downsampled, which summarizes the feature in that patch.

- GlobalPooling – An aggressive way to summarize the presence of a feature in the matrix. It is to downsample the feature map into a single value. Both maximum and average can be applied in GlobalPooling.

2.6.4.3. Dropout Layer

The dropout layer is usually used to solve an overfitting problem in a model. When a model is overfitted during training, it is unable to predict accuracy on unknown data. This layer is utilized where a percentage of the hidden neurons in a layer is randomly unactivated during training, reducing the model's size. This percentage is arbitrary and is commonly between 0.1 to 0.3, depending on the data itself.

2.6.4.4. Normalization Layer

This method is used to normalize each input feature by maintaining the contribution of every feature, reducing Internal Covariate Shift while increasing the convergence rate of the algorithm. This will result in a shorter training time for the model.

A covariate shift is the change in the distribution of the input features, where the dataset is heavily biased to a certain class. By using a Batch Normalization Layer, the inputs will be subtracted by the batch mean and divided by the batch standard deviation. The data will then be standardized before feeding it to the next layer.

2.6.4.5. Fully Connected Layer

The Fully Connected layer, which consists of neurons with activation functions, weights and biases, is connected before an output layer forming the last few layers in a CNN architecture. It takes in a flattened vector from the previous layers in the feature extraction process and computes the mathematical functions before producing a classification output. This layer is also known as the Dense Layer and can be viewed as Multilayer Perceptron.

2.7. Supporting Modalities

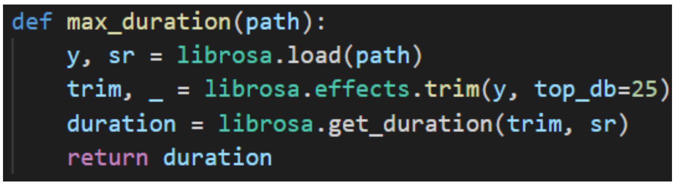
Several technologies can be incorporated to make a better prediction in the SER system. Although standalone systems can achieve high recognition rates, it has not yet successfully recognize emotion fully. With supporting modalities, it can enhance the power of the recognition system. Some supporting modalities include visual signals, physiological signals, word recognition, linguistic features, brain signals, etc.

In this project, a supporting modality of a visual signal is added together with the SER system to have a realistic sense in a therapy session. It is an emotional recognition of facial expression from a video feed through a camera. This combination of audio and visual data can be used in physical sessions to better capture the patient's emotion in therapy or to use it in a virtual context where the therapy session is being held online, which is especially helpful during this period of time.

3. Methodology

3.1. Dataset preparation

In "path.py", the path of four different datasets is used to record, sort and filter different audio files corresponding to their emotion classes. Each dataset is iterated to record which emotion the file represents and the duration of each audio file. The duration is used for further comparison during data preprocessing. The function "max_duration" uses the Librosa library to do some trimming before finding the duration. The trimming is done to the silence part of the audio file as it does not contain any information on what we need. We set any decibel below 25 to be silent and to be trimmed off.



```python
def max_duration(path):
    y, sr = librosa.load(path)
    trim, _ = librosa.effects.trim(y, top_db=25)
    duration = librosa.get_duration(trim, sr)
    return duration
```

Snippet 1. Trimming of the audio file before getting the duration of the audio

3.2. Data Preprocessing

In this project, most of the preprocessing and feature extraction methods are applied using the Librosa library. From the four datasets used, all audio files have different time lengths. Therefore segmentation must be done before feeding the data for training. In Figure 17, a view of the whole dataset before preprocessing, we can see that we have an unrelated 'calm' class, and there is also some imbalance data in the 'surprise' class.

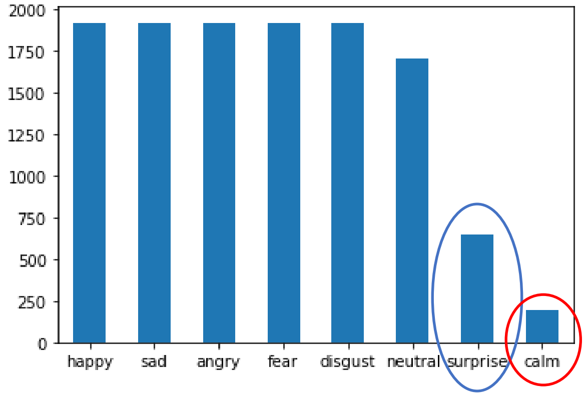


Figure 17. All classes in the combined dataset. An extra class in red circle, and imbalance data in blue circle

There are some techniques that can fix an imbalanced dataset, such as oversampling and undersampling. However, the amount of data points for the 'surprise' class seems enough for training. Hence we will only drop the 'calm' class for the training process.

Next, we have to check the lengths of the audio files. In Figure 18, we can see that rounding off to the nearest integer, most audio data is dominated at 2 seconds long. Since audio data with 5 seconds or longer consist of a small percentage in the dataset, we only consider audio data less than 4.5 seconds.

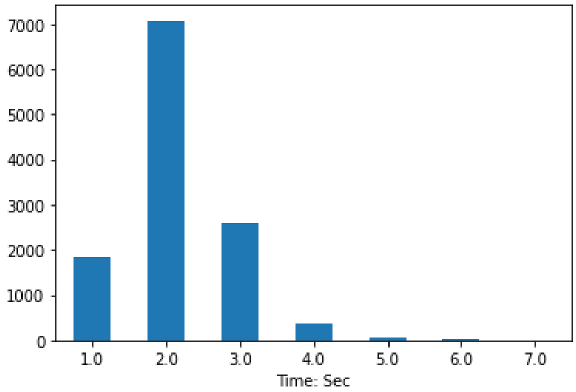
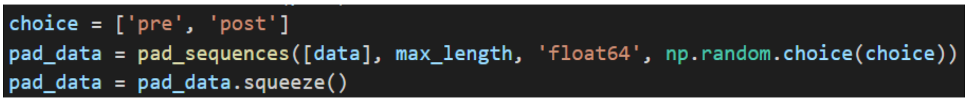


Figure 18. Counts of each audio data length in seconds (nearest integer)

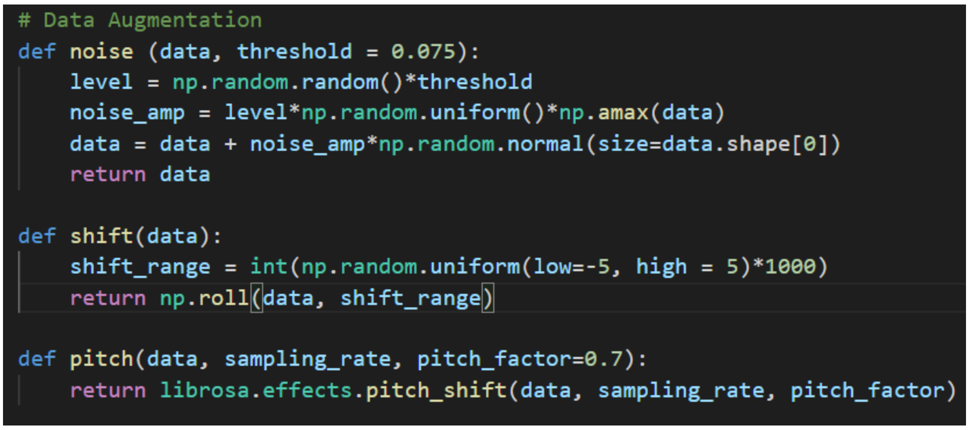
To ensure the same length of feature extraction in each audio file, padding must be done to either the start or the end of the data. Keras.preprocessing library offers pad_sequence() to pad the data to the maximum length we found in the dataset. A random choice of pre or post is also added to randomized the data padding.



```
choice = ['pre', 'post']
pad_data = pad_sequences([data], max_length, 'float64', np.random.choice(choice))
pad_data = pad_data.squeeze()
```

Snippet 2. Data padding for each audio loaded.

Each audio data will also go through data augmentation individually, adding random noise up to a threshold of 0.075, a random pitch shift of a factor of 0.7 and a random audio shift.

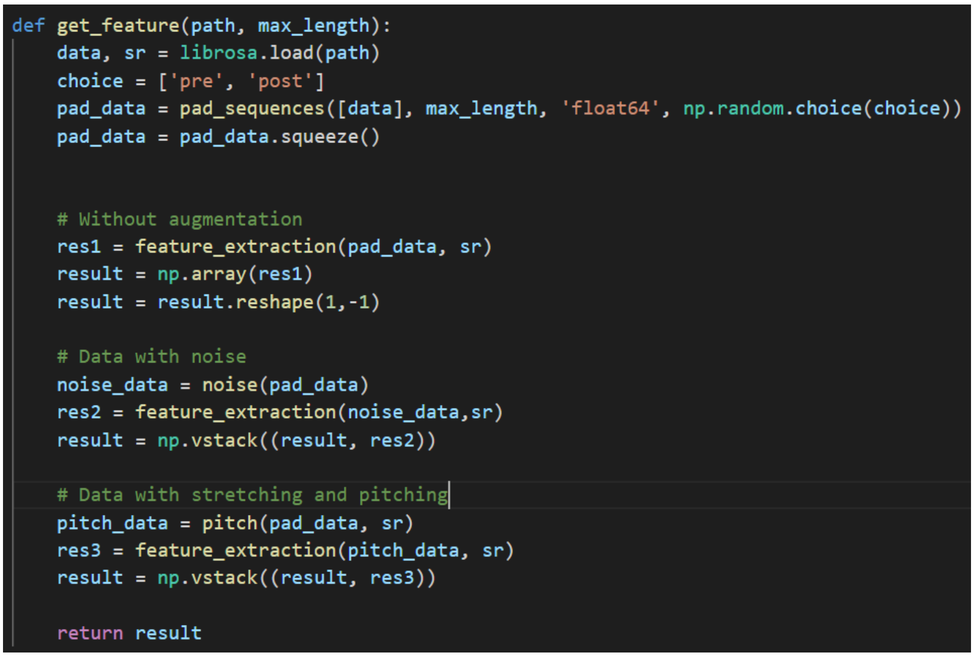


```
# Data Augmentation
def noise (data, threshold = 0.075):
    level = np.random.random()*threshold
    noise_amp = level*np.random.uniform()*np.amax(data)
    data = data + noise_amp*np.random.normal(size=data.shape[0])
    return data

def shift(data):
    shift_range = int(np.random.uniform(low=-5, high = 5)*1000)
    return np.roll(data, shift_range)

def pitch(data, sampling_rate, pitch_factor=0.7):
    return librosa.effects.pitch_shift(data, sampling_rate, pitch_factor)
```

Snippet 3. The different data augmentation for this project.

3.3.Feature Extraction

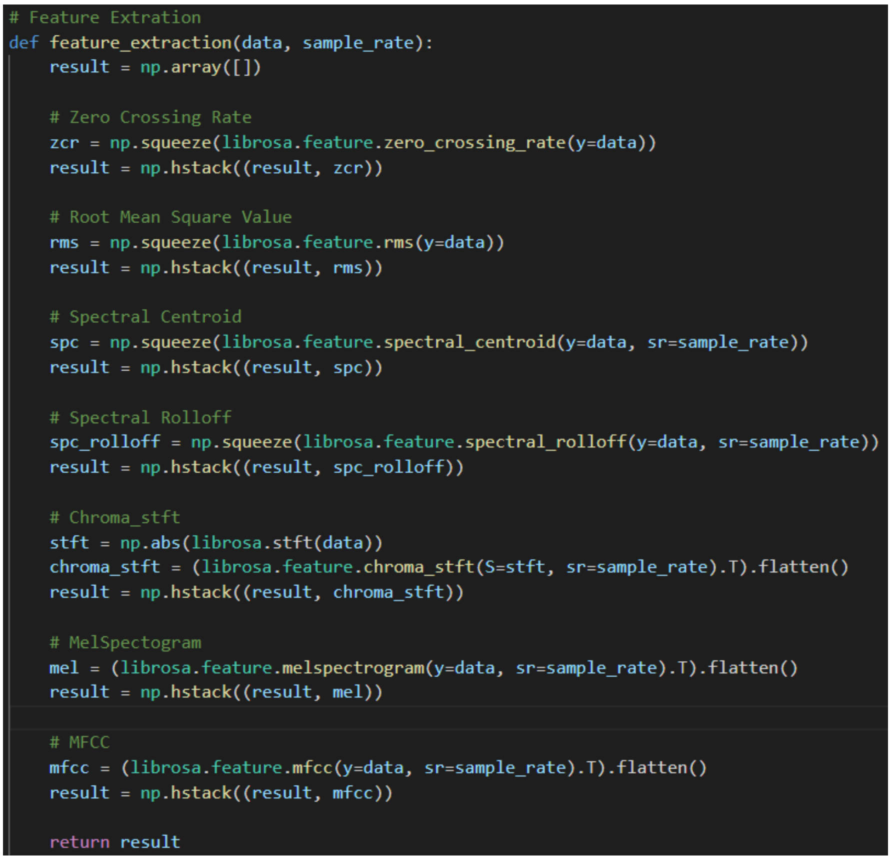
As the dataset is huge, feature extraction will be done and saved into an excel file ("features.csv") before feeding it to the model for training. In "feature.py", each audio data's features will be extracted three times: data without any augmentation, random noise added, and random pitch augmented. As shown in Snippet 3 below, the function "get_feature" takes in an audio file path to load the data



```python
def get_feature(path, max_length):
    data, sr = librosa.load(path)
    choice = ['pre', 'post']
    pad_data = pad_sequences([data], max_length, 'float64', np.random.choice(choice))
    pad_data = pad_data.squeeze()


    # Without augmentation
    res1 = feature_extraction(pad_data, sr)
    result = np.array(res1)
    result = result.reshape(1,-1)

    # Data with noise
    noise_data = noise(pad_data)
    res2 = feature_extraction(noise_data,sr)
    result = np.vstack((result, res2))

    # Data with stretching and pitching
    pitch_data = pitch(pad_data, sr)
    res3 = feature_extraction(pitch_data, sr)
    result = np.vstack((result, res3))

    return result
```

Snippet 4. Each audio data loaded will have data augmented before feature extraction.

After every augmentation, each audio is sent to the following function, "feature_extraction", to extract all 7 different features: Zero Crossing Rate, Root Mean Square Value, Spectral Centroid, Spectral Rolloff, Chroma Short-Time Fourier Transform, MelSpectrogram, Mel-Frequency Cepstral Coefficients. All data processed will then be stacked into a data frame before saving it as an excel file.

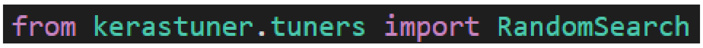


```python
# Feature Extration
def feature_extraction(data, sample_rate):
    result = np.array([])

    # Zero Crossing Rate
    zcr = np.squeeze(librosa.feature.zero_crossing_rate(y=data))
    result = np.hstack((result, zcr))

    # Root Mean Square Value
    rms = np.squeeze(librosa.feature.rms(y=data))
    result = np.hstack((result, rms))

    # Spectral Centroid
    spc = np.squeeze(librosa.feature.spectral_centroid(y=data, sr=sample_rate))
    result = np.hstack((result, spc))

    # Spectral Rolloff
    spc_rolloff = np.squeeze(librosa.feature.spectral_rolloff(y=data, sr=sample_rate))
    result = np.hstack((result, spc_rolloff))

    # Chroma_stft
    stft = np.abs(librosa.stft(data))
    chroma_stft = (librosa.feature.chroma_stft(S=stft, sr=sample_rate).T).flatten()
    result = np.hstack((result, chroma_stft))

    # MelSpectogram
    mel = (librosa.feature.melspectrogram(y=data, sr=sample_rate).T).flatten()
    result = np.hstack((result, mel))

    # MFCC
    mfcc = (librosa.feature.mfcc(y=data, sr=sample_rate).T).flatten()
    result = np.hstack((result, mfcc))

    return result
```

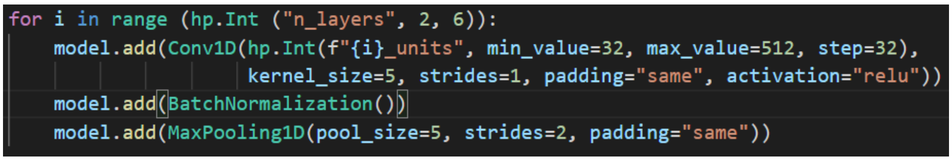
Snippet 5. 7 types of feature extractions.

3.4. Architecture

For this sequential data, we will be using a Convolutional Neural Network architecture to train our model. This architecture uses Conv1D as its convolution layer to apply feature extraction in the training process. As mention above, the number of layers in a CNN architecture is arbitrary. We use an inbuilt Keras library called kerastuner to do a random search on the number of layers and hidden units used.



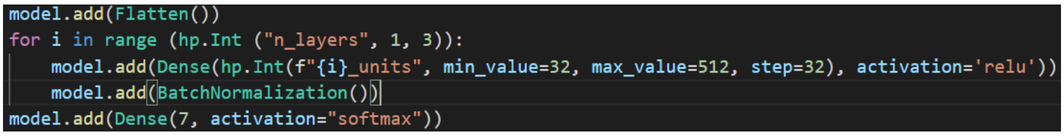
```
from kerastuner.tuners import RandomSearch
```



```
for i in range (hp.Int ("n_layers", 2, 6)):
    model.add(Conv1D(hp.Int(f"{i}_units", min_value=32, max_value=512, step=32),
                     kernel_size=5, strides=1, padding="same", activation="relu"))
    model.add(BatchNormalization())
    model.add(MaxPooling1D(pool_size=5, strides=2, padding="same"))
```

Snippet 6. Parameters to be searched randomly for Conv1D layers.

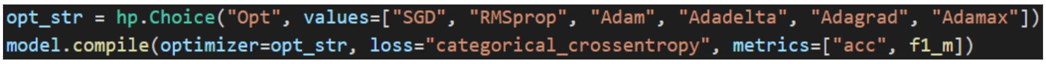
For the classification process, Dense layers are added after flattening the output from the feature extraction process. Similar to the convolutional layer, we do a random search on the number of dense layers and their hidden units used.



```
model.add(Flatten())
for i in range (hp.Int ("n_layers", 1, 3)):
    model.add(Dense(hp.Int(f"{i}_units", min_value=32, max_value=512, step=32), activation='relu'))
    model.add(BatchNormalization())
model.add(Dense(7, activation="softmax"))
```

Snippet 7. Parameters to be searched randomly for Dense layers.

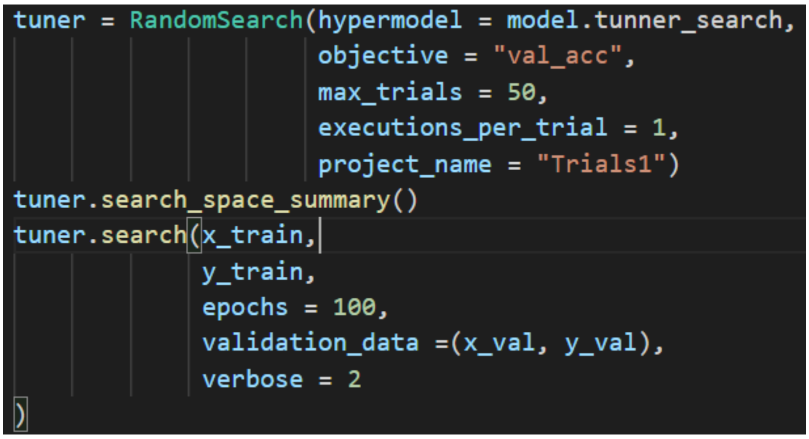
Another parameter that is randomly searched is the optimizer. We searched on the 6 commonly used optimizers, SGD, RMSprop, Adam, Adadelta, Adagrad, Admax, and retained their default settings.



```
opt_str = hp.Choice("Opt", values=["SGD", "RMSprop", "Adam", "Adadelta", "Adagrad", "Adamax"])
model.compile(optimizer=opt_str, loss="categorical_crossentropy", metrics=["acc", f1_m])
```

Snippet 8. Parameters to be searched randomly for the optimizer.

The search is then done with the following code shown in Snippet 9. The hypermodel is taken in with the objective of maximum validation accuracy. The hyperparameters above are being tested for 50 trials of different combinations.



```
tuner = RandomSearch(hypermodel = model.tunner_search,
                     objective = "val_acc",
                     max_trials = 50,
                     executions_per_trial = 1,
                     project_name = "Trials1")
tuner.search_space_summary()
tuner.search(x_train,
             y_train,
             epochs = 100,
             validation_data =(x_val, y_val),
             verbose = 2
)
```

Snippet 9. Tuner doing a RandomSearch on the hyperparameters.

The results of this hyperparameter search return an optimum of 4 Conv1D layers, 3 Dense layers, and an optimizer of Adagrad. However, after some experimentation of the model, it produces overfitting results against the validation data. To solve this problem, dropout layers and kernel regularizer is added.

Regularizers are penalties applied to the layer during optimization. These penalties are added to the loss function that the model optimizes. There are two different types of regularizers called L1 and L2. Both shrink the coefficients differently to avoid overfitting. L1 inclined to shrink the coefficients to zero while L2 shrinks the coefficients evenly. Therefore, L1 is usually useful for feature selection since any variables can be dropped associated with its coefficients going to zero. On the other hand, L2 is suitable for codependent features as they tend to increase coefficient variance, making the coefficients unstable. In this project, the L2 regularizer is used since we do not wish to make any feature selection of sort. The function of the L2 regularizer, also known as weight decay, can be written as *R(W),* the sum of squares of the weight matrix *W*.

$$R(W) = \sum_i \sum_j W_{i,j}^2$$

The final model architecture after experimentation is as following:

- Conv1D- units:1024 / kernel size:5
- BatchNorm
- MaxPooling1D- pool size:5 / stride:2
- Conv1D- units:512 / kernel size:5
- BatchNorm
- MaxPooling1D- pool size:5 / stride:2
- Dropout- rate:0.3
- Conv1D- units:128 / kernel size:5
- BatchNorm
- MaxPooling1D- pool size:5 / stride:2
- Dropout- rate:0.2
- Conv1D- units:64 / kernel size:3
- BatchNorm
- MaxPooling1D- pool size:5 / stride:2
- Conv1D- units:32 / kernel size:3
- BatchNorm
- MaxPooling1D- pool size:3 / stride:2
- Flatten
- Dense- units:512
- BatchNorm
- Dense- units:128
- BatchNorm
- Dense- units:64
- BatchNorm
- Dense- units:7 / Softmax output

The shape of training data is given as input in the 1st layer. A kernel size of 5 for the 1st three conv1D layers and 3 for the next two layers are used to cover the whole data size. Same padding is used to ensure the data is trained at the same length throughout the model. Max pooling of size 5 for the 1st four layers and size 3 for the last layer with stride two is used to summarize the features. Three fully connected layers of 512, 128 and 64 units are connected after the output of feature extraction flattened. All layers have a batch normalization layer applied after. A 7-way classification final layer with softmax activation is then applied at the end of the model.

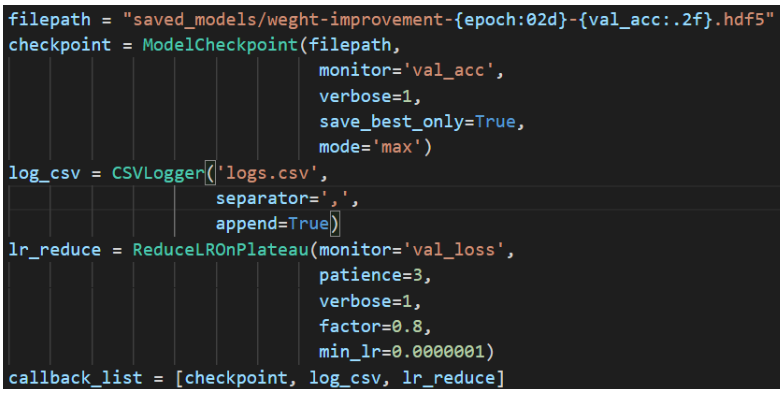
3.5.Training

Importing "features.csv", the feature data is split into 3 sets using the train test split from sklearrn library—a training set of 70%, a validation set of 20% and a testing set of 10%. This is controlled with a seed value of 42.

A list of callbacks is also used to record and save the best model since multiple runs are needed for this extensive data file. The list includes:

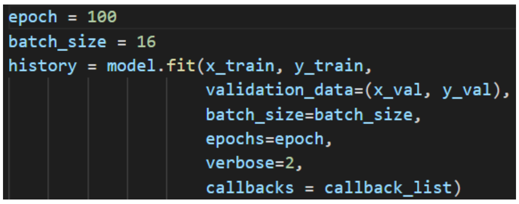
- ModelCheckpoint – monitor: validation accuracy / save best only: True
  - ➢ This callback function is used to save the model and weights at a certain interval to preserve the best model or to resume training from the saved state
- ReduceLROnPlateau – monitor: validation loss / patience: 3 / factor: 0.8 / min lr: 0.0000001
  - ➢ This callback function is used to reduce the learning rate when the monitored value stops improving. This will solve the problem of the model reaching a plateau, slowing down the training process.
- CSVLogger – separator: "," / append: True
  - ➢ This callback function saves all epoch results to a CSV file, including events like "loss", "acc", or other metrics.

The training data is then fed to the model for training with 100 epochs and a batch size of 16.



```
filepath = "saved_models/weght-improvement-{epoch:02d}-{val_acc:.2f}.hdf5"
checkpoint = ModelCheckpoint(filepath,
                              monitor='val_acc',
                              verbose=1,
                              save_best_only=True,
                              mode='max')
log_csv = CSVLogger('logs.csv',
                    separator=',',
                    append=True)
lr_reduce = ReduceLROnPlateau(monitor='val_loss',
                              patience=3,
                              verbose=1,
                              factor=0.8,
                              min_lr=0.0000001)
callback_list = [checkpoint, log_csv, lr_reduce]
```

Snippet 10. Callback list used.



```
epoch = 100
batch_size = 16
history = model.fit(x_train, y_train,
                    validation_data=(x_val, y_val),
                    batch_size=batch_size,
                    epochs=epoch,
                    verbose=2,
                    callbacks = callback_list)
```

Snippet 11. Feeding training data to the model for training.

4. Results

The results of the model came back with an accuracy of 0.61~ at its peak. It is calculated with 3 other metrics, namely Precision, Recall and F1-score.

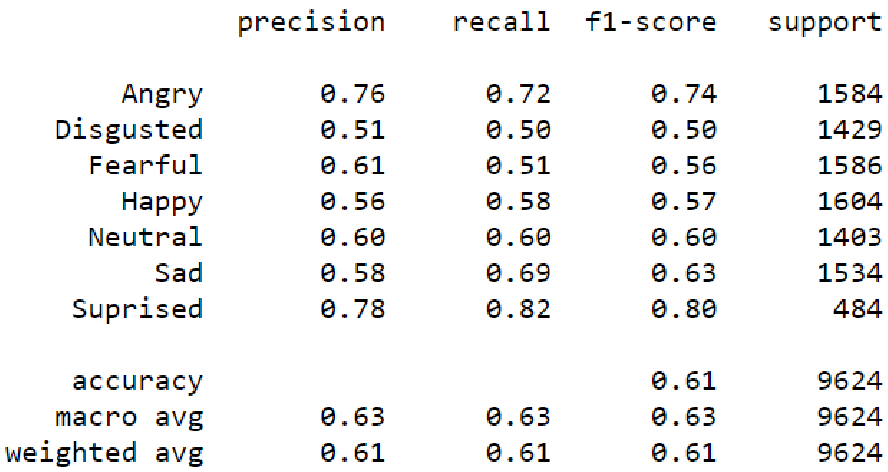
Precision is the ratio between the true positives versus the true positives plus the false positives, while Recall is the measure of the accuracy in identifying true positives, which is the ratio between true positives versus true positives plus false negatives. Related to this, F1-score considers both Precision and Recall, which is the balane of both. This gives a better understanding of how accurate the model CORRECTLY predicts the categories. The closer it is to 1, the better the model performs.

$$\text{Precision} = \frac{True\ Positive}{True\ Positive + False\ Positive}$$

$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} = \frac{\text{TP}}{\text{TP} + \frac{1}{2}(\text{FP} + \text{FN})}$$

$$\text{Recall} = \frac{True\ Positive}{True\ Positive + False\ Negative}$$

Figure 19. The formula for Precision, Recall and F1-score.



```
              precision    recall  f1-score   support

       Angry       0.76      0.72      0.74      1584
   Disgusted       0.51      0.50      0.50      1429
     Fearful       0.61      0.51      0.56      1586
       Happy       0.56      0.58      0.57      1604
     Neutral       0.60      0.60      0.60      1403
         Sad       0.58      0.69      0.63      1534
    Suprised       0.78      0.82      0.80       484

    accuracy                           0.61      9624
   macro avg       0.63      0.63      0.63      9624
weighted avg       0.61      0.61      0.61      9624
```

Figure 20. The classification report of the model after training.

Besides Precision, Recall and F1-score, we can use the confusion matrix to show how well the model does. It is the comparison of the predicted categories versus the actual label, where the identity of this matrix is correctly predicted while the rest are false positives.

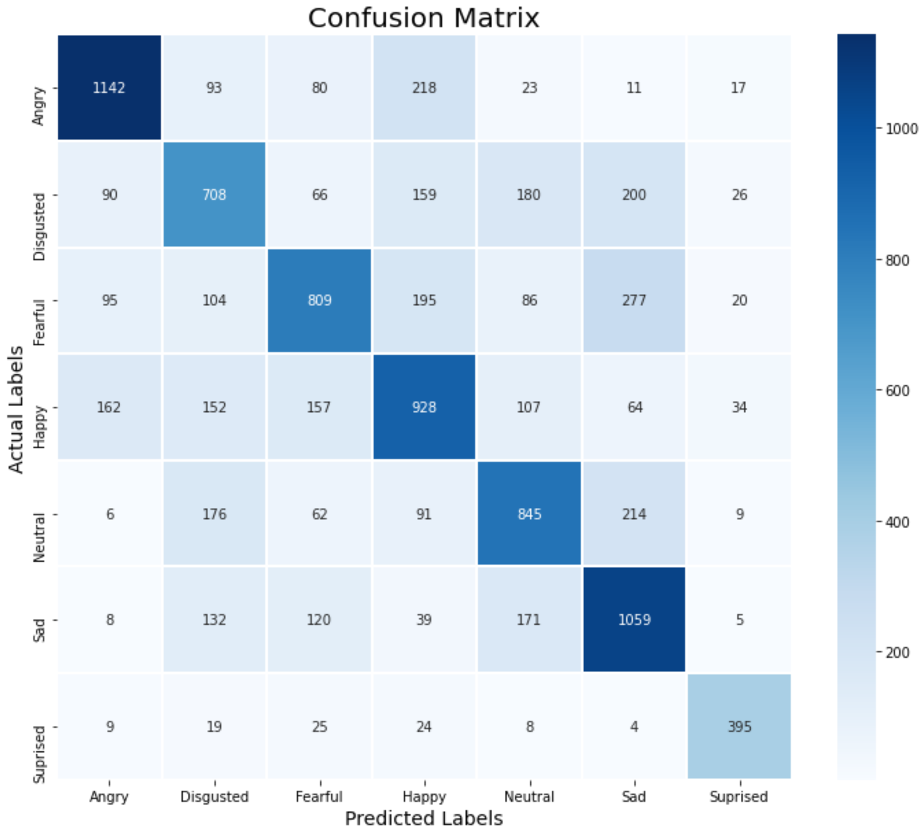


Figure 21. The confusion matrix, showing how well the model does on the test dataset.

5. Supporting Modality Application

In this project, the supporting modality of a visual signal is incorporated. This visual modality is a Facial Emotion Recognition system, which can detect and process human emotions from facial expressions. As this is a supporting modality, we will not go through the in-depth details.

This Facial Emotion Recognition uses the Cohn-Kanade Dataset (CK+), a widely used database for testing and evaluating algorithms. It comprises 7 basic emotion categories: Anger, Contempt, Fear, Sadness, Disgust, Happy and Surprise.

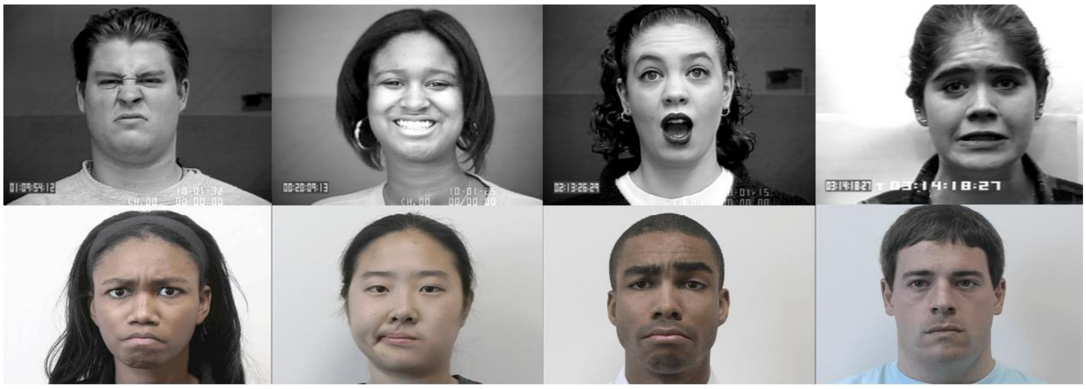


Figure 22. Examples of the CK+ dataset, a representation of the categories collected. From top left to bottom right: Disgust, Happy, Surprise, Fear, Angry, Contempt, Sadness, and Neutral.[34]

For this system, a CNN architecture is built as follows:

- Conv2D- units:512 / filter size: 3x3
- BatchNorm
- MaxPooling- pool size: 2x2 / stride: 2x2
- Conv2D- units: 256 / filter size: 3x3
- BatchNorm
- MaxPooling- pool size: 2x2 / stride: 2x2
- Conv2D- units: 128 / filter size: 3x3
- BatchNorm
- MaxPooling- pool size: 2x2 / stride: 2x2
- Conv2D- units: 64 / filter size: 3x3
- BatchNorm
- MaxPooling- pool size 2x2 / stride 2x2
- Flatten
- Dense- units: 64
- Dense- units: 7 / Softmax output

This model comprises convolutional 2D layers and fully connected layers. The shape of the image data (48x48) is given as input in the 1st layer. A filter size of 3x3 is used in each Conv2D layer to convolve the image data. Padding remains the same after each layer of convolution to retain the image size. Max pooling of size 2x2 with stride 2x2 is used to summarize the features. After flattening the feature extraction process output, a fully connected layer of 64 units process the data before applying the final layer with a softmax output of 7 emotion categories. This model also uses an "Adagrad" as the optimizer.

Training of this model results in reasonably accurate predictions from real-time testing. Figure 23 below shows the possible emotions this model predicted during testing.

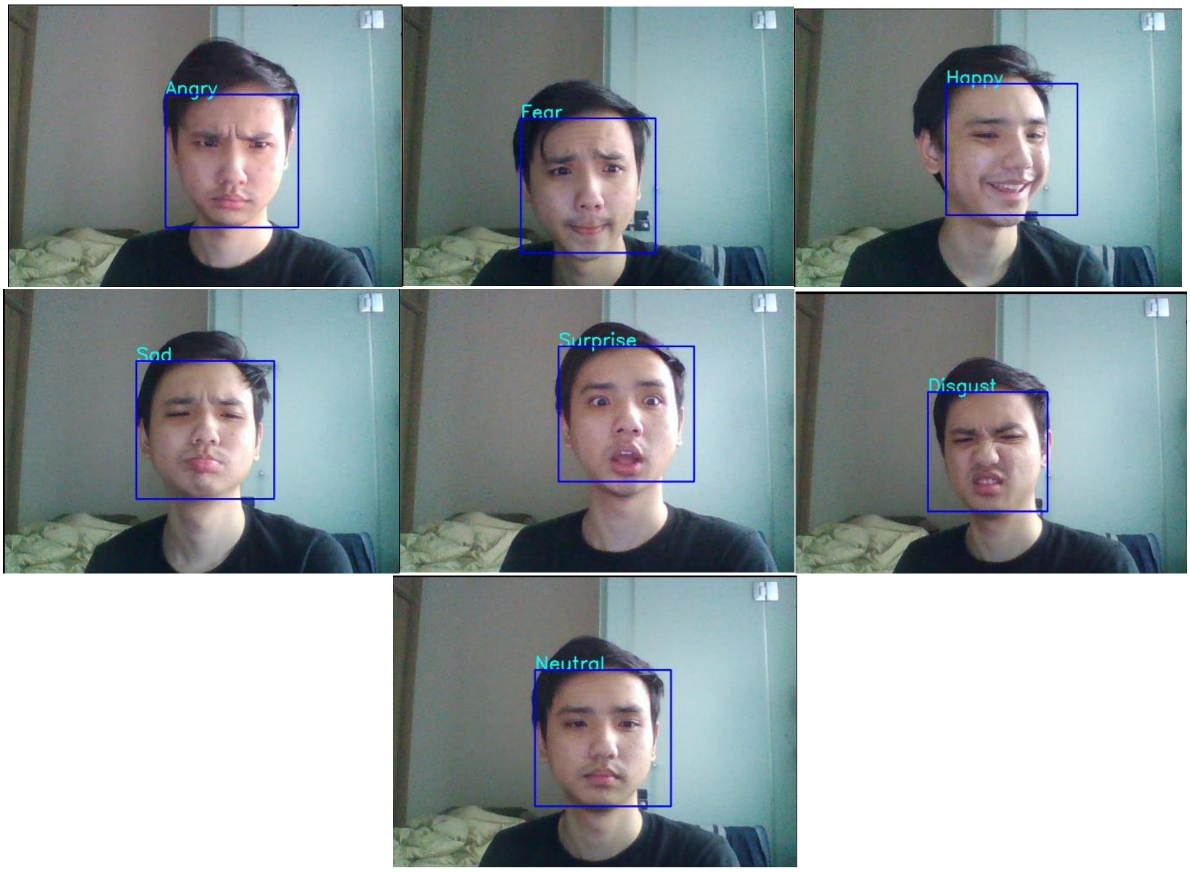


Figure 23. The 7 emotions predicted from the model.

To integrate both the visual modality and the SER system, a simple web application is created with HTML. Using the Flask module, a Python-based web application framework for beginners, we can easily host a local server and develop a web application as a platform for the system.

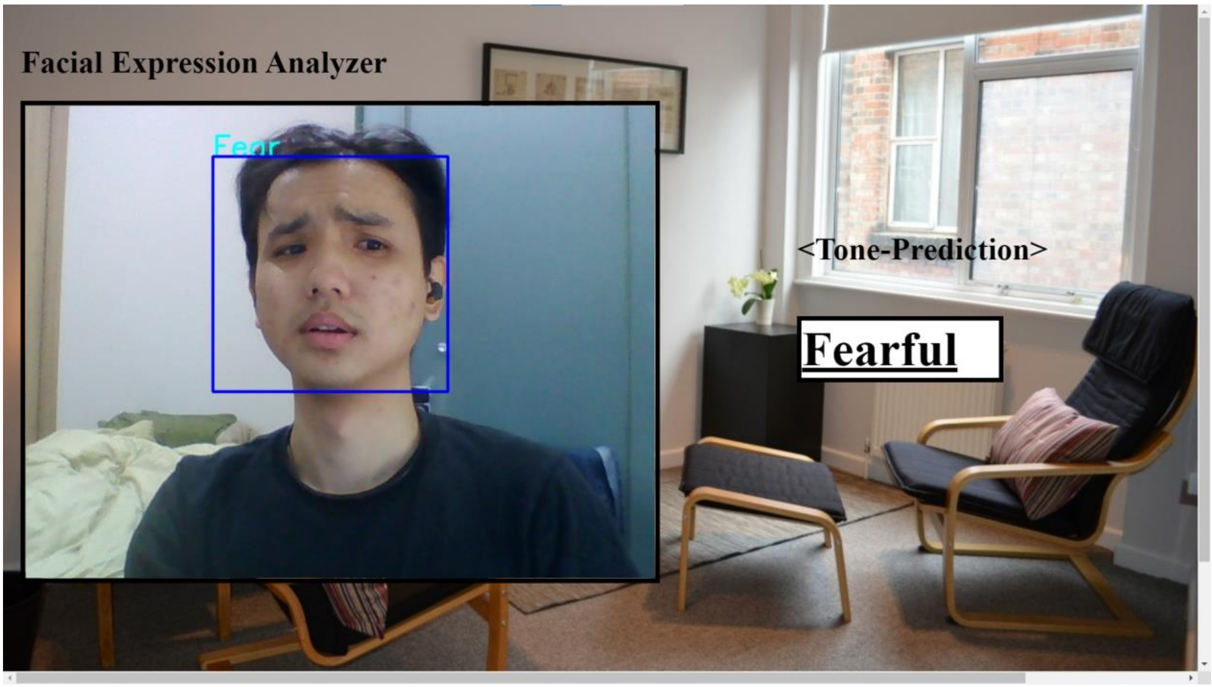


Figure 24. A locally hosted web-based application for the system.

On the left side of this application shows the visual modality mentioned, a Facial Expression Analyzer that is a real-time prediction on the user's current emotion through a live camera feed. The right side of this application will show the user's current emotion through a tone prediction from the SER system. It can be done by taking input from a microphone or through an internal feed, such as playing an audio file.

6. Conclusion

By using the Deep Learning method to construct a Speech Emotion Recognition system, we have demonstrated that 1D Convolutional Neural Network (CNN) architecture can achieve state of the art accuracy. It is also shown that this system can be used on a web application as the platform for real-time prediction, usable in situations like therapy sessions.

Through the experiments done in this project, I've understood the type of features and how they affect the model's training. I've learned that different time and frequency domain features contribute differently towards the training. For example, Spectral Rolloff or Zero-Crossing Rate features produce information in a lower level like loudness or signal shapes, while MFCC and Chroma STFT produce information of a higher level like pitch or spectral shape.

Another learning point would be the methods of building the architecture of the model. Hypertuning of parameters helped significantly in selecting various arbitrary numbers for specific parameters to optimize the model.

6.1. Future Work

An extension of this project is possible at different levels. For a start, feature engineering can be investigated by tuning various parameters in certain features, such as different window functions, hop sizes or frame sizes when extracting MFCC. Other model architecture can be experimented on with the possible improvement of the result, such as RNN with LSTM or GRU, to retain the context of the speech better and understand and predict emotions with higher accuracy.

In terms of framework, the Flask module supports functionality extensions such as database integration or hosting therapy sessions over networks, which is especially helpful during the Covid-19 period. The application can also be enhanced by having different supporting modalities as add-ons.

In terms of the dataset, audio data can be recorded with different accents to fit the user's usage. It is understood that this system is trained primarily with actors using an English accent. To better fit in the Singapore context or lingos, a local dataset can be collected and used for training, aiding therapy sessions locally.

Lastly, as a whole, this application is not limited to the usage in aiding therapy. It can be used in the surveillance of participants' emotions during meetings or journal reflections with the consent of participants.

7. Reference

[1] D. Campbell, "People with bipolar disorder may wait 13 years for diagnosis," *Support the Guardian*, 2012. [Online]. Available: https://www.theguardian.com/society/2012/jun/27/bipolar-disorder-diagnosis-survey.

[2] A. J. Mitchell, A. Vaze, and S. Rao, "Clinical diagnosis of depression in primary care: a meta-analysis," *The Lancet*, vol. 374, no. 9690, pp. 609-619, 2009.

[3] E. Kvarnstrom, "The Dangers of Mental Health Misdiagnosis: Why Accuracy Matters – Bridges to Recovery," *Bridges to Recovery*, 2017. [Online]. Available: https://www.bridgestorecovery.com/blog/the-dangers-of-mental-health-misdiagnosis-why-accuracy-matters.

[4] J. M. Bermudez, "Experiential tasks and therapist bias awareness," *Contemporary family therapy*, vol. 19, no. 2, pp. 253-267, 1997.

[5] A. F. Wisch and J. R. Mahalik, "Male therapists' clinical bias: Influence of client gender roles and therapist gender role conflict," *Journal of Counseling Psychology*, vol. 46, no. 1, p. 51, 1999.

[6] D. DeVault *et al.*, "SimSensei Kiosk: A virtual human interviewer for healthcare decision support," in *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*, 2014, 6, pp. 1061-1068.

[7] M. Castelluccio, "Creating ethical chatbots," *Strategic Finance*, vol. 101, no. 6, pp. 53-55, 2019.

[8] B. W. Schuller, "Speech emotion recognition: Two decades in a nutshell, benchmarks, and ongoing trends," *Communications of the ACM*, vol. 61, no. 5, pp. 90-99, 2018.

[9] P. Gupta and N. Rajput, "Two-stream emotion recognition for call center monitoring," *Eighth Annual Conference of the International Speech Communication Association*, 2007.

[10] W.-J. Yoon, Y.-H. Cho, and K.-S. Park, "A study of speech emotion recognition and its application to mobile services," *International Conference on Ubiquitous Intelligence and Computing*, pp. 758-766, 2007.

[11] L.-S. A. Low, N. C. Maddage, M. Lech, L. B. Sheeber, and N. B. Allen, "Detection of clinical depression in adolescents' speech during family interactions," *IEEE Transactions on Biomedical Engineering*, vol. 58, no. 3, pp. 574-586, 2010.

[12] P. Verma, "300+ Emotions And Feelings: All Emotions List With Definition," *Design Epic Life*, 2021. [Online]. Available: https://designepiclife.com/list-of-emotions/.

[13] D. Bitouk, R. Verma, and A. Nenkova, "Class-level spectral features for emotion recognition," *Speech communication*, vol. 52, no. 7-8, pp. 613-625, 2010.

[14] O.-W. Kwon, K. Chan, J. Hao, and T.-W. Lee, "Emotion recognition by speech signals," in *Eighth European conference on speech communication and technology*, 2003, 20.

[15] H. Karimova, "The Emotion Wheel: What It Is and How to Use It," *Positive Psychology*, 2021. [Online]. Available: https://positivepsychology.com/emotion-wheel/.

[16] "Our Basic Emotions Infographic | List of Human Emotions | UWA Online," *UWA Online*, 2021. [Online]. Available: https://online.uwa.edu/infographics/basic-emotions/.

[17] J. Beck, "New research says there are only four emotions," *The Atlantic Magazine*, 2014.

[18] K. Cherry, "The 6 types of basic emotions and their effect on human behavior," *Verywell Mind*, 2019.

[19] K. Gasper, L. A. Spencer, and D. Hu, "Does neutral affect exist? How challenging three beliefs about neutral affect can advance affective research," *Frontiers in Psychology*, vol. 10, p. 2476, 2019.

[20] B. Anālayo, "What about neutral feelings," *Insight J*, vol. 43, pp. 1-10, 2017.

[21] S. G. Koolagudi and K. S. Rao, "Emotion recognition from speech: a review," *International journal of speech technology*, vol. 15, no. 2, pp. 99-117, 2012.

[22] H. Cao, D. G. Cooper, M. K. Keutmann, R. C. Gur, A. Nenkova, and R. Verma, "Crema-d: Crowd-sourced emotional multimodal actors dataset," *IEEE transactions on affective computing*, vol. 5, no. 4, pp. 377-390, 2014.

[23] S. R. Livingstone and F. A. Russo, "The Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS): A dynamic, multimodal set of facial and vocal expressions in North American English," *PloS one*, vol. 13, no. 5, p. e0196391, 2018.

[24] S.-u. Haq, *Audio visual expressed emotion classification*. University of Surrey (United Kingdom), 2011.

[25] K. Dupuis and M. K. Pichora-Fuller, "Recognition of emotional speech for younger and older talkers: Behavioural findings from the Toronto emotional speech set," *Canadian Acoustics*, vol. 39, no. 3, pp. 182-183, 2011.

[26] P. Knees and M. Schedl, *Music similarity and retrieval: an introduction to audio-and web-based strategies*. Springer, 2016.

[27] L. Roberts, "Understanding the Mel Spectrogram," *Analytics Vidhya*, 2020.

[28] https://en.wikipedia.org/wiki/Recurrent_neural_network (accessed.

[29] M. Nguyen, "Illustrated Guide to LSTM's and GRU's: A step by step explanation," *Online] Towards Data Science*, 2018.

[30] A. Van Den Oord, S. Dieleman, and B. Schrauwen, "Deep content-based music recommendation," in *Neural Information Processing Systems Conference (NIPS 2013)*, 2013, vol. 26, 31: Neural Information Processing Systems Foundation (NIPS).

[31] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *Proceedings of the 25th international conference on Machine learning*, 2008, 32, pp. 160-167.

[32] O. Avilov, S. Rimbert, A. Popov, and L. Bougrain, "Deep learning techniques to improve intraoperative awareness detection from electroencephalographic signals," in *2020 42nd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*, 2020, 33: IEEE, pp. 142-145.

[33] S. Balaji, "Binary Image classifier CNN using TensorFlow," *Techiepedia*, 2020. [Online]. Available: https://medium.com/techiepedia/binary-image-classifier-cnn-using-tensorflow-a3f5d6746697.

[34] P. Lucey, J. F. Cohn, T. Kanade, J. Saragih, Z. Ambadar, and I. Matthews, "The extended cohn-kanade dataset (ck+): A complete dataset for action unit and emotion-specified expression," in *2010 ieee computer society conference on computer vision and pattern recognition-workshops*, 2010, 36: IEEE, pp. 94-101.