Summer 2022

Detecting User Emotions From Audio Conversations With the Smart Assistants

Sunanda Guha
Missouri State University, sg75s@MissouriState.edu

DETECTING USER EMOTIONS FROM AUDIO CONVERSATIONS WITH THE SMART ASSISTANTS

A Master's Thesis

Presented to

The Graduate College of

Missouri State University

In Partial Fulfillment

Of the Requirements for the Degree

Master of Science, Computer Science

By

Sunanda Guha

August 2022

DETECTING USER EMOTIONS FROM AUDIO CONVERSATIONS WITH THE SMART ASSISTANTS

Computer Science

Missouri State University, August 2022

Master of Science

Sunanda Guha

ABSTRACT

With the proliferation of smart home devices like Google Home or Amazon Alexa, significant research endeavors are being carried out to improve the user experience while interacting with these smart assistants. One such dimension in this endeavor is ongoing research on successful emotion detection from short voice commands used in smart home environment. Besides facial expression and body language, etc., speech plays a pivotal role in the classification of emotions when it comes to smart home application. Upon successful implementation of accurate emotion recognition, the smart devices will be able to intelligently and empathetically suggest appropriate actions based on the users' current emotional state. Keeping that in focus, this research aims to advance the existing literature on emotion detection from voice commands in smart home applications. Initially, I chose two publicly available datasets as audio conversation datasets to highlight my application's most effective classification algorithm. Through a comparative analysis, I have concluded that the Tree-based Pipeline Optimization Tool (TPOT) algorithm outperforms other machine learning techniques to detect accurate emotion from an audio. On a concurrent study, I observed that Mel Frequency Cepstral Coefficient (MFCC) in combination with Mel Spectrogram (MEL) result in higher classification accuracy than other existing audio feature combinations available in literature. Upon this conclusion, I have adapted TPOT combined with MEL and MFCC audio feature for our in-house smart home dataset. This Institutional Review Board (IRB) approved in-house dataset contains 5000 smart home voice commands covering five distinctive emotional states from 12 different users. Moving forward, I proposed four new audio features named Chunk Gap Length (CGL), Mean Chunk Duration (MCD), Mean Word Duration Per Chunk (MWDPC), and Per Chunk Word Count (PCWC) to be utilized along with existing MFCC and MEL for improving the accuracy of emotion detection. My evaluation results show that combinations of custom features with MFCC and MEL provide better accuracy in detecting the correct emotion compared to MFCC and MEL alone.

KEYWORDS: speech emotion recognition, machine learning, smart home, voice command, Mel Frequency Cepstral Coefficient (MFCC), Mel Spectrogram (MEL), pitch, Sound Pressure Level (SPL), Chunk Gap Length (CGL), Tree-based Pipeline Optimization Tool (TPOT)

DETECTING USER EMOTIONS FROM AUDIO CONVERSATIONS WITH THE SMART ASSISTANTS

By

Sunanda Guha

A Master's Thesis
Submitted to the Graduate College
Of Missouri State University
In Partial Fulfillment of the Requirements
For the Degree of Master of Science, Computer Science

August 2022

Approved:

Razib Iqbal, Ph.D., Thesis Committee Chair

Lloyd Smith, Ph.D., Committee Member

Siming Liu, Ph.D., Committee Member

Julie Masterson, Ph.D., Dean of the Graduate College

ACKNOWLEDGEMENTS

Firstly, I would like to convey my heartful gratitude to my research mentor, Dr. Razib Iqbal, for his continuous support and motivation since the very first day I joined his MuSyc Lab. I am grateful to him for providing me with the opportunity to work on this exciting research topic. Without Dr. Iqbal's inspiration, guidance, constructive criticism, and great suggestions, the completion of this thesis would have been an impossible task. I am thankful to him for guiding me throughout the whole journey, pushing me beyond boundaries to improve my analytical abilities, interpretation skills, problem-solving and decision-making skills, and encouraging me by celebrating every small research success. I am privileged to learn from his exceptional scientific knowledge and his extraordinary human qualities mixed with kindness, humor, and understanding. For all this and more, I am greatly indebted to him for believing in me at times when even I was doubtful about my capabilities. He is undoubtedly the greatest research mind I met during my graduate study, and I am honored to have him as my mentor. I look up to him as my role model and I wish him every success in his professional and personal life.

I would like to use this opportunity to express my gratitude to the rest of my thesis committee members for their guidance and feedback throughout my degree. I would also like to convey my gratitude to the course professors whom I served as a teaching assistant. Thank you all for being so understanding during my tenure at Missouri State University. Without your support, I could not have completed my degree smoothly. I would also like to thank my fellow MuSyC lab members for the time we spent together. Your company and encouragement helped me to meet the deadlines, gather new ideas to solve research problems, and for jovial conversations to rest my mind outside of research.

Finally, I would like to express my gratefulness to the gracious, the most powerful, the creator of this universe, the Almighty God, for keeping me healthy, strong, and energetic to complete this hard but successful journey. Words cannot describe how grateful I am to my parents and brother for all the sacrifices they made to make me prosper at every step of my life. Their unconditional love and continued support to pursue higher education overseas have been the key motivating factors throughout my graduate career. Last but not least, I would like to thank my husband, Dr. Somen Baidya, for being there with me at difficult times and sharing the hurdles and joys of this endeavor.

I dedicate this thesis to my parents.

TABLE OF CONTENTS

Introduction 1
- Motivation 1
- Research Problem 1
- Research Contributions 3
- Thesis Outline 4

Literature Review 5

Audio Features 15
- Chunk Gap Length (CGL) 15
- Mean Chunk Duration (MCD) 16
- Mean Word Duration Per Chunk (MWDPC) 17
- Per Chunk Word Count (PCWC) 18
- Mel-Frequency Cepstrum Coefficients (MFCC) 18
- Mel Spectrogram (MEL) 20

Review of Classifiers 21
- K-Nearest Neighbor (KNN) 21
- Single Layer Perceptron (SLP) 22
- Multi-Layer Perceptron (MLP) 23
- Support Vector Machine (SVM) 24
- Logistic Regression (LR) 25
- Adaptive Boosting (AdaBoost) 26
- Random Forest (RF) 27
- Tree-based Pipeline Optimization Tool (TPOT) 28

Implementation 31
- Audio Dataset 32
- Feature Extraction 38
- Classification Algorithms 48

Performance Evaluation 49

Conclusions 88

LIST OF TABLES

Table 1. Comparing related works based on dataset, emotions, features, algorithms, and accuracy 13

Table 2. Considered Data from RAVDESS and TESS dataset 33

Table 3. Smart voice commands from the smart home dataset 34

Table 4. Smart home dataset description 37

Table 5. Experimental results of different times of recorded dataset 50

Table 6. Experimental results of emotion detection for different classifiers using MFCC feature for RAVDESS dataset..... 51

Table 7. Experimental results of emotion detection for different classifiers using the RAVDESS dataset 52

Table 8. Experimental results of emotion detection for TPOT classifiers using the TESS dataset 53

Table 9. Experimental results of emotion detection for different classifiers using the smart home dataset (S007)..... 54

Table 10. Experimental results of emotion detection for TPOT classifier using the combined smart home dataset..... 63

Table 11. Results for TPOT classifier using different participants of the smart home dataset 86

Table 12. Emotion detection for unlabeled data using the different combinations of features 87

LIST OF FIGURES

Figure 1. Graphical overview of custom audio features 16
Figure 2. Block diagram of the MFCC architecture [51] 20
Figure 3. A schematic overview of the KNN algorithm 22
Figure 4. A schematic overview of the SLP algorithm 23
Figure 5. A schematic overview of the Multilayer Perceptron (MLP) algorithm 24
Figure 6. A schematic overview of the SVM algorithm 25
Figure 7. A schematic overview of the LR algorithm 26
Figure 8. A schematic overview of the AdaBoost algorithm 27
Figure 9. A schematic overview of the RF algorithm 28
Figure 10. A schematic overview of the TPOT algorithm 29
Figure 11. Emotion detection workflow diagram 31
Figure 12. Utterance detection of mono audio data when energy threshold = calculate energy single channel (data,1) 40
Figure 13. Utterance detection of mono audio data when energy threshold = calculate energy single channel (data,2) 41
Figure 14. Utterance detection of mono audio data when energy threshold = calculate energy single channel (data,2)-10 42
Figure 15. Confusion matrix for the combined smart home dataset considering MEL features using TPOT classifier 55
Figure 16. Confusion matrix for the combined smart home dataset considering MFCC features using TPOT classifier 56
Figure 17. Confusion matrix for the combined smart home dataset considering Chroma features using TPOT classifier 57
Figure 18. Confusion matrix for the combined smart home dataset considering Contrast features using TPOT classifier 58
Figure 19. Confusion matrix for the combined smart home dataset considering Tonnetz features using TPOT classifier 59
Figure 20. Confusion matrix for the combined smart home dataset considering MEL, MFCC features using TPOT classifier 60
Figure 21. Confusion matrix for the combined smart home dataset considering MEL, MFCC, Chroma features using TPOT classifier 61
Figure 22. Confusion matrix for the combined smart home dataset considering MEL, MFCC, Contrast features using TPOT classifier 62
Figure 23. Confusion matrix for the combined smart home dataset considering MEL, MFCC, Contrast, Chroma, Tonnetz features using TPOT classifier 63
Figure 24. Confusion matrix for the S001 considering MEL, MFCC features using TPOT classifier 64
Figure 25. Confusion matrix for the S001 considering MEL, MFCC, MWDPC features using TPOT classifier 65
Figure 26. Confusion matrix for the S002 considering MEL, MFCC features using TPOT classifier 66

Figure 27. Confusion matrix for the S002 considering MEL, MFCC, PCWC features using TPOT classifier 67
Figure 28. Confusion matrix for the S002 considering MEL, MFCC, MWDPC, PCWC features using TPOT classifier 68
Figure 29. Confusion matrix for the S003 considering MEL, MFCC features using TPOT classifier 69
Figure 30. Confusion matrix for the S003 considering MEL, MFCC, MCD, CGL features using TPOT classifier 70
Figure 31. Confusion matrix for the S004 considering MEL, MFCC features using TPOT classifier 71
Figure 32. Confusion matrix for the S004 considering MEL, MFCC, MWDPC features using TPOT classifier 72
Figure 33. Confusion matrix for the S005 considering MEL, MFCC features using TPOT classifier 73
Figure 34. Confusion matrix for the S005 considering MEL, MFCC, CGL features using TPOT classifier 74
Figure 35. Confusion matrix for the S006 considering MEL, MFCC features using TPOT classifier 75
Figure 36. Confusion matrix for the S006 considering MEL, MFCC, MCD features using TPOT classifier 76
Figure 37. Confusion matrix for the S007 considering MEL, MFCC features using TPOT classifier 77
Figure 38. Confusion matrix for the S007 considering MEL, MFCC, MCD, CGL features using TPOT classifier 78
Figure 39. Confusion matrix for the S008 considering MEL, MFCC features using TPOT classifier 79
Figure 40. Confusion matrix for the S008 considering MEL, MFCC, MWDPC features using TPOT classifier 80
Figure 41. Confusion matrix for the S009 considering MEL, MFCC features using TPOT classifier 81
Figure 42. Confusion matrix for the S009 considering MEL, MFCC, CGL features using TPOT classifier 82
Figure 43. Confusion matrix for the S010 considering MEL, MFCC features using TPOT classifier 83
Figure 44. Confusion matrix for the S010 considering MEL, MFCC, MCD features using TPOT classifier 84
Figure 45. Confusion matrix for the S010 considering MEL, MFCC, MWDPC features using TPOT classifier 85

LIST OF ALGORITHMS

Algorithm 1. Chunk Gap Length Extraction 43

Algorithm 2. Mean Chunk Duration Extraction 45

Algorithm 3. Mean Word Duration Per Chunk Extraction 46

Algorithm 4. Per Chunk Word Count Extraction 47

LIST OF ACRONYMS

CGL -	Chunk Gap Length
MCD -	Mean Chunk Duration
MWDPC -	Mean Word Duration Per Chunk
PCWC -	Per Chunk Word Count
MFCC -	Mel-Frequency Cepstrum Coefficients
MEL -	Mel Spectrogram
KNN -	K-Nearest Neighbor
SLP -	Single Layer Perceptron
MLP -	Multi-Layer Perceptron
SVM -	Support Vector Machine
LR -	Logistic Regression
AdaBoost -	Adaptive Boosting
RF -	Random Forest
TPOT -	Tree-based Pipeline Optimization Tool
RAVDESS -	Ryerson Audio-Visual Speech Set
TESS -	Toronto emotional Speech Set
SPL -	Sound Pressure Level
DNN -	Deep Neural Networks
SER -	Speech Emotion Recognition
EMO-DB -	Berlin Emotional Speech Database
IRB -	Institutional Review Board

INTRODUCTION

Motivation

Emotion detection from voice commands is an interesting research avenue due to the recent increase in the popularity of smart home devices like Amazon Alexa and Google Home. Automatic emotion recognition from short voice commands can make a smart home smarter by enabling automatic policy enforcement [1] to enhance the inhabitant's quality of experience [2]–[4]. With the overwhelming usage of smart devices in our daily life, communication with the device interface is one of the promising areas of research and innovation. The voice commands that we use to communicate with smart devices contain two key features - context and emotion [4]. Modern-day technology has successfully implemented the interlink between user and device based on the context of our speech [5]. Some smart devices have already been programmed to perform appropriate actions based on the context of our conversation [6]–[13]. In contrast, emotion is believed to carry much more sophisticated information related to the state of our mind but remains an unexplored dimension when it comes to getting utilized in smart home devices. Upon successful retrieval of the exact emotion from our conversation, it can be used to assist with emergency services, call centers, medical services, and many other potential fields. For example, if the smart assistant can detect fearful emotion in the user's voice command, it can suggest dialing 911, call next of kin or sound an alarm.

Research Problem

Humans can typically understand the emotional state of another human because of years of learning and practice. This can be difficult to replicate because the amount of data that would need to be fed into the learning algorithm would be extensive. While humans can consider other

factors like the speaker's physical state and facial expressions, speech-only detection systems cannot incorporate these features and hence, do not always yield the most consistent and accurate results [14]. While speech is only one aspect of this emotion detection puzzle, it is an aspect of utmost importance because of its inseparable integration in humans [15]. Keeping that in focus, the audio emotion detection system works to diminish the gap between human and machine capabilities to recognize the current emotional state of users. Methodically, for emotion detection from any sort of audio event, there are two aspects we need to consider simultaneously – i) selecting the appropriate audio features, and ii) selecting a classification algorithm. Most of the reported studies on audio emotion detection incorporate different combinations of audio features and machine learning algorithms to improve the accuracy of emotion detection [22]–[24]. Due to the complex nature of human speech, there are various challenges in retrieving accurate information by extracting audio features. Each emotional state has distinct speech features categorized as prosodic and dynamic features. Prosodic features such as pitch, sound pressure level [16], and energy are commonly used in the field of emotion detection [16]–[18] but they sometimes fail to distinguish certain emotions like happy or angry [19]. Therefore, dynamic features, e.g., Mel-frequency Cepstral Coefficient (MFCC) [20], [21], are considered in addition to the prosodic features. Researchers have also provided insight into which classification algorithms would be well-suited to adopt when it comes to detecting emotion from speech through years of research. These options include K-Nearest Neighbor (KNN) [18], Deep Neural Networks (DNN) [22], Convolutional Neural Network (CNN) [23], and Support Vector Machine (SVM) [24].

Research Contributions

The primary focus of this thesis is to find audio features from smart home short voice commands and investigate a suitable classification algorithm for emotion classification. For this purpose, I have first extracted a combination of existing audio features and applied some machine learning algorithms on the publicly available RAVDESS dataset [25] and TESS dataset [26]. By the end of the primary phase, I have concluded Tree-based Pipeline Optimization Tool (TPOT) along with the combination of MFCC and MEL audio features outperforms all other combinations. Thus, I picked MEL, MFCC and TPOT and applied into an in-house custom dataset solely developed for this research. The dataset is Institutional Review Board (IRB) approved and consists of 5000 smart home voice commands covering five distinctive emotional states – happy, normal, sad, fearful, and angry from ten different users [27]. The motivation behind developing such a dataset is, that most of the voice commands used with a smart home assistant are short and do not carry elaborate information about the context, otherwise known as lexical cues, as compared to the regular human-human conversation as available in the existing literature. Consequently, there is a window of increasing the classification accuracy by incorporating dynamic features. To increase the accuracy of the detection algorithm further, I introduce four new audio features named Chunk Gap Length (CGL), Mean Chunk Duration (MCD), and Mean Word Duration Per Chunk (MWDPC), and Per Chunk Word Count (PCWC) along with existing MEL and MFCC. The novelty of this study resides in analyzing short smart home voice commands (in short, smart commands) and interactions with smart assistants for audio emotion detection which earlier studies did not explore to the best of my knowledge. Also, combining the custom audio features and existing features enhances the system's accuracy for each user in the smart home dataset using TPOT.

Thesis Outline

The rest of the thesis is organized as follows: In Section *Literature Review*, I provide an overview of similar studies and present a summary of their reported accuracy. I then describe the audio features in Section *Audio Features* and briefly discuss the classifiers in Section *Review of Classifiers*. The workflow diagram to detect emotion along with audio feature implementation is presented in Section *Proposed Methodology and Implementation*. I explain the dataset and performance evaluation in Section *Performance Evaluation*. Finally, I present my concluding remarks in Section *Conclusion*.

LITERATURE REVIEW

In the field of emotion detection from an audio conversation, several studies have been conducted which incorporate some specific audio features and classification algorithms. This section will present an overview of some of the state-of-the-art research on audio emotion detection.

In [24], a speech recognition system was presented where the authors applied a multilevel SVM classifier on a multi-lingual dataset. This dataset consists of Assamie, Dimasa, Bodo, Karbi, and Mishing languages. The authors chose 600 samples for each language in their study. A unique combination of 49 features, including 4 prosodic features (pitch, energy, zero-crossing rate, and Log-entropy), 6 quality features (3 format frequencies, spectral roll-off, spectral flux, spectral centroid), 14 MFCC, 12 Linear Predictive Coding Coefficients, and 13 Mel-Energy spectrum Dynamic Coefficients were utilized. This specific feature combination achieved 79.3% accuracy for Assamie, 78.57% for Dimasa, 82.8% for Bodo, 89.23% for Karbi, and 81.43% for Mishing Language. In a similar study [18], the authors applied the KNN algorithm to detect speech emotions. The authors used Berlin emotional database [28] and incorporated pitch, zero-crossing count, entropy, and MFCC as classification features for four emotional states (angry, sad, neutral, and happy) and achieved 86.02% accuracy.

In a more recent study, in [29], authors utilized IEMOCAP and AVEC datasets to predict happy, sad, neutral, angry, excited, and frustrated emotional states using the Recurrent Neural Network (RNN) classifier. Authors extracted textual features, i.e., n-gram features, audio and visual feature using 3D-CNN [30] and openSMILE [31]. The system can identify emotional states in conversation with 78.80% accuracy. A new dataset named Multimodal EmotionLines Dataset

has been proposed in [32], where 13,000 utterances have been collected from 1,433 dialogues of the popular TV series “Friends”. The authors of this study applied DialogueRNN to detect happy, anger, disgust, sadness, surprise, and neutral emotional states. Here, 1-D CNN was used to extract textual features, and the openSMILE toolkit was incorporated to extract audio features which provided 6373-dimensional features and achieved 67.56% accuracy. This dataset can help extract new features from audio, video, and textual modalities. In [23], the authors proposed a system to detect the emotion of adult persons in the nursing home using CNN. In this case, 95% accuracy was achieved by applying the normalization and augmentation process together. The authors have incorporated the RAVDESS dataset and extracted spectrogram features to detect happy, sad, angry, fearful, surprised, and disgust emotional states.

In [20], the authors applied a Gaussian mixture model (GMM). This study was conducted on a recorded dataset of 27 speakers of the Assamese language and reported detecting happy, surprise, and angry emotions based on the MFCC feature. The system achieved 76.5% accuracy. In [33], researchers have applied RNN on the IEMOCAP dataset [34] to detect happy, sad, neutral, and angry emotions. To achieve their goal, the authors extracted raw spectral features i.e., 257-dimensional magnitude Fast Fourier Transform (FFT) vectors along with low-level descriptors, i.e., fundamental frequency, voicing probability, frame, energy, zero-crossing rate, and 12 MFCC, and achieved an accuracy of 58.8%. The authors proposed a weighted time pooling strategy in this study that considers emotionally salient parts of an utterance. Another study based on a deep learning network has been reported in [22]. In this study, the authors developed a unique architecture, named ADRNN, which incorporated dilated CNN with residual block and Bidirectional LSTM based on the attention mechanism. The study reported 85.39% accuracy in the speaker-independent experiment, 64.74% accuracy in the IEMOCAP dataset, and

85.39% accuracy in the Berlin EMODB dataset by extracting features from the 3-D log Mel spectrogram. In [35], the authors incorporated a deep neural network to detect neutral, happy, sad, and angry emotions from the IEMOCAP dataset by extracting the MFCC feature with 70.6% accuracy. In [36], both CNN and RNN have been applied to recognize speech emotions neutral, anger, fear, disgust, sadness, boredom, and happiness from the Berlin EMODB dataset by analyzing the Fourier transform coefficients of the input audio.

Real-time speech emotion recognition was attempted on the open-source Berlin Emotional Speech Database (EMO-DB) dataset in [37]. The EMO-DB dataset consists of 535 short German utterances categorized into seven different emotions- anger, boredom, anxiety, happiness, sadness, disgust, and neutral. Each sample in the dataset was initially recorded with a 48 kHz sample rate. Then, each recording was down sampled to have a 16 kHz sample rate. Instead of classifying extracted acoustic features, the authors proposed converting each utterance into a visual spectrogram and then classifying the images. They used a transfer learning approach with a trained image-based convolution neural network known as AlexNet. AlexNet was first introduced in [38]. The model has been trained on 1.2 million images and has been able to distinguish images between 1,000 object classifications [37] successfully. The model consists of a three-channel input (red, green, and blue) allowing two-dimensional images with a resolution of 256 × 256. Five convolutional layers follow the input layer, with each convolution layer containing max pooling and normalization layers. Three fully connected layers exist after the last convolutional layer with an exponential SoftMax function that converts the output from the last fully connected layer into a normalized vector. This normalized vector contains floating-point numbers in the set [0,1] that represent the probability for each possible classification. The class with the highest probability is

returned. Using this model with spectrogram representations of the EMO-DB dataset, the authors achieved an average accuracy of 82% with a return time of 1.033-1.026 seconds.

A hybrid model for speech emotion recognition, using both a GMM and deep neural network (DNN) proposed in [39]. In this project, the authors created their dataset for classification, titled the Emirati Speech Database (ESD). A group of 15 men and 15 women, ranging from 14 to 55 years old, were involved in the production of the ESD. A list of eight commonly spoken sentences in the United Arab Emirates was given to each speaker. Each participant spoke every sentence nine times in the following emotions: neutral, happy, sad, disgust, anger, and fear. All lengths of the recordings in the ESD are between two and five seconds. Each sentence was initially recorded at a sample rate of 44.6 kHz, and then the recordings were downsampled to 16 kHz. For feature extraction, they compute MFCC of the sentence inputs and then send this data to a cascaded GMMDNN classifier. The GMM first computes the log probability of training voice vectors during model training. The previously computed log probability competes with stored voice data during model testing, and a "GMM tag" is generated. In this context, a GMM tag is a vector consisting of a 0 or 1 associated with each possible emotion. This means that six GMM tags are created for each input sentence as there are six possible classification outputs. Once the GMMs tag is created, this is used as the input for the DNN. The DNN consists of a CNN with four hidden layers; Within these four hidden layers are 256 rectified linear hidden units and a gradient descend optimization stage. Overall, the DNN creates a probability value for each emotion. At the end of the GMMDNN hybrid model, a decision block returns the classification with the highest probability value. Using this architecture, the authors achieved a classification accuracy of 83.97% over the ESD. This paper does not present an average or range of return times based on the GMM-DNN hybrid model.

In [40], the authors compared the classification accuracy of standard vector machine (SVM) and Multi-Layer perception (MLP) models using the Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS). The RAVDESS data set comprises 24 professional voice actors, who each perform 60 trials in the song channel and 44 trials in the song channel. The emotions expressed within both channels include calm, happy, sad, angry, and fear. All the files in the RAVDESS dataset are 16 bits encoded .wav files with a sample rate of 48 kHz. The authors note that the disgust and surprising emotion categories were dropped from both channels in the original dataset, as these categories were missing in the song channel. Before classification, feature extraction is completed using the Librosa Python library. The following five features were extracted from the RAVDESS samples for classification: Melscaled spectrogram, Mel-frequency Cepstral Coefficients (MFCCs), Chromagram, spectral contrast, and Tonnetz. These features were then sent to an SVM and MLP model for classification. The optimized SVM model yielded a total classification accuracy of 82%. The optimization, in this case, consists of a grid search, where all possible combinations of parameters are tested, and the combination that yields the highest accuracy is kept. The parameters that yielded 82% accuracy were 41.8 as C, 0.03 as gamma, and the radial basis function (RBF) as the kernel. As for the MLP model, an accuracy of 75% was achieved by parameter grid search optimization. The following parameters were found to yield the highest accuracy: “adam” for the solver, “constant” for learning rate, “(100,50)” for the hidden layer sizes, “1e-08” for epsilon, “0.001” as alpha, and “relu” as the activation function. The authors explain that many misclassified samples occurred between the sad and fearful classifications while using the MLP model.

In [41], the authors proposed a similar solution in [40] by converting audio samples into visual spectrograms and then performing classification with a convolutional neural network.

However, in this case, both a trained AlexNet model and a custom CNN model alternative are used and compared. Badshah et al. note that using a custom-trained CNN model can be more effective as it suits the structure of the spectrogram-based input data better than a generalized pre-trained model such as AlexNet. Additionally, facilitating transfer learning on pre-trained models usually increases complexity and decreases training performance. The dataset used in this research was the Berlin dataset, which consists of speech data from four different individuals. All the data is sorted into seven different emotion classifications: neutral, fear, anger, happy, sadness, disgust, and boredom. The model architecture was shared across the trials using the trained and custom model. The architecture accepts a three-channel (red, green, and blue) 256×256 images in the form of a spectrogram. The first three layers in the network consist of convolutional layers. The first convolutional layer has 120 kernels with a 4-pixel stride setting. The first convolutional layer is a 3×3 size max pooling layer with a 2-pixel stride setting and a rectified linear unit (ReLU) layer. The second and third convolutional layers are followed by the same max pooling and ReLU layers as the first convolutional layer, except the kernel sizes are 256 and 384, respectively. The second and third convolutional layers have a stride setting of 1 pixel. After the convolutional layers, three fully connected layers have neuron counts of 2048, 2048, and 7, respectively. Badshah et al. note that dropout layers were added after the first two fully connected layers to prevent model overfitting. After running two tests on the same dataset between the custom trained model and the pre-trained AlexNet model, it was found that the custom trained model was superior, with an accuracy of 84.3%.

In [42], the authors emphasize the role that pre-feature extraction processing serves in the accuracy of a speech emotion recognition model. This paper does not contain information about a proposed speech emotion recognition (SER) system. However, the authors present several

preprocessing methods that can be used to improve the accuracy across different models for SER. The first preprocessing stage presented is called “pre-emphasis”, which involves running the speech through a high-pass filter. High pass filters gradually attenuate frequencies in decibels per octave. In other words, high-frequency bands are allowed to pass while low-frequency bands are at the least attenuated or eliminated. By using high pass filters on the input data, low-frequency bands typically outside the human vocal range are eliminated. Because this subset of data contained in the input signal does not indicate human emotions, eliminating it can improve the accuracy of different SER models. Another preprocessing stage introduced is normalization, which involves applying a constant gain to an audio signal such that the peak of the signal reaches a specified decibel value. Maintaining a constant sound level between different classification groups alleviates the possibility of the volume being too influential on classification results. Lastly, the authors mention silence removal as a viable stage in the preprocessing chain. Silence removal involves detecting silent sections in audio before and after speech commands in audio recordings. In this context, “silence” refers to sections where there is no relevant speech. These “silent” sections can still contain subtle environmental noise that serves as meaningless data. Removing these sections in speech data as a preprocessing method is another way of omitting redundant data, increasing the classification accuracy of SER models. The system proposed in this paper is distinct in that it is purposed to run on smart home devices, which do not possess the computing power of a standard desktop computer. Using a CNN was considered. However, converting all incoming audio to visual representations and then classifying this information using a CNN is not viable for real-time processing on a smart home device. In this context, real-time is a return time of under two seconds. Other proposed solutions in the literature that use multiple neural networks in parallel to achieve accurate classification cannot also perform in real-time. We

found that an MLP model can return classification results within two seconds. The total return time of the neural network allows the system to perform both audio preprocessing and classification in under one second on certain systems. It is important to note that the time it takes to train our proposed system does not happen under two seconds; The time varies depending on the number of training samples used. We are operating under the assumption that smart home devices containing an SER system would not have model training capabilities. Another major concern with implementing existing solutions into smart home devices is the dataset used for training. Commands given to smart home devices are typically short, containing 2-6 words. Commonly used datasets in SER literature (EMO-DB, RAVDESS, etc.) do not accurately reflect the typical input into smart home devices; The samples are either too long or too short. We have created an internal dataset used for training that accurately reflects the typical length and delivery given to smart home devices. Additionally, the introduction of an audio preprocessing chain allows our proposed system to perform accurately in a wider range of environments. All submitted audio during training and testing is funneled through a denoising stage (i.e., vocal isolator) that mitigates background audio. This was added to combat accuracy loss in noisy environments where a low signal-to-noise ratio is likely.

An overview of the studies mentioned above is presented in Table 1. Here, we can see that different studies that have considered different datasets, numbers of emotions, features, and algorithms and achieved different accuracy. While the objectives of these studies are similar, the proposed approaches can be hard to compare due to their associated differences in corpora, algorithms, and feature extraction techniques.

Table 1. Comparing related works based on dataset, emotions, features, algorithms, and accuracy

Dataset	No. of Emotions	Features	Algorithm	Accuracy
MESDNEI [24]	7	Pitch, ZCR, Short-term energy, Log-entropy, 3 format frequencies, Spectral Roll-off, Spectral Flux, Spectral centroid, 14 MFCC, 12 LPCC,13 MESDC	Multilevel SVM	78-89%
Berlin Database [18]	4	Energy, pitch, ZCC, Entropy, MFCC	KNN	86.02%
IEMOCAP, AVEC [29]	6	n-gram features, text features using 3-D CNN and OpenSMILE	RNN	78.80%
MELD [32]	6	Textual feature using 1-D CNN, audio feature using OpenSMILE	DialogueRNN	67.56%
RAVDESS [23]	6	Spectrogram	CNN	95%
Assamese Dataset [20]	3	MFCC	GMM	76.5%
IEMOCAP [33]	4	257-D FFT, fundamental frequency, voicing probability, frame, energy, ZCR, 12 MFCC	RNN	58.8%
IEMOCAP, BerlinEMODB [22]	4	3-D long Mel spectrogram	CNN, BiLSTM	69-90%
IEMOCAP [35]	4	MFCC	DNN	70.6%
Berlin EMODB [36]	7	Fourier Transform coefficients	CNN, RNN	73-90%
Berlin EMODB [37]	7	Spectral magnitude	AlexNet	82%
ESD [39]	6	MFCC	GMM, DNN	83.97%
RAVDESS [40]	5	Melscaled spectrogram, Mel-frequency Cepstral Coefficients (MFCCs), Chromagram, spectral contrast, and Tonnetz	SVM, MLP	82%
Berlin EMODB [41]	7	Spectrogram	CNN	84.3%

Although some of the studies have demonstrated excellent classification accuracy, to the best of the scope of this research, no studies have performed the emotion detection analysis on a dataset solely based on the smart home context. To address this issue, I created a dataset that is based on everyday voice commands that we use to interact with smart assistants and does not contain lexical cues about any specific emotional state of the user. As evident from the list of

features in Table 1, MEL and MFCC are two of the most widely adopted audio features incorporated in previous studies on audio emotion detection. Also, there is still some window of improvement when it comes to the accuracy of the machine learning algorithms reported in the previous literatures. Hence, to detect emotions from my custom dataset with improved accuracy, I introduced Chunk Gap Length (CGL), Mean Chunk Duration (MCD), Mean Word Duration Per Chunk (MWDPC), and Per Chunk Word Count (PCWC) features, and combined them with MFCC and MEL for emotion classification. These audio features along with the dataset are the building block of my proposed methodology. While finding the appropriate classification algorithm is a multifaceted problem by itself, I performed a comparative study of 7 widely used classification algorithms along with auto-machine learning (AutoML) system, combined with the audio features, to determine the appropriate classifier to detect emotions from two publicly available datasets, RAVDESS and TESS. Afterward, I incorporated the algorithm associated with the highest classification accuracy into the smart home dataset and validated my conclusion.

AUDIO FEATURES

In this section, I am going to provide an overview of the novel and existing audio features that have been implemented in this research, to detect emotion from short conversations used in the smart home environment.

Chunk Gap Length (CGL)

To harvest the influence of the dynamic features of rhythm and stress in determining the emotion of a user, I introduced a novel feature named Chunk Gap Length (CGL). In CGL, a gap between the chunks in a voice command is considered where a chunk may contain more than one word. For example, if a smart home user suddenly stutters while delivering a voice command, then it might lead to a larger gap between the chunks which could be associated with the sign of that person being nervous or under a stressful condition [43]. Similarly, voice commands from a user with an angry emotional state may have smaller gaps between subsequent chunks. These gaps between chunks can also be more prevalent when a person is happy or sad [17].

A graphical depiction of the gap between two consecutive chunks in a sample audio file is presented in Fig 1. Here, the total gap between “remind me”, “to”, “do, “laun”, and “dry” chunks are considered as CGL. As can be seen from this figure, an audio recording will contain varying signal strengths representing the user’s voice activity. When the signal strength falls below a certain threshold then that can be considered noise or silence. As can be seen in Figure 1, the silence periods in the beginning and the end of each audio recording are not useful for the voice activity analysis. Hence, these two segments of silence at the beginning and the end of each audio file are not considered while calculating the CGL. Such assumption holds its validity as the silence periods in the beginning and at the end of each audio recording do not contain any

information about the users' emotional state and rather might lead to an erroneous input while calculating the silence period due to coherent inconsistency. The red dotted line in Figure 1 refers to the threshold of silence which is calculated from the energy of the audio data.

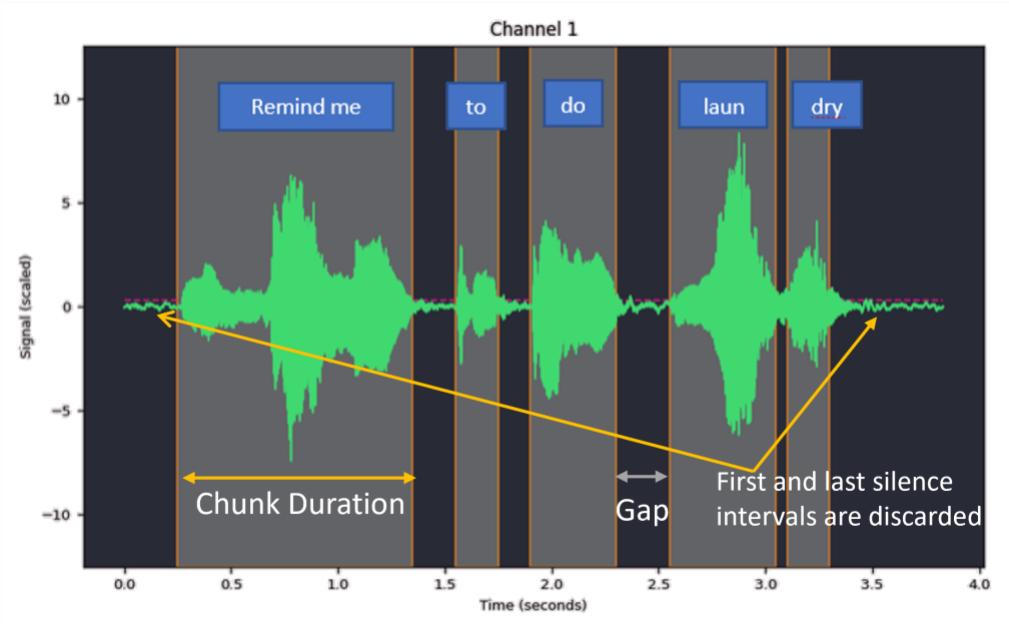


Figure 1. Graphical overview of custom audio features

Mean Chunk Duration (MCD)

Based on the same hypothesis of CGL, the time to utter the same number of words in different states of mind may vary depending on the emotional state of a user [44]. Therefore, I introduce the Mean Chunk Duration (MCD) feature where I calculate the meantime of the detected audio chunks, without the gaps between the chunks, in a voice command.

To distinguish the time segments associated with the audio event, I used the same signal threshold as CGL. For better understanding, a pictorial depiction of chunk duration is also shown in Figure 1, where “remind me”, “to”, “do, “laun”, and “dry” are the chunks of voice activity, and the average duration of uttering these chunks represent MCD. Compared to CGL, the sum of

the chunk duration for MCD is not considered because some voice commands may return larger chunk sizes. For example, "go home" versus "remind me to do laundry". Whereas in CGL, the sum of silence periods is more appropriate otherwise vital information about the user's emotional state based on the length of the silence between chunks might be lost.

Mean Word Duration Per Chunk (MWDPC)

In Mean Word Duration Per Chunk (MWDPC), the average duration of each word in a voice command is calculated, by splitting the command into chunks and discarding the gaps between chunks. The split chunks for each command based on the voice activity threshold are divided by the number of words in each chunk.

For example, in Figure 1, “Remind me” is a chunk with 2 words. For MWDPC, the duration of uttering this chunk is calculated and then divided it by the number of words in this chunk. The same for the other chunks in the voice command is calculated similarly. Finally, the average of the word duration for all the chunks in that command is measured. To detect Mean Word Duration Per Chunk, the following equation (1) is followed:

$$\text{MWDPC} = \text{avg}\ \frac{t_{chunk}}{n_{word\ in\ chunk}} \tag{1}$$

where t_{chunk} is each chunk duration of an audio event and $n_{word\ in\ chunk}$ is the number of words in each chunk. In Figure 1, there are 5 chunks. Thus, the MWDPC of the command “remind me to do laundry” is:

$$\text{MWDPC} = \text{avg}\ \left(\frac{t_{Remind\ me}}{2} + \frac{t_{to}}{1} + \frac{t_{do}}{1} + \frac{t_{laun}}{1} + \frac{t_{dry}}{1} \right) \tag{2}$$

Per Chunk Word Count (PCWC)

The final custom feature used in this research is Per Chunk Word Count (PCWC). PCWC represents how fast or slow the user is uttering the words in terms of word count per chunk per command. To obtain PCWC, the average number of words per chunk in each command is calculated as per the following equation (3):

$$PCWC = avg\ (nw_{chunk1} + nw_{chunk2} + nw_{chunk3} + \ldots\ldots\ldots\ldots\ldots + nw_{chunkn}) \quad (3)$$

where nw_{chunkn} represents the number of words in a given chunk. For example, PCWC of the command in Figure 1 is PCWC = avg (2+1+1+1+1) = 1.2.

Mel-Frequency Cepstrum Coefficients (MFCC)

MFCC is a widely adapted audio feature that can provide an accurate representation of the produced by the vocal tract of humans during speech generation [45], [46]. There have been several studies on improving the accuracy of speech emotion recognition using MFCC [47]–[51]. An overview of the key steps to calculate the MFCC from an audio signal is presented below.

Mel-Frequency Wrapping. Audio signal of speech is a slowly time-varying or quasi-stationary signal, and hence needs to be examined over an optimum period, which is not too short to provide enough samples to get an accurate spectral estimate and again, not too long that the signal changes too much throughout the period. Typically, audio frame of ~20-40 ms period is considered sufficient to provide good spectral resolution of the speech [45], [46]. To ensure maximum signal continuity, the audio frames are then processed through appropriate window functions which minimize the spectral distortion of the signal. After this processing, the time-domain signal is converted into a frequency domain power spectrum using the Fast-Fourier Transform (FFT), which highlights the prominent frequency components present in a particular frame. To determine the existing energy in different frequency bands, the resultant power

spectrum is wrapped into a Mel-frequency scale. 'Mel' scale is a subjective representation of the actual frequency of a signal that relates to the perceived audio frequency by the human brain [47], [48]. The Mel-scale wrapped power spectrum is convolved with the Mel filter bank which is a series of overlapping triangular bandpass filters with center frequencies spaced in a fashion to simulate the auditory system. The output of the convolution is named as log Mel spectrum which manifests the information about the phoneme being produced by the vocal tract, in the frequency domain [48], [49], [51].

Cepstrum Coefficient. The resulting log Mel spectrum from the Mel filter bank is converted to several cepstral coefficients using Discrete Cosine Transform (DCT) in the final step, which are known as Mel Frequency Cepstrum Coefficients (MFCC). The mathematical representation of DCT of the log Mel spectrum resulting from a series of filter banks can be expressed by equation (4)[47], [48].

$$C_n = \sum_{k=1}^{G} (\log S_k) cos\left\{ n\left(k - \frac{1}{2}\right)\frac{\pi}{G} \right\}, \quad k = 1,2,\dots,K \tag{4}$$

Where, C_n is the coefficient of the acoustic vector that represents the set of MFCC and S_k is the accumulated power density of the output from kth Mel filter bank. Conventionally, each MFCC acoustic vector comprises of 13 coefficients. Some variants of the MFCC also include additional 26 features calculated from the delta (velocity) and double delta (acceleration) of the original 13 coefficients, in total 3of 9 features [49]. The 13 key coefficients of the MFCC acoustic vector incorporates the power spectral envelope of an audio segment, where the velocity and acceleration features carry information regarding the dynamics of MFCC coefficients over time. Combinedly these features provide a strong tool for speech emotion recognition. A summary of the general MFCC architecture is depicted in Figure 2.

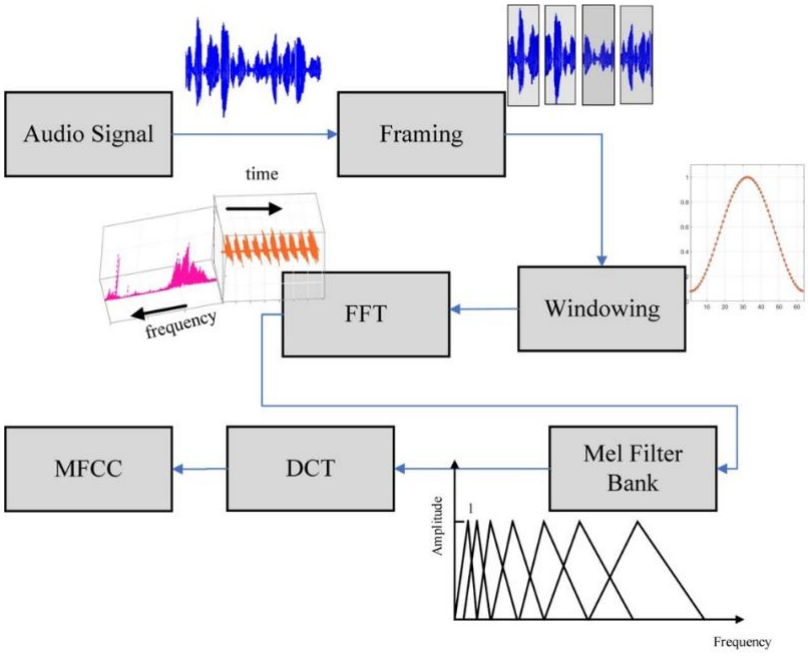


Figure 2. Block diagram of the MFCC architecture [51]

Mel Spectrogram (MEL)

Mel spectrogram is another feature that is closely related to MFCC from an operational point of view and is oftentimes used in conjunction with MFCC for speech emotion recognition [49], [51]. In the Mel spectrogram, the signal strength of an audio event is represented overtime at the 'Mel' scale. 'Mel' scale is a logarithmic transformation of a signal's original frequencies: this transformation mimics the human ear sensitivity at different frequency levels. To obtain the Mel spectrogram of audio input, Short Term Fourier Transform (STFT) is applied on overlapping windowed frames of the digitally sampled signal under test. Then the obtained spectrogram magnitudes are mapped to the ‘Mel’ scale to get the Mel spectrogram [51].

REVIEW OF CLASSIFIERS

Successful implementation of an audio emotion detection system can be achieved when an appropriate classification algorithm is combined with the extracted audio features as described in the *Audio Feature* section. Besides audio features, we must choose an appropriate classification algorithm to train the system. In this section, I am presenting a brief discussion on the different classification algorithms that are considered in this thesis to train the audio emotion detection system.

While there are several classification algorithms to choose from, the following classification algorithms are considered to present a comparative discussion of their level of accuracy in this research– KNN, Single Layer Perceptron (SLP), MLP, SVM, Logistic Regression (LR), AdaBoost, and Random Forest (RF) along with the TPOT AutoML system. Although there are many complicated models used by other researchers in the section *Literature Review,* I have chosen these simple models such as linear models like SLP, LR to reduce the time complexity. Here we are considering deploying our model in IoT devices thus the decision has to be made in real time. Also, these algorithms are some of the most frequently used algorithms to detect audio emotion. They belong to the supervised learning category because they are all fed with labeled data and are expected to return an outcome that fits within a range created by the labeled data.

K-Nearest Neighbor (KNN)

Among the aforementioned classifiers, KNN training is the simplest as it takes the training data at first, and then the sample data is compared with the training data using Euclidean distance, and finally, the closest points are selected for prediction [52]. KNN requires no

previous knowledge of the data distribution and operates by plotting all training data onto a theoretic graph using the features extracted. Then, a test sample is plotted on the graph as well, the Euclidean distance between the test sample and each training sample is then calculated and compared to find the closest point. That point's labeled outcome then becomes the predicted emotion. This is displayed in Figure 3 showing the test sample as a red star and all other training samples as the black point.

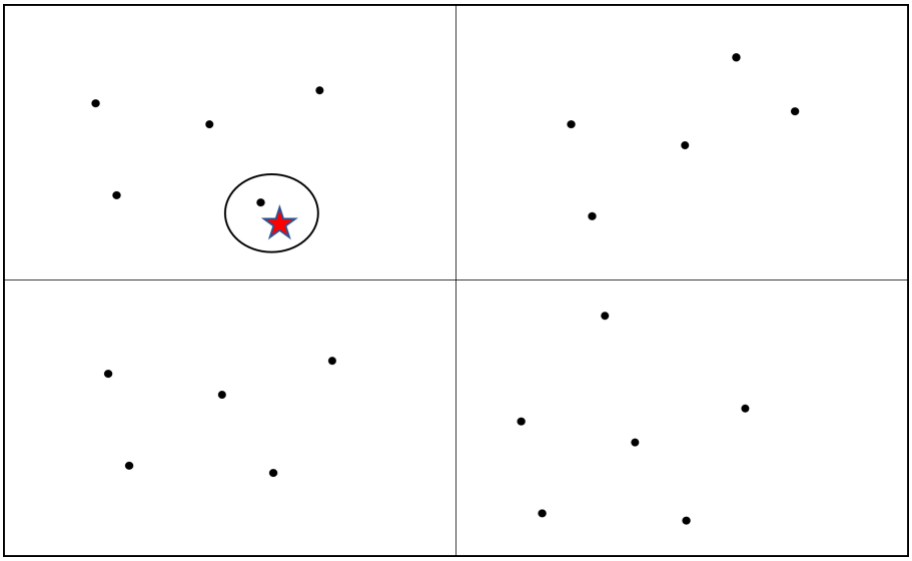


Figure 3. A schematic overview of the KNN algorithm

Single Layer Perceptron (SLP)

Contrary to KNN, SLP is trained by a labeled dataset, where the initial weights associated with the input are looped over to determine the optimum contribution of each weight that will result in the desired label as the output [53] - [35]. The overall algorithm of SLP gets more convoluted with the introduction of weights, reiteration, and the number of intermediate layers. It is also known as a perceptron, which is a system of nodes similar to neurons in the human brain that are connected by edges with weights. These edges are then used to apply the weights to the input data and this result is the prediction. Neural Network (NN) is trained by feeding in labeled

training data that is looped over to help to train the weights on the edges so that the labeled result can be reached. Figure 4 displays how the features extracted are fed into the weighted edges, which are adjusted over each training sample to produce the required outcome. The training sample is then fed into these weights and the produced outcome is the predicted emotion.

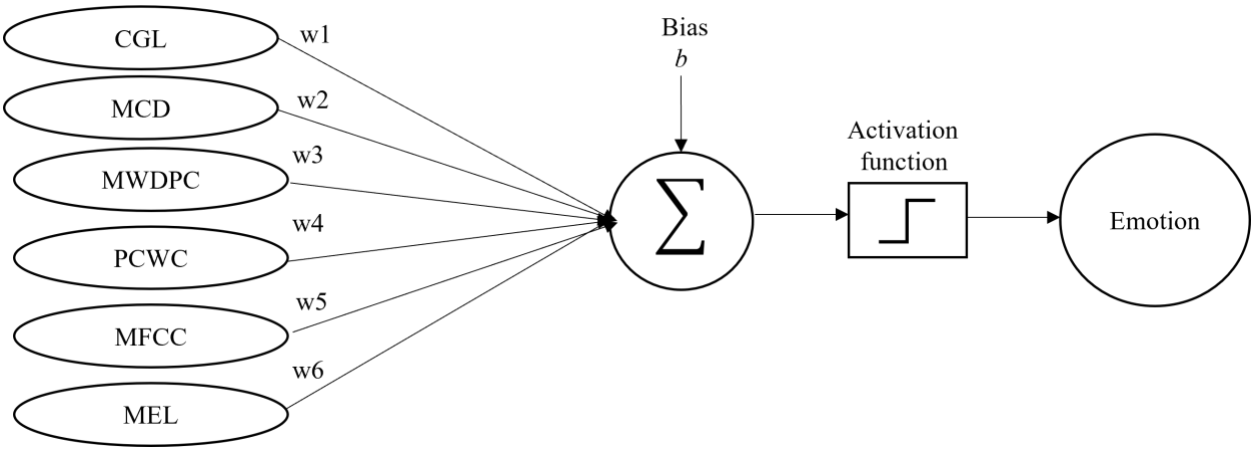


Figure 4. A schematic overview of the SLP algorithm

Multi-Layer Perceptron (MLP)

Although SLP and MLP share similar training procedures, MLP is complex due to the introduction of the multiple hidden layer and activation functions [55]. Inherently, both SLP and MLP have the potential disadvantage of either underfitting or overfitting. This happens when either too little or too much training data is supplied. It is an artificial network of interconnected nodes modeled after the human brain like NN. These neurons are connected by edges. This network is fed labeled data for training and when test data is entered into the system, it is put through multiple layers before coming to the result. Those layers are as follows: input layer, hidden layer, and output layer. The input layer is just the beginning layer that brings in the data to be examined. The hidden layer is created when inputs from the input layer are weighted, and an activation function is applied to produce an output. This output is then calculated again using

the same process of weighting and activation and the resulting number is the final output. A simple example showing MLP layers is shown in Figure 5.

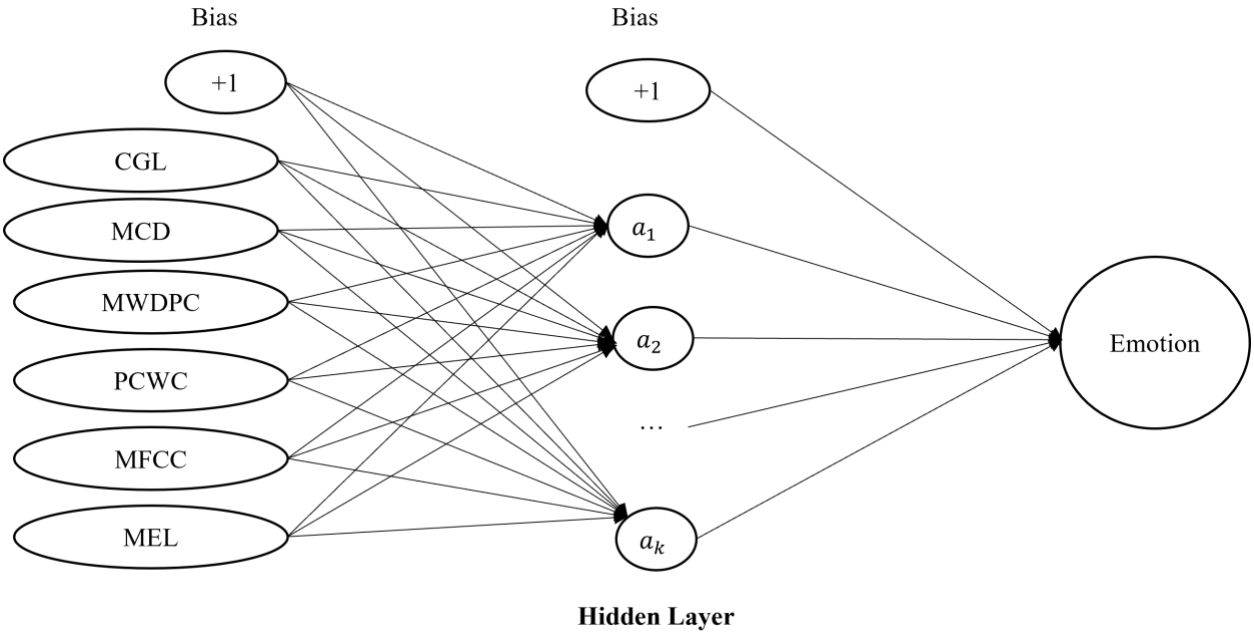


Figure 5. A schematic overview of the Multilayer Perceptron (MLP) algorithm

Support Vector Machine (SVM)

SVM is another such classification algorithm, which utilizes the binary learning model. Since the learning method of SVM is based on sorting the labeled dataset into only two categories, it provides a better result to classify binary classes [52]. The data is called support vectors. It is a binary learning model. A decision boundary is drawn to classify the data into two groups. The decision boundary is called a hyperplane. The distance between support vectors and hyperplane should be as far as possible. Then new unlabeled data is to be fed into the system based on their category the system shows predicted emotion. The SVM training process is shown in Figure 6.

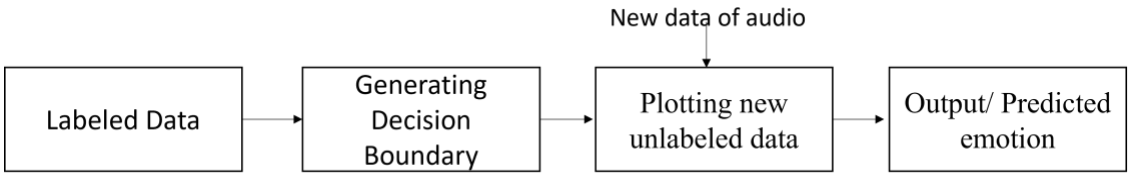


Figure 6. A schematic overview of the SVM algorithm

Logistic Regression (LR)

In comparison to the common classification algorithms, LR utilizes traditional statistics to develop an interlink between one dependent binary class and one or more nominal, ratio-level, or interval independent variables. The output from LR is provided in a form of the probability of desired outcome [54]. Due to its ability to correlate more than two variables, LR can handle multi-class classification in a better way. It is a technique that is used for traditional statistics along with machine learning. There is one dependent binary class and one or more ordinal, nominal, ratio-level, or interval dependent variables. Logistic regression develops the bonding between one dependent binary class and one or more nominal, ratio-level, or interval independent variables. LR shows the relationship between features and the probability of an outcome as shown in Figure 7. Here, Ɵ is the weight associated with different audio features which control the impact of each feature in training the algorithm.

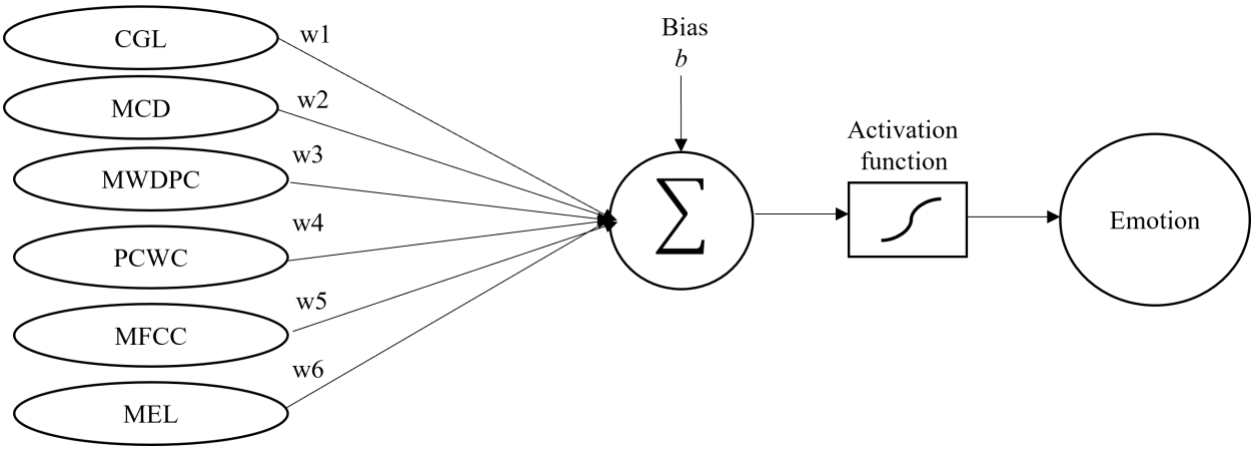


Figure 7. A schematic overview of the LR algorithm

Adaptive Boosting (AdaBoost)

AdaBoost is also called boosting technique. Here several base learner models are developed sequentially. The labeled dataset is fed into the 1st base learner model. If the 1st base learner model incorrectly classifies some of the records, then the records are fed into the next sequential layer. This process is called boosting technique. In AdaBoost, the records are fed with sample weight into the system. Then a decision tree is created with the help of one depth. It is also called stumps. One stump would be created for each record. The total error is calculated for each of the misclassified records. Then the performance and error are calculated for each stump and the weight gets updated. In AdaBoost [56], the training dataset is also fed with sample weights into the first base learner model similar to SLP and MLP. The advantage of AB is that it incorporates several base learner models sequentially, where each subsequent learner model takes the misclassified data from the previous base learner model with updated weight. As the model is trained in sequential layers with updated weight, this technique is also known as boosting technique and can effectively handle weak learners. AdaBoost training mechanism is shown in Figure 8.

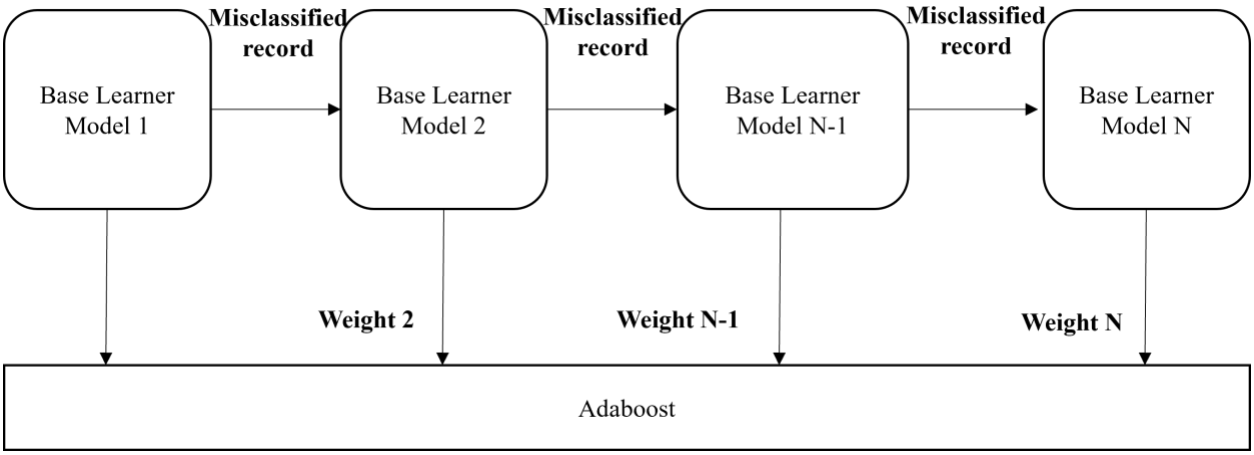


Figure 8. A schematic overview of the AdaBoost algorithm

Random Forest (RF)

RF classifier [57] also introduces several base learner models where each model is fed with a random combination of labels and features from the training dataset. When the test data is introduced to the trained model, every base learner model generates its output. The class that is predicted by the majority of the base learner models is considered the outcome. As a result, RF performs well when there are multiple combinations of labels and features. It also uses a decision tree as the base learner model. At first, some rows and some features are selected as samples called row sampling and feature sampling. Row and feature sampling are always less than the total row and features. The sample row and features are fed into the decision tree as the base learner model. This is also called bootstrapping. Then the row and feature sampling get replaced and fed into the second base learner model. Thus, all the models are fed with different row and column sampling for training. Then the test data is fed into the system and every model generates its output. The majority of the output from all the models is counted as the final output for the test data. RF training mechanism is shown in Figure 9.

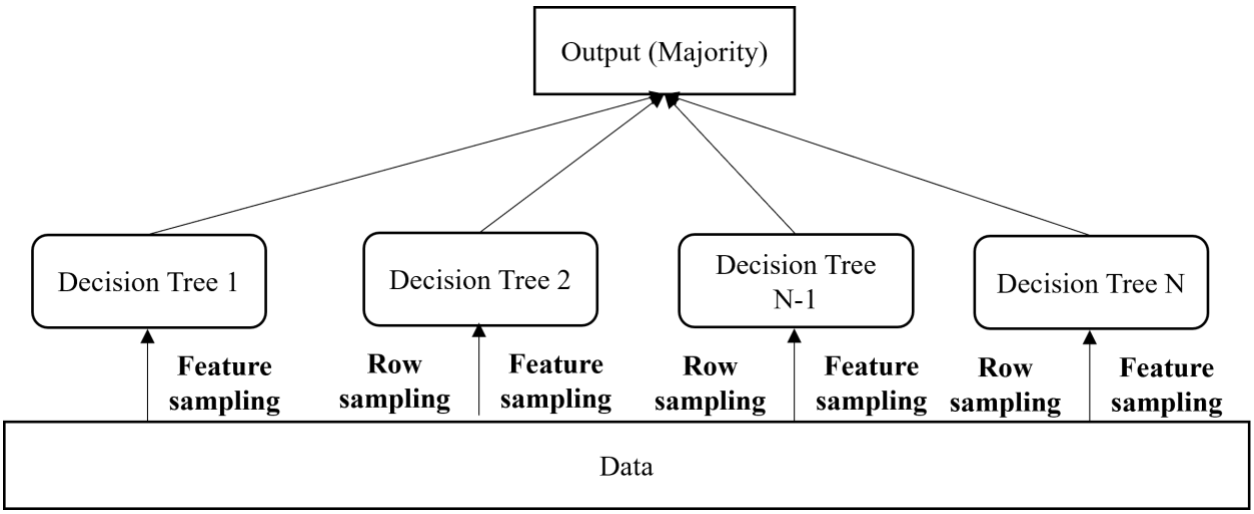


Figure 9. A schematic overview of the RF algorithm

Tree-based Pipeline Optimization Tool (TPOT)

Besides the algorithms discussed in the previous subsections, there exist a plethora of classification techniques that a system designer can choose from, such as XGBoost, DecisionTree, etc., with the additional dependency of data preprocessing and choice of the number of ideal parameters for each classifier. This tedious yet significant aspect of machine learning processes prompted the introduction of automated machine learning (AutoML) pipelines that optimizes the process of designing an effective classifier [58]. To identify the appropriate classifier, an AutoML known as TPOT is incorporated in this research, which is a python-based tool that optimizes machine learning pipelines using genetic programming. The pipelines of TPOT are arranged in a tree-like formation, where genetic programming is performed to obtain accurate predictive models [59]. Each node of the tree is known as ‘operators’ such as preprocessors, decomposition functions, feature selectors, and estimators. The pipelines comprising the ‘leaf’ node of the tree are provided with identical copies of the input dataset and the prediction accuracy of each pipeline is delivered as output to the ‘root’ node. The system then performs genetic mutation of the operators for example introducing a preprocessing step or

reducing the dimensionality of the features based on their relative importance in prediction. Some ‘operators’ such as feature selectors or model selectors, which can take input from multiple previous operators also undergoes crossover by exchanging features between different pipelines. The next generation of the programming tree with mutated operators is constructed based on the fitness score from each pipeline. Consequently, the machine learning pipelines trained by TPOT consist of a relatively small number of operators yet ensure high performance [60]. It is an AutoML (Automatic Machine Learning) tool. TPOT is an automated machine learning tool that optimizes machine learning pipelines using genetic programming. It uses sci-kit learn library of python for machine learning classifiers. TPOT combines a flexible expansion tree with genetic programming, and stochastic search algorithms to build machine learning pipelines. It reduces dimensionality and searches intelligently over machine learning pipelines containing classification models, preprocessors, feature selection techniques, and hyperparameters of all objects. It considers scikit-learn API and searches over supervised classification. algorithms associated, transformers and hypermeters with it. Figure 10 shows the TPOT mechanism.

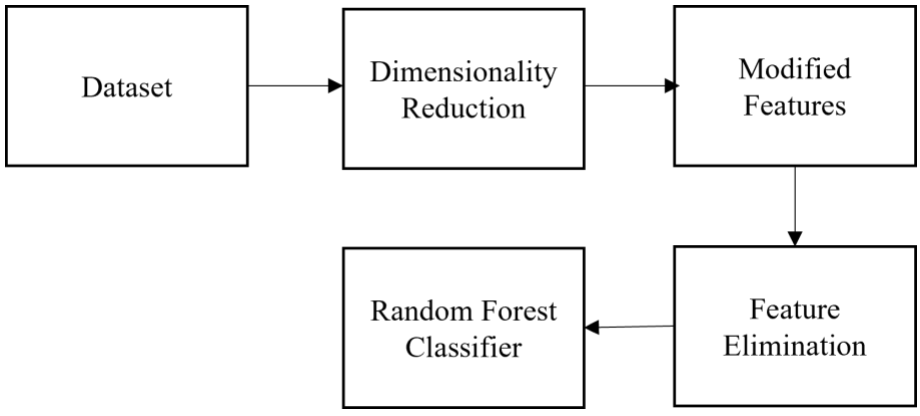


Figure 10. A schematic overview of the TPOT algorithm

To summarize the algorithms that have been discussed in this section, all of them include some advantages and disadvantages that must be considered when trying to determine which one is best for the smart home application. K-Nearest Neighbor training is the simplest in the fact that it is fed the training data, then the sample data is compared using Euclidean distance. K-Nearest Neighbor is semi-simple in the fact that it just plots the points of the training data features, and then compares the Euclidean distance between those points and the plotted test point, and then selects the closest point for prediction. NN gets to be more complex with the introduction of weights and reiteration, adding to the number of steps. Multi-Layer Perceptron is the most complex of the algorithms discussed due to the introduction of the hidden layer and activation function, requiring the algorithm to take more steps than similar algorithms like NN. NN and Multi-Layer Perceptron share similar training processes with the main difference being that the training process of Multi-Layer Perceptron is more complex due to the use of the hidden layer. NN and Multi-Layer Perceptron both have the potential problem of either underfitting or overfitting. This is when either too little or too much training data is supplied. Support Vector Machine shows a better result to classify binary classes whereas logistic regression can handle multi-class classification in a better way. To handle weak learners, AdaBoost performs well whereas random forest performs well due to its bootstrapping process.

IMPLEMENTATION

A complete workflow diagram of the proposed methodology is shown in Figure 11. First, the audio data is recorded using an audio recorder interface developed using Python. The model training is initiated by importing the audio files and splitting the entire dataset into training and test data. To train the model, first, the audio features mentioned in Section *Audio Feature*s are extracted from the training dataset. In addition to the audio features, the training dataset also consists of labels/identifiers for happy, normal, sad, angry, and fearful emotions corresponding to each of the audio files. Then the classification algorithms mentioned in Section *Review of Classifiers* are implemented to train the system models by aligning the feature attributes to a specific label. For each trained classification model, the accuracy of that model is tested by calculating its ability to detect appropriate emotions from the test dataset, using the same audio features.

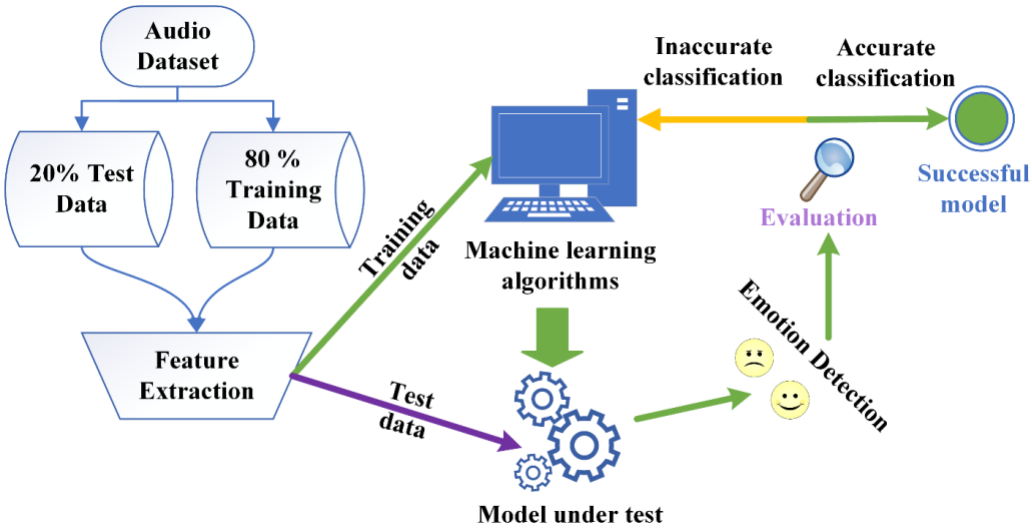


Figure 11. Emotion detection workflow diagram

Each model predicts the emotion label of each audio file in the test dataset based on its learning from the training dataset by analyzing the feature attributes. Then, the predicted emotions are matched with the actual emotions, and a confusion matrix of true positives, false positives, true negatives and false negatives are created. The accuracy of each model is then determined from the confusion matrix based on the number of false positives and true positives.

With the choice of audio features and classification algorithm, the next step is to implement the combination of audio features and classification algorithm to the verify audio dataset for system model training and test purposes. The dataset must provide enough quality samples that convey the correct emotion. There are many existing audio datasets, and they have a lot of emotions and various audio features that can be extracted from them. Primarily, the proposed methodology was implemented on the RAVDESS dataset [25] and TESS dataset [26] to train and test the model. In this section, details on these two datasets are provided along with the preparation of our in-house dataset which is solely based on the voice command used in the smart home environment.

Audio Dataset

RAVDESS. To identify a suitable classification algorithm amongst the classifiers reviewed in the section of *Review of Classifiers*, the RAVDESS dataset to train and test the audio emotion detection model is utilized at first [25]. This dataset is open source, publicly available, and has been widely used to detect speech emotion from the audio conversation, e.g. [23], [62]. It consists of 1440 files with 60 audio recordings in English from 24 professional actors (12 male and 12 female), vocalizing in a neutral North American accent. There are 8 distinct emotions in the dataset, which are normal, calm, happy, sad, angry, fearful, disgust, and surprised. However, only normal, happy, angry, sad, and fearful emotions are chosen among the 8 emotional states in

RAVDESS dataset for my study. The resultant dataset considering the three emotions consisted of 800 audio recordings.

TESS. The second publicly available dataset is Toronto Emotional Speech Set (TESS) [26]. In this dataset, 2 actresses aged 26 and 64 years old spoke 200 target words. There are 2,800 files in total consisting of 7 emotions which are happiness, anger, disgust, fear, pleasant, surprise, neutral, and sadness. For this research, only 5 emotions named happiness, anger, fear, neutral, and sadness are considered, consisting of 2,000 files. The summary of the data used from the RAVDESS and TESS dataset is tabulated in Table 2.

Table 2. Considered Data from RAVDESS and TESS dataset

Dataset	**RAVDESS**	**TESS**
Speaker	12 males, 12 females	2 females
Age range	21-33	32
Emotions	Normal, happy, angry, sad, fear	Neutral, happy, angry, sad, fear
Audio Files	800	2000

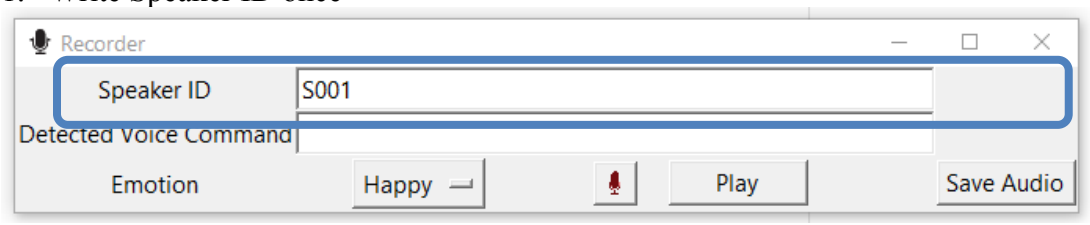
Smart Home Dataset. One of the shortcomings of the existing audio dataset is that they are based on scripted dialogues [25] or movie clips [32], which are impractical in a smart home context, especially when the used voice commands that dictate the operation of smart appliances or communicate with the smart assistants consist of only a few words. To date, there has not been any report of a smart home voice command dataset to the best of my knowledge. Hence, I created an Institutional Review Board (IRB) approved smart home voice command dataset for this research. The commands are concise and do not exceed a total word count of 6. The dataset consists of 50 voice commands consisting of 2-6 words per command as tabulated in Table 3.

Table 3. Smart voice commands from the smart home dataset

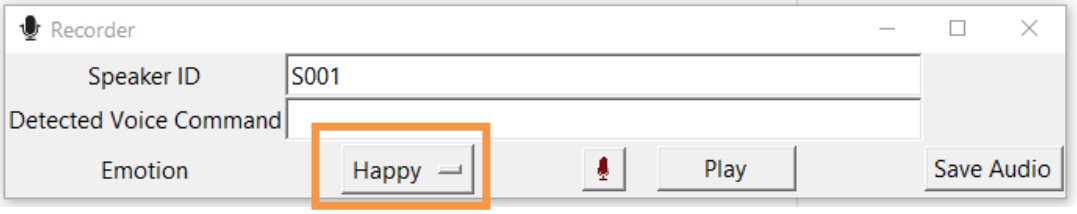
Command no.	Command	Command no.	Command
1	Go home	26	How’s the weather tomorrow?
2	Dial 911	27	Is it cold today?
3	Call Mom	28	Cancel the 2pm alarm
4	Spell Banana	29	Turn the volume down
5	Call Police	30	Set temperature to 75
6	Play music	31	What's the best operating system?
7	Cancel alarm	32	Set an alarm for 8am
8	Show camera	33	Cancel all the alarms today
9	make coffee	34	What is my flash briefing
10	lock doors	35	Why are you so annoying?
11	Make it warmer	36	Remind me to do laundry.
12	Play the news	37	When is my son's birthday?
13	Read my calendar	38	What do you think of siri?
14	I am sad	39	How's traffic today in Springfield?
15	turn on light	40	I am not happy today
16	turn off fan	41	What is on my calendar today?
17	Change your voice	42	Are you afraid of the dark?
18	Who are you?	43	What is the value of Pi
19	Lower the temperature	44	What is zero divided by zero
20	Set 10minutes timer	45	Is it going to snow today?
21	Tell me a joke	46	How will be the weather at night?
22	Sing me a song	47	Who’s the president of USA now?
23	Tell me a story	48	What’s the Italian of “Good morning”
24	Read me a haiku	49	How’s the weather of Bangladesh now?
25	What’s the weather today?	50	Do not answer to my kid

As can be seen from Table 3, these are some of the most used voice commands in daily smart home context and do not contain any specific lexical cues about the emotion of a user by itself. Here, all the samples were recorded using the audio recorder interface. Python Tkinter and Google speech recognition library was used to develop the audio recorder. I used CMTeck USB Computer Microphone, Plug & Play Desktop Omnidirectional Condenser PC Laptop Mic to record the audio samples from each participant. As I invited different age ranges of native as well as non-native speakers, I wanted to make sure the recorded emotional utterance was annotated and validated. Thus, in the designed audio recorder interface, when the user selects particular emotion and records it, the speech recognition library converts the audio to text and displays the "Detected Voice Command". To make sure the recordings are unbiased and not influenced by the aim of this study, I have given control to the user of choosing if a specific version of the recorded voice commands should be included in the dataset. By observing the detected voice command, the user can assess if the audio data has been recorded correctly and choose to accept or reject the "Save Audio" step (i.e. Step 8). Also, the user can listen to the recorded command by clicking on the "Play" button before saving the file or re-recording the audio data if it is unsatisfactory. Each audio file in the dataset is saved in the folder named as the selected emotion and uses the following naming convention: <Speaker ID>_<detected command>_<serial number>.wav. This ensures every user recorded each command correctly and 2 separate times for each emotion. The reason for choosing to record twice for each emotion is validated further in the *Performance Evaluation* subsection. Also, to validate my recorded dataset, I went through each of the recorded command and requested the user to rerecord if needed. The instruction to interact with the audio recorder interface are given below:

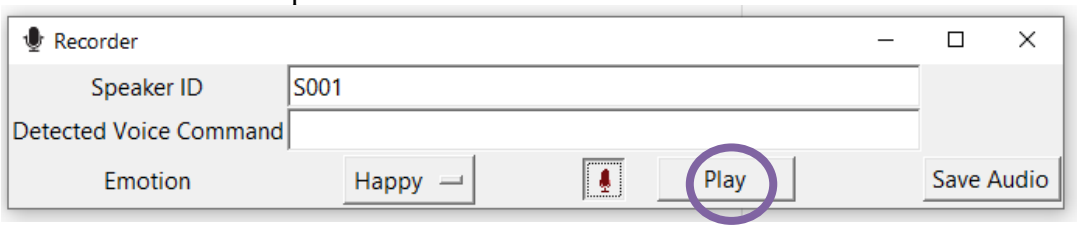
1. Write Speaker ID once



2. Select emotion from the drop-down menu. Click on it to select the other emotion

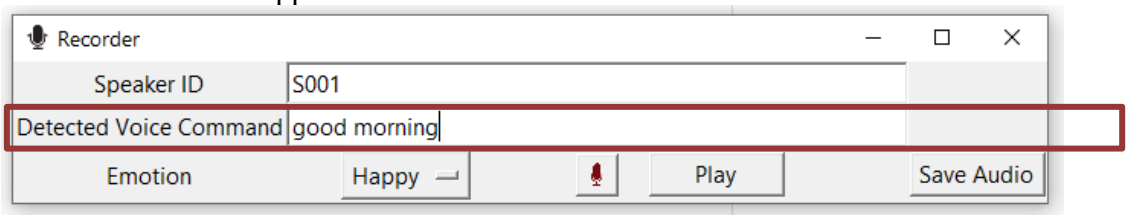


3. Click on the Microphone button

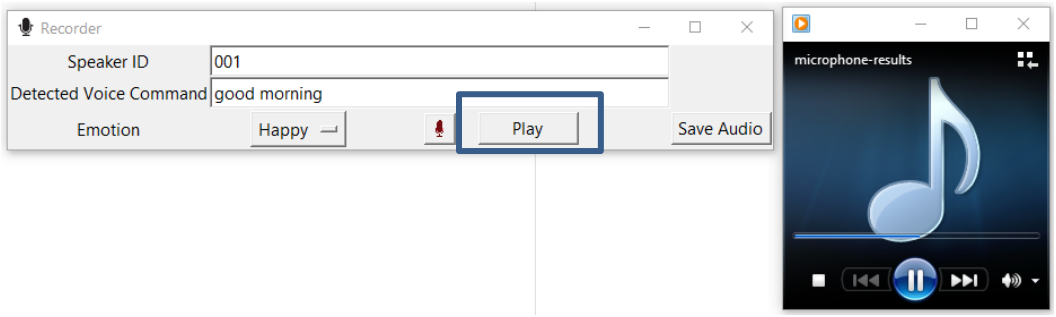


4. Speak the voice command in your selected emotion

5. See the text that appeared in the "Detected Voice Command" is correct or not

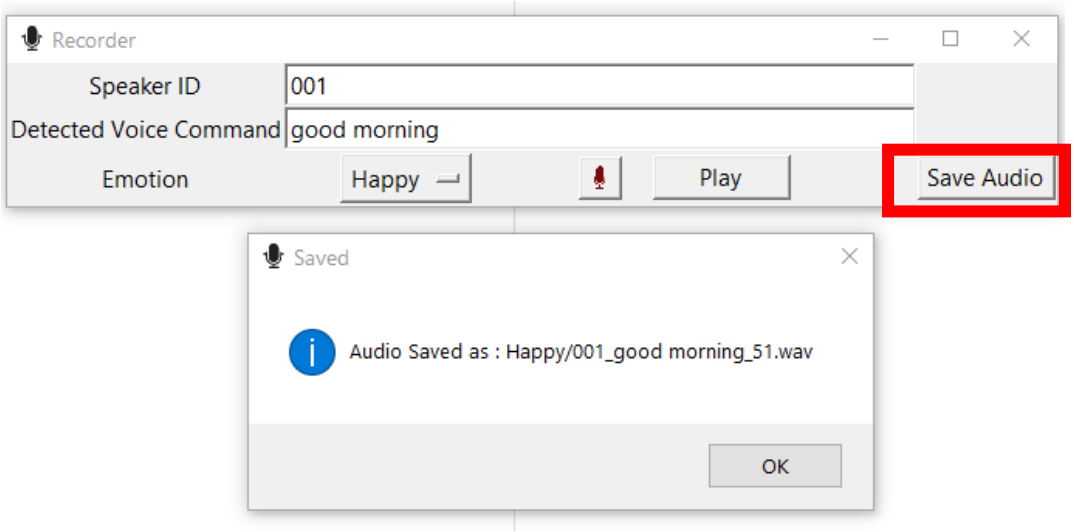


6. Play (if you want to listen) after unmuting the computer's speaker



7. Keep the Computer's speaker muted

8. Click on “Save Audio”. It will show the saved information.



9. Thank you. Please start following instructions 2-9 for the next commands.

To prepare the custom dataset of smart home voice commands, I invited 10 volunteer speakers whose demographics are given in Table 4.

Table 4. Smart home dataset description

ID	Gender	Age Range	Native English Speaker
S001	Female	15-20	Yes
S002	Male	21-25	Yes
S003	Male	21-25	No
S004	Nonbinary	21-25	Yes
S005	Female	15-20	Yes
S006	Male	31-35	No
S007	Male	36-40	No
S008	Female	26-30	No
S009	Male	21-25	Yes
S010	Male	21-25	Yes

The final dataset consists of 5000 voice commands imitating happy, normal, angry, sad, and fearful emotional states. Only these 5 specific emotions are considered in this study as researchers have suggested that invoking these emotions would result in distinguishable variation in audio features of human conversation [27]. Each participant recorded each of the 50 commands 2 separate times using 5 different emotions in indoor environments in one or more recording sessions.

Feature Extraction

The audio data for our in-house dataset on smart home voice commands is first recorded in *.wav* format and lossless mono channel using an audio recorder interface designed for this study. Afterward, the recorded data is converted into bytes and an open-source audio activity detection python tool named *auditok* is used to calculate the root means square energy (RMSE) of single-channel mono audio data according to equation 5 [61]:

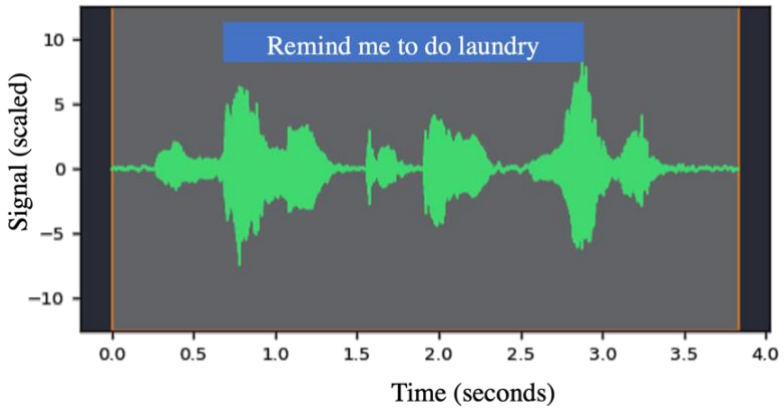
$$RMSE = 20 \log \left(\sqrt{1/N \sum_r^N a_r^2}\right) \quad (5)$$

Here, a_r is the r-th sample from the digitally converted audio file and N is the number of samples in the dataset. The RMSE is represented with a logarithmic operation to ensure better resolution for defining noise threshold during audio activity detection. Based on the audio activity, four new features named CGL, MCD, MWDPC, PCWC are extracted for each audio file and then combined with existing MFCC and MEL features for emotion classification. The four new audio features that are being considered in this thesis are directly correlated with the voice activity detection in smart home environments. To split input audio into segment of voice activity and silence period, I used four specific parameters- namely minimum_duration, maximum_duration, maximum_silence, drop_trailing_silence, and analysis_window thresholds.

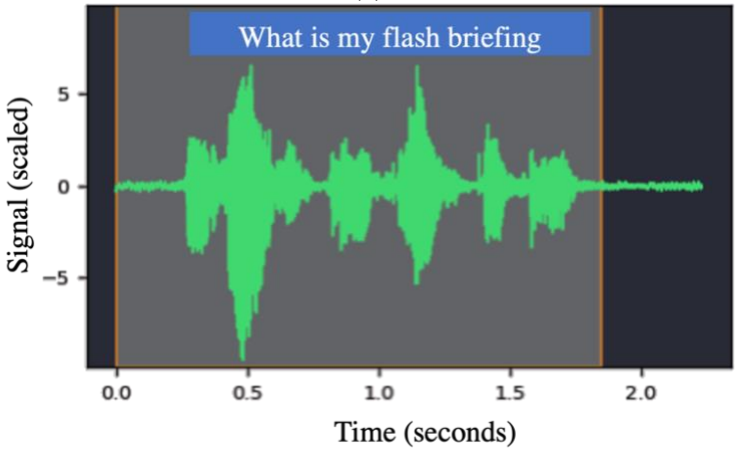
Minimum and maximum duration are important to properly detect and record a voice activity event. For example, a high value for minimum duration might result in recording a lot of noise whereas the audio will be shortened if the audio length is more than the maximum duration. The maximum_silence is the maximum duration of uninterrupted silence in recorded audio for it to be considered as a single voice activity event. Drop_trailing_silence is used to remove the trailing silence from each audio chunk. Finally, the analysis_window is the duration of audio analysis in seconds.

Chunk Gap Length (CGL). In Algorithm 1, the CGL feature extraction steps are presented. In the implementation, minimum_duration is set to 0.1s, maximum_duration is set to 5s, maximum_silence is set to 0.01s, analysis_window is set to 0.01s, and drop_trailing_silence is set to True (i.e. enable). As there is practically no silence period (i.e., with absolute zero signal strength) in a natural audio signal, an energy threshold is defined as silence while extracting CGL and MCD features from the input audio. The energy threshold for silence determines the specific segment of audio activity that is considered the silence period. As the audio data was recorded using a 16-bit or 2-byte binary encoding scheme (default bit rate for mono-channel audio recording in .wav format), the maximum energy threshold of an audio event was set to the RMSE value calculated using equation 5 while considering every 2-byte of the digited audio signal as one sample. Considering the maximum energy level to 0 dBFS (decibel related to full scale), the silence threshold was set to -10 dBFS which corresponds to approximately one-third below the maximum signal strength ($20*\log(1/3) \approx -10$ dB). An intuitive representation of the energy threshold and corresponding detected audio activity is shown in Figure 12 to Figure 14, where 3 different signal thresholds are shown to detect 3 different audio data.

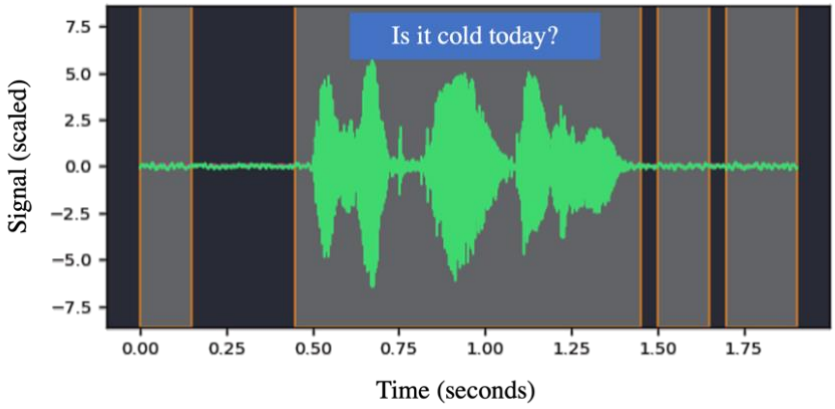
If 1 byte of data is chosen as an energy threshold parameter, the entire audio event is detected as 1 chunk shown in Figure 12.



(a)



(b)



(c)

Figure 12. Utterance detection of mono audio data when energy threshold = calculate energy single channel (data,1)

If 2 bytes of data are chosen to calculate the signal strength, then the correct audio events cannot be distinguished appropriately as shown in Figure 13.

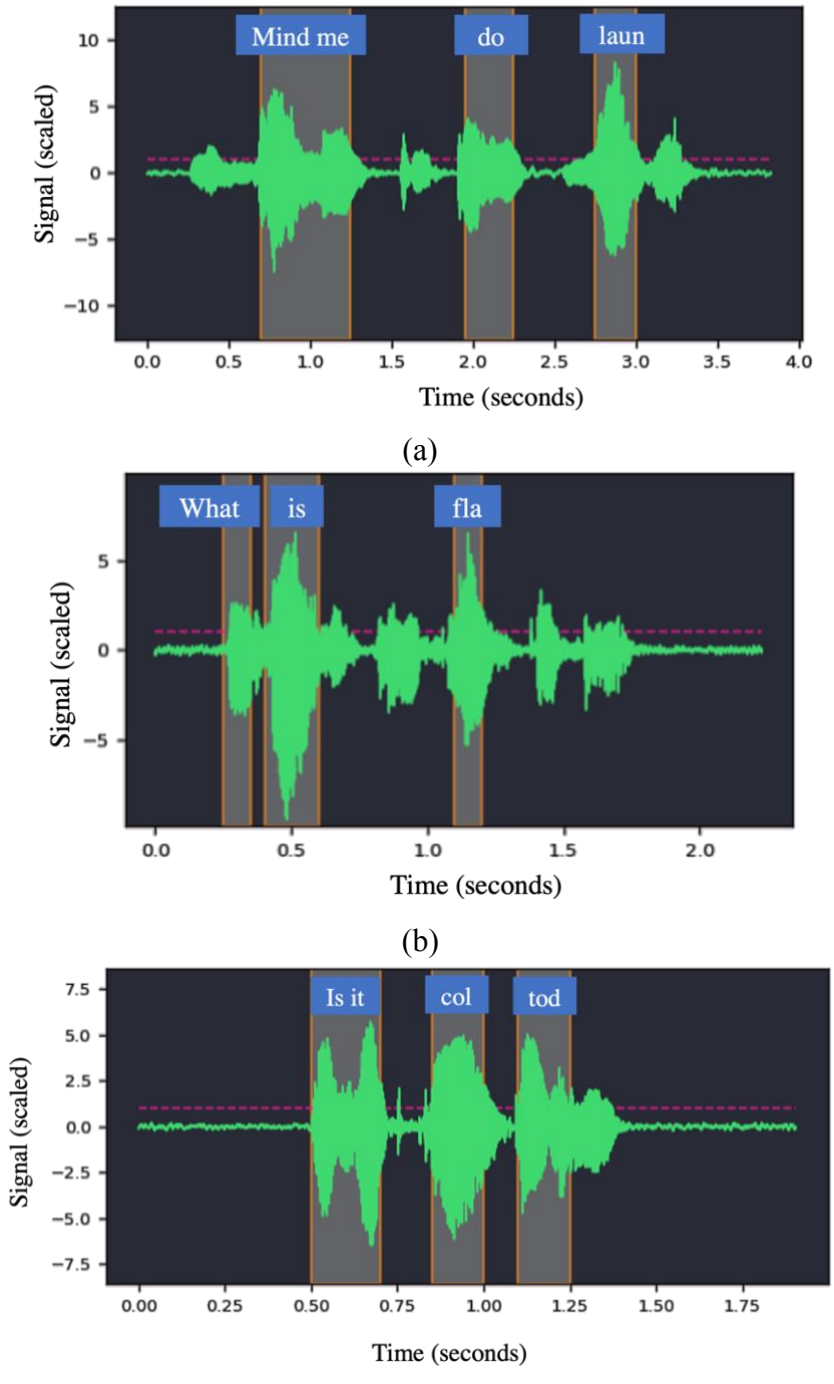
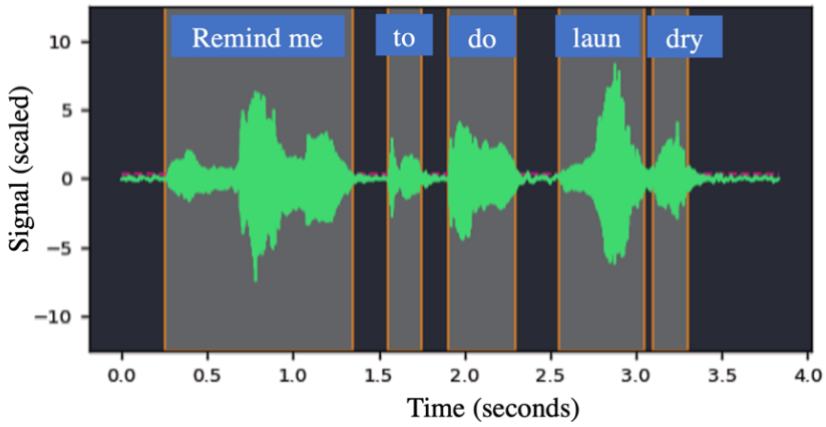
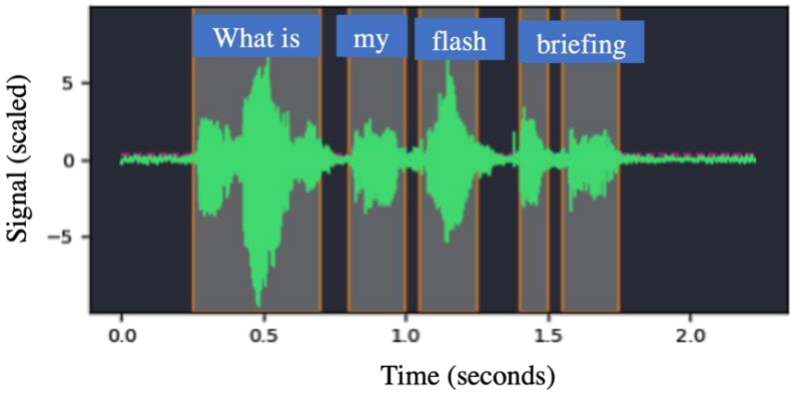


Figure 13. Utterance detection of mono audio data when energy threshold = calculate energy single channel (data,2)

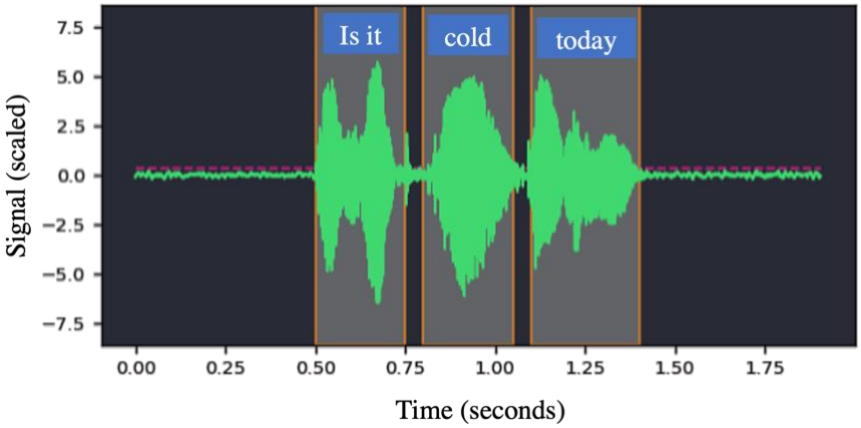
In comparison, as shown in Figure 14, if the energy of 2 bytes-10 dBFS is defined as the noise threshold all significant audio activity can be detected accurately.



(a)



(b)



(c)

Figure 14. Utterance detection of mono audio data when energy threshold = calculate energy single channel (data,2)-10

The interval of each silence period is measured from the time duration when the signal strength is below the noise threshold between two chunks. Also, when a word consists of any soft-spoken syllables and the associated signal strength in such a scenario drops below the noise threshold. For example, the word 'temperature' may be spoken in four syllables (tem.per.a.ture) by many non-native speakers, but the native pronunciation has only three syllables ('temp(ə)rəCHər'), where the '(ə)' syllable is considered as silent phonetics. In such situations, the continuity of the audio signal produced by that specific word will be broken if the soft-spoken syllable falls below the noise threshold and will affect the feature extraction accuracy. To avoid such circumstances, the minimum duration is set to 0.1s. Finally, the CGL feature for a specific audio input is measured from the mean duration of the silence periods.

Input: Audio file
Result: Chunk Gap Length (CGL)
Procedure: Split audio into segments as per following parameters;
    Minimum audio event duration = 100 ms;
    Maximum audio event duration = 5000 ms;
    Maximum continuous silence = 10 ms;
    Duration of analysis window = 10 ms;
    Signal energy threshold = 2 bytes - 10;
for *segments in audio file* **do**
 | Compute start and end time stamp of each segment
end
Create an array of time intervals between two consecutive audio segments by subtracting the end time of a previous segment from the start time of the later segment;
Calculate sum of the time interval array;
Return the sum of interval as CGL

Algorithm 1. Chunk Gap Length Extraction

Therefore, for the results reported in this thesis, any audio signal having a minimum duration of 100ms, a maximum duration of 5000ms duration of an analysis window of 10ms, and continuous silence of 10ms are considered audio activity, and the rest is considered silence. It is not practically comprehensible to set a fixed threshold for silence as the definition of noise may vary depending on the recording environment, ambient noise, and user historical data. Hence, we set a dynamic range of threshold which is approximately one-third or 10 dBFS below the maximum signal strength of each audio event and validated through several demonstrations that our assumption for silence threshold is valid in smart-home environment. However, the threshold may vary based on the application and recording environment.

Mean Chunk Duration (MCD). A pictorial depiction of the chunks in a sample audio file is shown in Figures 12 to Figure 14. Here, Figure 12 shows that if 1 byte of data width is chosen to calculate the energy threshold, then the whole command is taken as one audio chunk. If 2 bytes of data widths are chosen then the first and last chunks i.e., “Re” of “Remind” and “dry” of “laundry” are considered silent, as shown in Figure 13. But if the silence threshold is set to the energy of 2bytes -10 dBFS, then all the audio chunks can be detected correctly. Here, the command is divided into 5 chunks which are, “Remind me”, “to”, “do, “laun”, and “dry”. Similarly, the minimum threshold is set to 0.1s, max_dur is set to 5s, max_silence is set to 0.01s, analysis_window is set to 0.01s and the drop_trailing_silence is set to True.

In Algorithm 2, the MCD feature extraction steps are depicted. Based on the same concept described in the CGL implementation, the time difference between two successive time stamps of silences will correspond to the time interval of each chunk of words. The audio

Input: Audio file
Result: Mean Chunk Duration (MCD)
Procedure: Split audio into segments as per following parameters;
    Minimum audio event duration = 100 ms;
    Maximum audio event duration = 5000 ms;
    Maximum continuous silence = 10 ms;
    Duration of analysis window = 10 ms;
    Signal energy threshold = 2 bytes - 10;
for *segments in audio file* **do**
  | Append segment duration in a time array
end
Calculate mean of time array;
Return mean time as MCD

Algorithm 2. Mean Chunk Duration Extraction

recordings are separated into segments corresponding to the chunks of words in a voice command. To extract the MCD feature, then the mean time interval of the chunks of an input audio file is measured.

However, there is a probable chance of error if a word consists of any soft-spoken syllables and associated signal strength in such a scenario drops below the noise threshold. For example, the word 'temperature' may be spoken in four syllables (tem.per.a.ture) by many non-native speakers, but the native pronunciation has only three syllables ('temp(ə)rəCHər'), where the '(ə)' syllable is considered as silent phonetics. In such situations, the continuity of the audio signal produced by that specific word will be broken if the soft-spoken syllable falls below the noise threshold and will affect the feature extraction accuracy. To avoid such circumstances, I have incorporated a timing interval threshold determined by several trials for the model which ensures that if the time difference between two successive walls of silence is less than the minimum silence length then it will be considered a time segment of the previous word. If the

time difference between two successive walls of silence is less than the minimum duration, then it will be considered as a time segment of the previous word.

Mean Word Duration Per Chunk (MWDPC). In Algorithm 3, the MWDPC feature extraction steps are shown. To detect MWDPC, the split chunks of words for each command detected by MCD are divided by the the number of words in each chunk. The number of words in each chunk is detected by Google Speech Recognition tool. To avoid division by zero error, it is assumed that each chunk will have at least one word. Here, if speech recognition cannot detect any word, then it will count the word as 1 to avoid zero division error.

Input: Audio file
Result: Mean Word Duration Per Chunk (MWDPC)
Procedure: Split audio into segments as per following parameters;
    Minimum audio event duration = 100 ms;
    Maximum audio event duration = 5000 ms;
    Maximum continuous silence = 10 ms;
    Duration of analysis window = 10 ms;
    Signal energy threshold = 2 bytes - 10;
for *segments in audio file* **do**
    Convert audio segment to text (speech to text);
    Calculate number of words in the text;
    Calculate word duration in a segment;
    Append per word duration in a time array;
end
Calculate mean of time array;
Return mean time as MWDPC

Algorithm 3. Mean Word Duration Per Chunk Extraction

Per Chunk Word Count (PCWC). As shown in Algorithm 4, the average number of words per chunk is calculated to detect per chunk word count. The threshold values for different

parameters remained the same as CGL and MCD to detect chunks of words and to count the number of words in each chunk.

Input: Audio file
Result: Per Chunk Word Count (PCWC)
Procedure: Split audio into segments as per following parameters;
    Minimum audio event duration = 100 ms;
    Maximum audio event duration = 5000 ms;
    Maximum continuous silence = 10 ms;
    Duration of analysis window = 10 ms;
    Signal energy threshold = 2 bytes - 10;
for *segments in audio file* **do**
    Convert audio segment to text (speech to text);
    Calculate number of words in the text;
    Append number of words per segment in an array;
end
Calculate mean of word count array;
Return mean silence interval as PCWC

Algorithm 4. Per Chunk Word Count Extraction

Finally, to extract MFCC and MEL from the audio recordings, the python 'librosa' library is incorporated. Each input audio file is subdivided into small frames of 20-40ms for MFCC feature extraction. The first 40 MFCCs are considered for each frame in this experiment. To extract the Mel spectrogram, a "Hann" window of 2048 length is used with an STFT hop-length of 512 and the resultant spectrogram coefficients were then mapped to the Mel scale. The recordings were sampled at a rate of 48 kHz, which yields a 40 ms audio frame with 10 ms hop length (duration of frame = number of samples x (1/sampling rate)). This is in well accordance to the generic values used for windowing as stated in *Mel Frequency Wrapping* subsection of section *Review of Classifiers*.

Classification Algorithms

After extracting the audio features, I used 80% the of the dataset as training data and 20% of the dataset as test data to train our model by incorporating the classification algorithms mentioned in the *Review of Classifiers* section. To ensure the performance improvement is unbiased and no by chance, I maintained the same training and test data from each dataset. Python-based *sklearn* library was used to implement KNN, SLP, MLP, SVM, LR, AdaBoost and RF classifiers. For implementing TPOT, I used *tpot* library. Each of the classifiers have specific control parameters based on their respective algorithm and the classification accuracy also depends greatly on the selection of the control parameters. Here, I am providing a brief overview of the control parameters I employed for this study:

For KNN, I imported *KNeighborsClassifier* package from *sklearn* module and considered 5 neighbors in accordance with the number of emotions considered in our study. I used *Perceptron* package from *sklearn* library for the SLP classifier, where the maximum iteration number was to set to 10000000. For MLP, I employed *MLPClassifier* package from *sklearn* library, where I considered 10 hidden layers, logistic activation function with learning rate of 0.01 and maximum iteration of 100000. I imported *svm* package from *sklearn* library to implement SVM, where I considered linear kernel. For LR, I imported *LogisticRegression* package from the same *sklearn* library and considered *liblinear* solver with 0 random state. To implement *AdaBoost*, I imported *LabelEncoder* and *AdaBoostClassifier* packages with maximum depth of 1 and 200 estimators. I imported *RandomForestClassifier* from the *sklearn* library to implement RF, where I considered maximum depth of 2 with 0 random state 0. Finally, to implement TPOT, I imported *TPOTClassifier* package from the *tpot* module, where I chose 5 generations, population size of 20 and 42 random state.

PERFORMANCE EVALUATION

To evaluate different classifiers' performance, I have considered the confusion matrix for multiclass classification [63]. The confusion matrix compares the actual label with the predicted label for each of the machine learning models and is used to define the performance of a classification algorithm. The key performance indicator of a classifier that is considered most important in the performance evaluation is accuracy. There are several performance metrices that can be calculated from the confusion matrix to assess the accuracy of an algorithm. The most common parameters of accuracy are F-1 score, precision, and recall. The ratio of correctly predicted positive records to all the records of the real class is known as recall whereas precision shows the ratio of correct positive predictions to the total predictions. F-1 score shows the test accuracy, and it is determined by the function of recall and precision. To report the accuracy of each classifier, I utilized the F-1 score as the performance evaluation parameter.

In the context of audio emotion detection, accuracy is the algorithm's ability to correctly predict the emotional state of test samples. The very first demonstration of classification accuracy in the form of F-1 score is tabulated in Table 5, which highlights the justification behind choosing to record each command 2 times from the same users. Here, I applied TPOT on custom smart home dataset extracting MFCC feature where I considered 80% data as training data and 20% data as test data without any cross validation. As can be seen from the table, if I consider 2 recordings per command from one user, it constitutes a dataset with more accuracy than a single recording per command. So, the optimum number of recordings per command for each user is 2. For brevity and computational advantage, I have only considered the MFCC feature and TPOT classifier for generating Table 5.

Table 5. Experimental results of different times of recorded dataset

Feature	ID	$\textbf{accuracy}_{\textbf{TPOT}}$ 1 time	$\textbf{accuracy}_{\textbf{TPOT}}$ 2 times
MFCC	S001	77%	80%
	S002	83%	85%
	S003	62%	64%
	S004	90%	93%
	S005	81%	85%
	S006	86%	88%
	S007	94%	96%
	S008	83%	87%
	S009	94%	95%
	S010	88%	94%

As different classifiers have different capabilities as reviewed in section *Review of Classifiers*, it is pivotal to analyze their performance for the chosen dataset based on the selected features. The primary dataset that is considered is a modified version of the RAVDESS dataset with 800 recordings for five specific emotions - normal, happy, sad, fearful, and angry. The model is then trained and tested by splitting the dataset into 80% training data and 20% as testing data. Then the TESS dataset is chosen to identify suitable classification algorithms and audio features. I extracted the MFCC features from the modified RAVDESS dataset and incorporated the 7 classifiers reviewed in section Review of Classifiers, along with the TPOT AutoML system. In Table 6, the values corresponding to the parameters associated with system accuracy (i.e., F-1 score, Precision, Recall, and Support) for all the classifiers for MFCC only are presented. From Table 6, we can see that TPOT is associated with the highest F-1 score (72%) for the XGBoost algorithm. XGBoost is one of the machine learning pipelines incorporated in the TPOT system, which implements a gradient boosting algorithm to improve prediction

accuracy. Other traditional classifiers such as KNN, SVM, LR, SLP, AdaBoost, ML, RF were shown to have F-1 scores of 55%, 64%, 54%, 18%, 43%, 61%, and 46% respectively for the modified RAVDESS dataset for MFCC only.

Table 6. Experimental results of emotion detection for different classifiers using MFCC feature for RAVDESS dataset

Algorithm	**F-1 score**	**Precision**	**Recall**	**Support**
KNN	55%	55%	57%	96
SVM	64%	65%	63%	96
LR	54%	53%	52%	96
SLP	18%	37%	32%	96
TPOT	**72%**	73%	72%	96
AdaBoost	43%	44%	44%	96
MLP	61%	44%	39%	96
RF	46%	47%	48%	96

I extracted 5 existing features (MEL, MFCC, Chroma [64], Contrast [65], and Tonnetz [65]) and used their combinations on the selective RAVDESS dataset and applied the 7 classifiers reviewed in Section Review of Classifiers, along with the TPOT AutoML system. In Table 7, the F-1 scores for each classifier and the associated parameters are tabulated. As there can be $2^5 = 32$ unique subsets, only the combination of significant results is shown in Table 7.

Table 7. Experimental results of emotion detection for different classifiers using the RAVDESS dataset

Features	KNN	SVM	LR	SLP	TPOT	AdaBoost	MLP	RF
MEL	39%	37%	35%	32%	63%	50%	39%	38%
MFCC	55%	64%	54%	18%	72%	43%	61%	46%
Chroma	32%	24%	24%	26%	33%	29%	27%	27%
Contrast	40%	28%	26%	27%	39%	28%	32%	33%
Tonnetz	21%	18%	20%	22%	25%	21%	24%	24%
MEL, MFCC	53%	56%	59%	43%	**73%**	41%	18%	38%
MEL, MFCC, Chroma	53%	57%	58%	29%	68%	41%	73%	38%
MEL, MFCC, Contrast	53%	61%	61%	23%	72%	39%	67%	39%
All	53%	60%	60%	46%	71%	43%	61%	38%

From Table 7, we can conclude that TPOT gives the highest F-1 score of 73% for the MEL, MFCC combination utilizing the XGBoost algorithm, which is one of the machine learning pipelines incorporated in the TPOT system. Other traditional classifiers such as KNN, SVM, LR, SLP, AdaBoost, MLP, RF were shown to have F-1 scores of 53%, 56%, 59%, 43%, 41%, 18%, and 38% respectively for the modified RAVDESS dataset for MFCC and MEL. Similarly, in Table 8 we can observe that extracting MEL and MFCC using TPOT shows the best result for the TESS dataset i.e., 99%.

Even though MFCC and Mel spectrogram is eventually extracted from the signal strength associated with different frequency components in the audible range, the fundamental difference between those two is MFCC describes the power spectral envelope of an audio signal and provides insight into the rate of change of features which is pivotal in Automatic Speech Recognition (ASR), whereas MEL helps distinguish a specific emotion as it is a visual representation of human perception of audio events. One interesting fact about human perception of sound is it perceives audio differently at different frequency level. For example, the difference

Table 8. Experimental results of emotion detection for TPOT classifiers using the TESS dataset

Features	**TPOT**
MEL	97%
MFCC	98%
Chroma	88%
Contrast	90%
Tonnetz	72%
MEL, MFCC	**99%**
MEL, MFCC, Chroma	95%
MEL, MFCC, Contrast	95%
All	97%

of sound in 10 kHz and 11 kHz is not as prominent to human ear as the difference is sound at 100 Hz and 1kHz, even though the difference in frequency is same. This is most accurately captured by Mel spectrogram where the signal strength of an audio event is represented over time at the 'Mel' scale which is a logarithmic transformation of the signal's original frequencies. Hence, hypothetically, the combination of these two features should lead to better accuracy in speech emotion detection and that is what we can observe from the results shown in Table 7 and Table 8. Thus, to validate the impact of custom audio features, this research combines the custom audio features along with the best combination of existing features of MEL, MFCC.

To validate the model for the short voice command dataset, a smart home dataset recorded by user S007 is randomly chosen and associated audio features were extracted for the 8 classifiers in Table 9. Based on the results presented in this table, we can see that TPOT again outperforms other classifiers for all the combinations of features. Therefore, TPOT is selected as my de facto classifier.

Table 9. Experimental results of emotion detection for different classifiers using the smart home dataset (S007)

Features	KNN	SVM	LR	SLP	TPOT	AdaBoost	MLP	RF
MEL	87%	81%	84%	67%	98%	32%	93%	76%
MFCC	87%	96%	95%	80%	96%	42%	98%	72%
Chroma	33%	53%	48%	52%	54%	37%	58%	55%
Contrast	82%	84%	74%	73%	87%	25%	84%	82%
Tonnetz	43%	30%	31%	35%	46%	43%	49%	40%
MEL, MFCC	88%	95%	97%	80%	**99%**	36%	99%	75%
MEL, MFCC, Chroma	88%	95%	97%	79%	98%	36%	98%	76%
MEL, MFCC, Contrast	88%	98%	97%	88%	96%	26%	98%	79%
All	88%	98%	97%	75%	99%	26%	99%	76%

Afterward, I used the combined audio data for all users and applied different combinations of audio features using the TPOT classifier. The results are illustrated in Figure 15 to Figure 23, where the confusion matrices for the combined smart home dataset considering different features using the TPOT classifier is shown. For each algorithm, the bold diagonal blocks show the number of correctly classified labels whereas off-diagonal numbers are misclassified labels.

Figure 15 shows the confusion matrix after training the model on combined dataset extracting MEL feature using TPOT. The model used 5000 samples split into training and test data following 80%-20% train-test ratio. The training dataset consists of 4000 samples and test dataset consists of 1000 samples. The confusion matrix shows that among 1000 test samples, 150

samples were classified correctly as angry data, 180 samples were classified correctly as fearful data, 180 samples were classified correctly as happy data, 170 samples were classified correctly as normal data, and 180 samples were classified correctly as sad data. The rest of the test samples were misclassified and results in 85% classification accuracy.

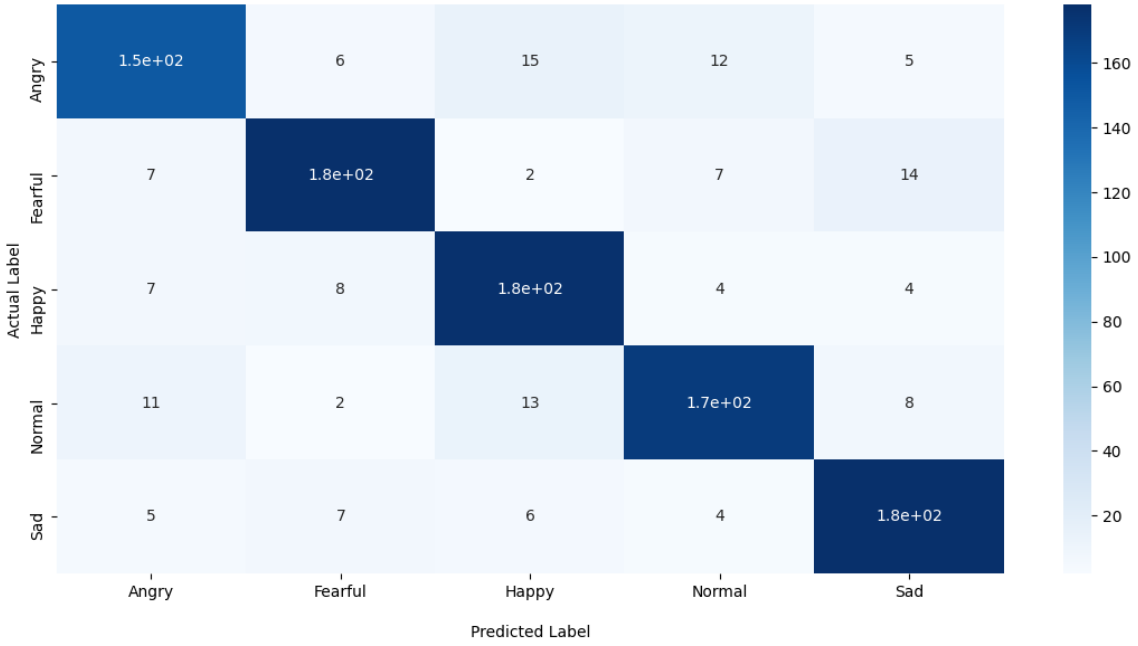


Figure 15. Confusion matrix for the combined smart home dataset considering MEL features using TPOT classifier

Figure 16 shows the confusion matrix after training the model on combined dataset extracting MFCC feature using TPOT. The model used 5000 samples split into training and test data following 80%-20% train-test ratio. The training dataset consists of 4000 samples and test dataset consists of 1000 samples. The confusion matrix shows that among 1000 test samples, 160 samples were classified correctly as angry data, 160 samples were classified correctly as fearful data, 160 samples were classified correctly as happy data, 170 samples were classified correctly

as normal data, and 170 samples were classified correctly as sad data. The rest of the test samples were misclassified and results in 81% classification accuracy.

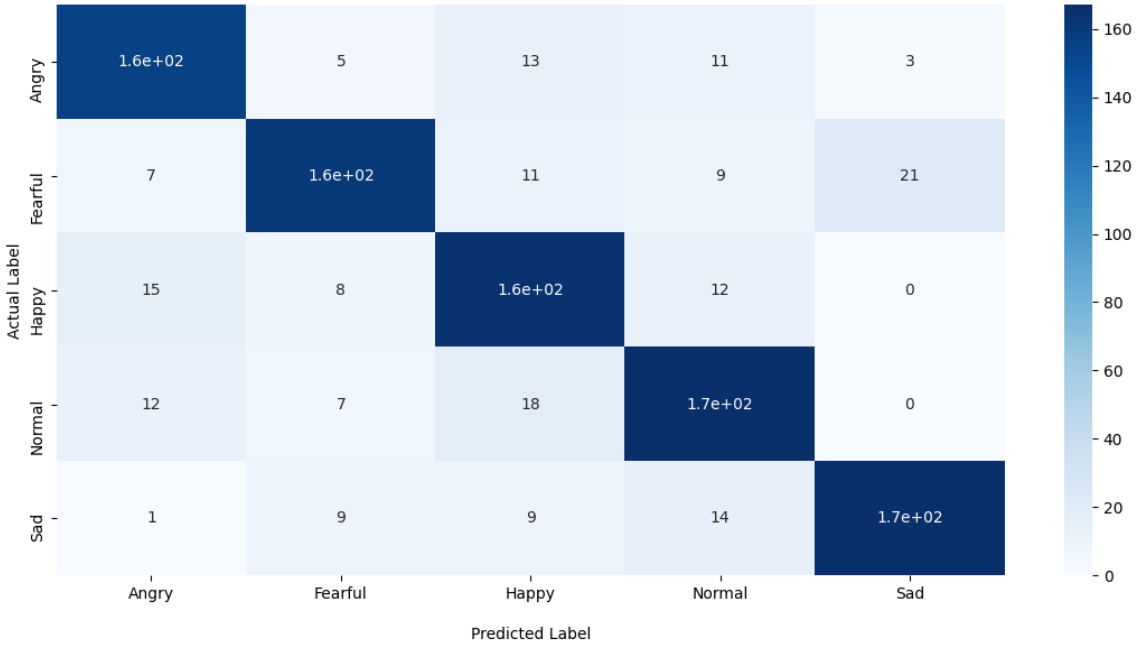


Figure 16. Confusion matrix for the combined smart home dataset considering MFCC features using TPOT classifier

Figure 17 shows the confusion matrix after training the model on combined dataset extracting Chroma feature using TPOT. The model used 5000 samples split into training and test data following 80%-20% train-test ratio. The training dataset consists of 4000 samples and test dataset consists of 1000 samples. The confusion matrix shows that among 1000 test samples, 84 samples were classified correctly as angry data, 92 samples were classified correctly as fearful data, 89 samples were classified correctly as happy data, 120 samples were classified correctly as normal data, and 94 samples were classified correctly as sad data. The rest of the test samples were misclassified and results in 48% classification accuracy.

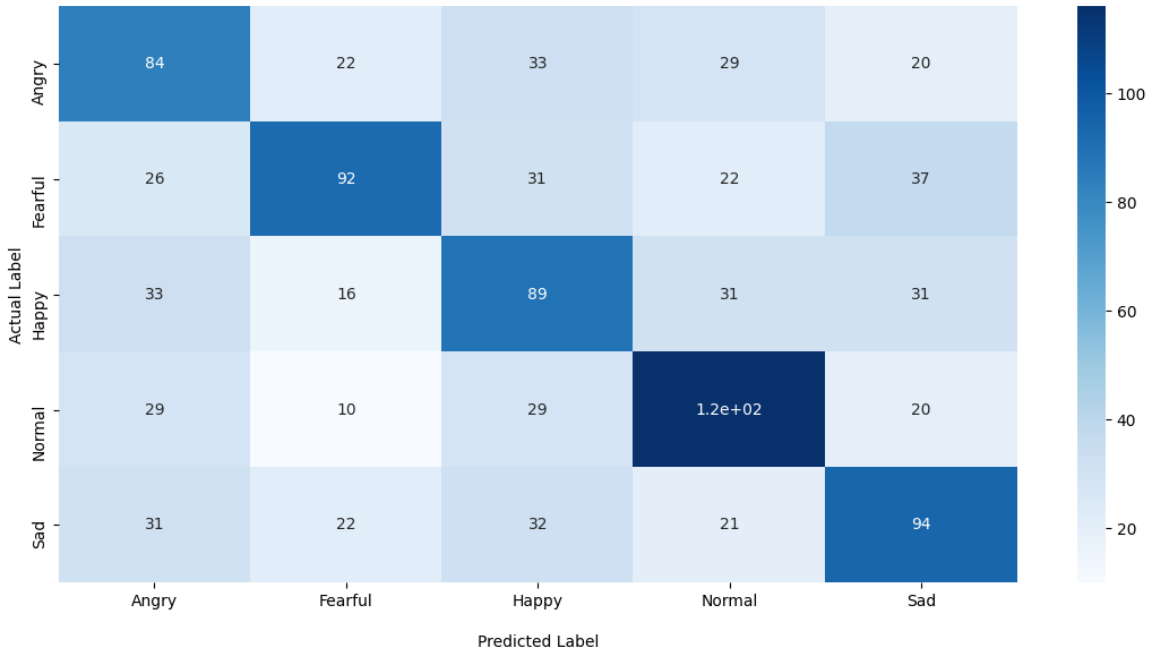


Figure 17. Confusion matrix for the combined smart home dataset considering Chroma features using TPOT classifier

Figure 18 shows the confusion matrix after training the model on combined dataset extracting Contrast feature using TPOT. The model used 5000 samples split into training and test data following 80%-20% train-test ratio. The training dataset consists of 4000 samples and test dataset consists of 1000 samples. The confusion matrix shows that among 1000 test samples, 110 samples were classified correctly as angry data, 120 samples were classified correctly as fearful data, 100 samples were classified correctly as happy data, 150 samples were classified correctly as normal data, and 130 samples were classified correctly as sad data. The rest of the test samples were misclassified and results in 31% classification accuracy.

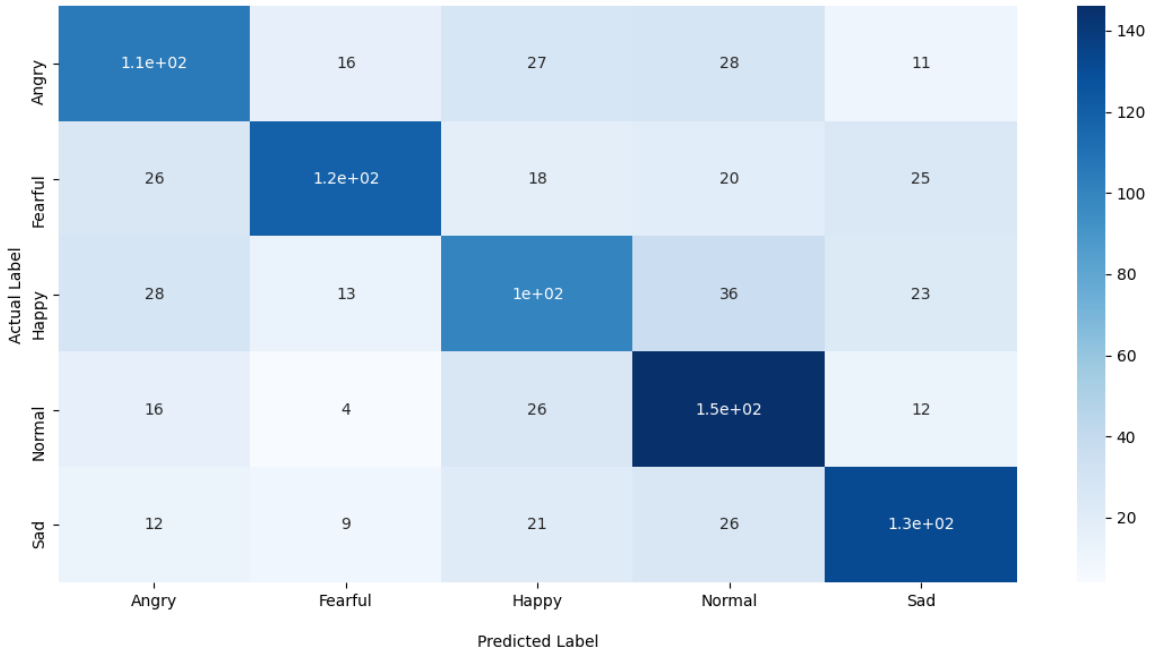


Figure 18. Confusion matrix for the combined smart home dataset considering Contrast features using TPOT classifier

Figure 19 shows the confusion matrix after training the model on combined dataset extracting Tonnetz feature using TPOT. The model used 5000 samples split into training and test data following 80%-20% train-test ratio. The training dataset consists of 4000 samples and test dataset consists of 1000 samples. The confusion matrix shows that among 1000 test samples, 58 samples were classified correctly as angry data, 62 samples were classified correctly as fearful data, 55 samples were classified correctly as happy data, 63 samples were classified correctly as normal data, and 73 samples were classified correctly as sad data. The rest of the test samples were misclassified and results in 31% classification accuracy.

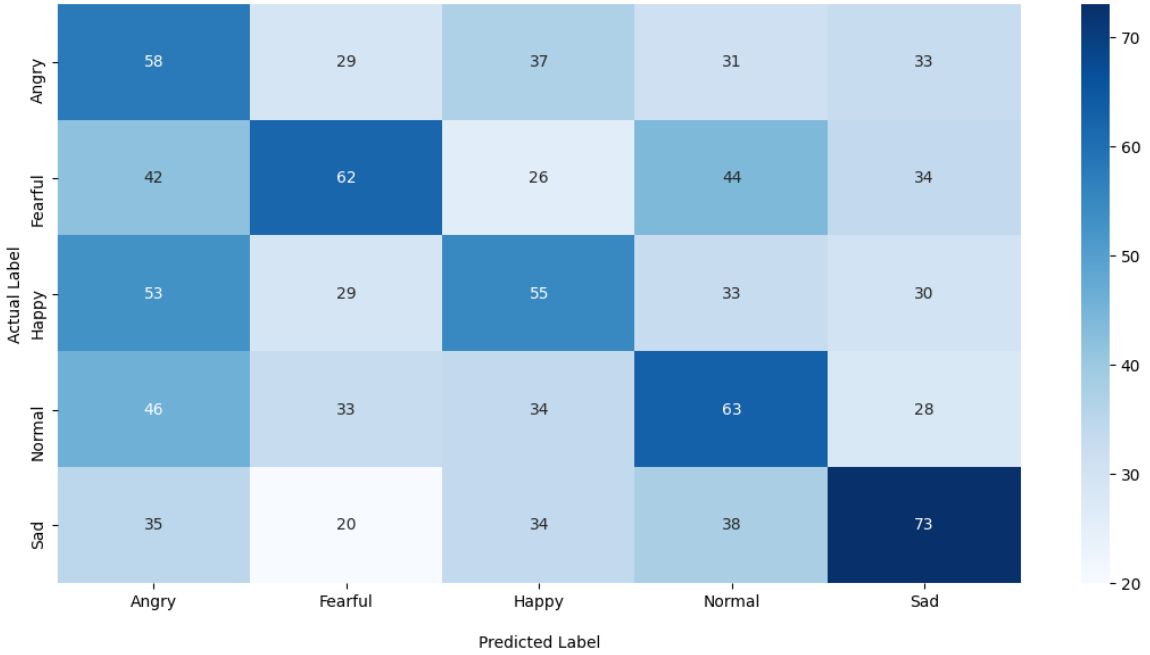


Figure 19. Confusion matrix for the combined smart home dataset considering Tonnetz features using TPOT classifier

Figure 20 shows the confusion matrix after training the model on combined dataset extracting MEL, MFCC feature using TPOT. The model used 5000 samples split into training and test data following 80%-20% train-test ratio. The training dataset consists of 4000 samples and test dataset consists of 1000 samples. The confusion matrix shows that among 1000 test samples, 160 samples were classified correctly as angry data, 190 samples were classified correctly as fearful data, 180 samples were classified correctly as happy data, 180 samples were classified correctly as normal data, and 180 samples were classified correctly as sad data. The rest of the test samples were misclassified and results in 90% classification accuracy, which is more than MEL or MFCC only.

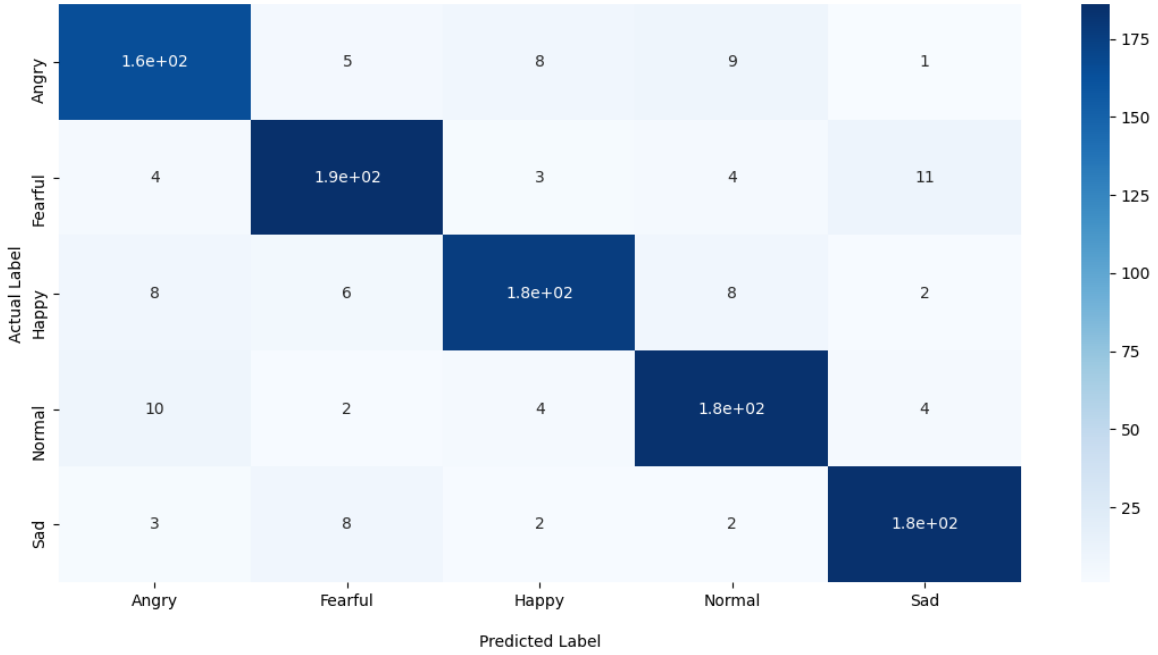


Figure 20. Confusion matrix for the combined smart home dataset considering MEL, MFCC features using TPOT classifier

Figure 21 shows the confusion matrix after training the model on combined dataset extracting MEL, MFCC, Chroma feature using TPOT. The model used 5000 samples split into training and test data following 80%-20% train-test ratio. The training dataset consists of 4000 samples and test dataset consists of 1000 samples. The confusion matrix shows that among 1000 test samples, 170 samples were classified correctly as angry data, 190 samples were classified correctly as fearful data, 180 samples were classified correctly as happy data, 180 samples were classified correctly as normal data, and 190 samples were classified correctly as sad data. The rest of the test samples were misclassified and results in 88% classification accuracy.

Figure 21. Confusion matrix for the combined smart home dataset considering MEL, MFCC, Chroma features using TPOT classifier

Figure 22 shows the confusion matrix after training the model on combined dataset extracting MEL, MFCC, Contrast feature using TPOT. The model used 5000 samples split into training and test data following 80%-20% train-test ratio. The training dataset consists of 4000 samples and test dataset consists of 1000 samples. The confusion matrix shows that among 1000 test samples, 170 samples were classified correctly as angry data, 190 samples were classified correctly as fearful data, 180 samples were classified correctly as happy data, 180 samples were classified correctly as sad data, and 190 samples were classified correctly as normal data. The rest of the test samples were misclassified and results in 88% classification accuracy.

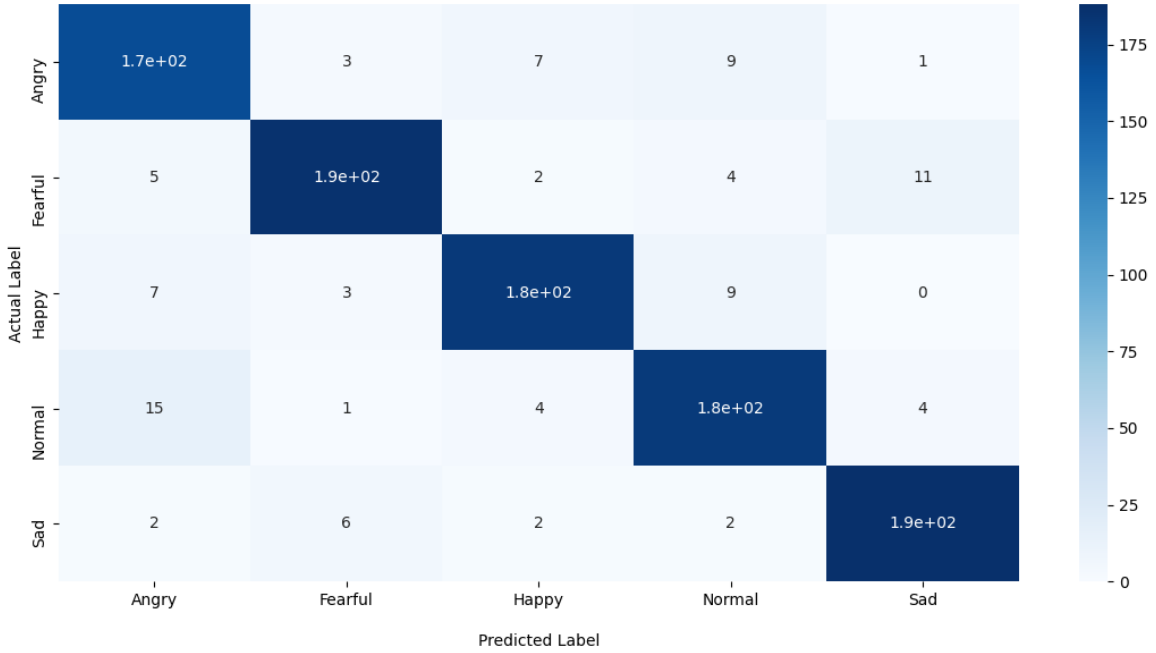


Figure 22. Confusion matrix for the combined smart home dataset considering MEL, MFCC, Contrast features using TPOT classifier

Figure 23 shows the confusion matrix after training the model on combined dataset extracting MEL, MFCC, Contrast, Chroma, Tonnetz feature using TPOT. The model used 5000 samples split into training and test data following 80%-20% train-test ratio. The training dataset consists of 4000 samples and test dataset consists of 1000 samples. The confusion matrix shows that among 1000 test samples, 160 samples were classified correctly as angry data, 180 samples were classified correctly as fearful data, 180 samples were classified correctly as happy data, 180 samples were classified correctly as normal data, and 180 samples were classified correctly as sad data. The rest of the test samples were misclassified and results in 89% classification accuracy.

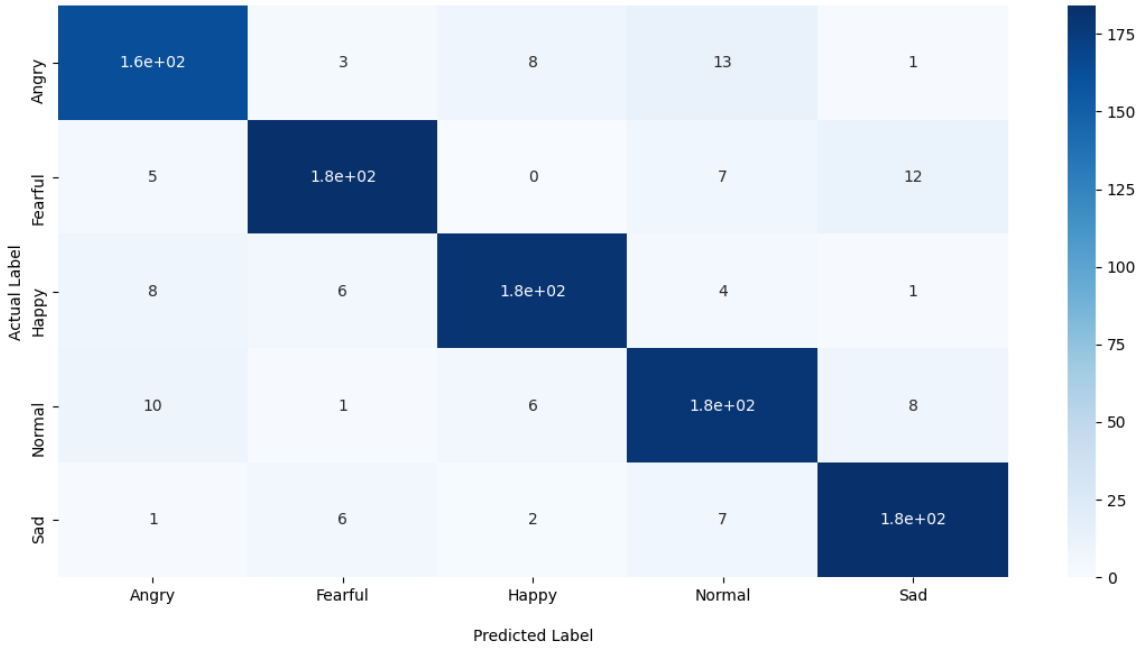


Figure 23. Confusion matrix for the combined smart home dataset considering MEL, MFCC, Contrast, Chroma, Tonnetz features using TPOT classifier

The results highlighted in Figure 15 to Figure 23 is summarized in Table 10. From the table results, we see that the combination of MEL, and MFCC shows the best result i.e., 90%.

Table 10. Experimental results of emotion detection for TPOT classifier using the combined smart home dataset

Features	**F-1 score**	**Precision**	**Recall**	**Support**
MEL	85%	85%	85%	1000
MFCC	81%	82%	82%	1000
Chroma	48%	48%	47%	1000
Contrast	60%	61%	60%	1000
Tonnetz	31%	31%	31%	1000
MEL, MFCC	90%	**90%**	90%	1000
MEL, MFCC, Chroma	88%	88%	87%	1000
MEL, MFCC, Contrast	88%	88%	87%	1000
All	89%	89%	89%	1000

Finally, the TPOT classification model is trained and tested with the custom smart home dataset for each participant by splitting each user's dataset into 80% as training data and 20% as testing data. Figures 24 to Figure 45 show the confusion matrix for each participant's smart home dataset considering different features using the TPOT classifier.

Figure 24 shows the confusion matrix after training the model on S001 users' dataset extracting MEL, MFCC features using TPOT. The actor is a female of 15-20 age and a native English speaker. The model used 500 samples split into training and test data following 80%-20% train-test ratio. The training dataset consists of 400 samples and test dataset consists of 100 samples. The confusion matrix shows that among 100 test samples, 19 samples were classified correctly as angry data, 19 samples were classified correctly as fearful data, 14 samples were classified correctly as happy data, 17 samples were classified correctly as normal data, and 15 samples were classified correctly as sad data. The rest of the test samples were misclassified and results in 84% classification accuracy.

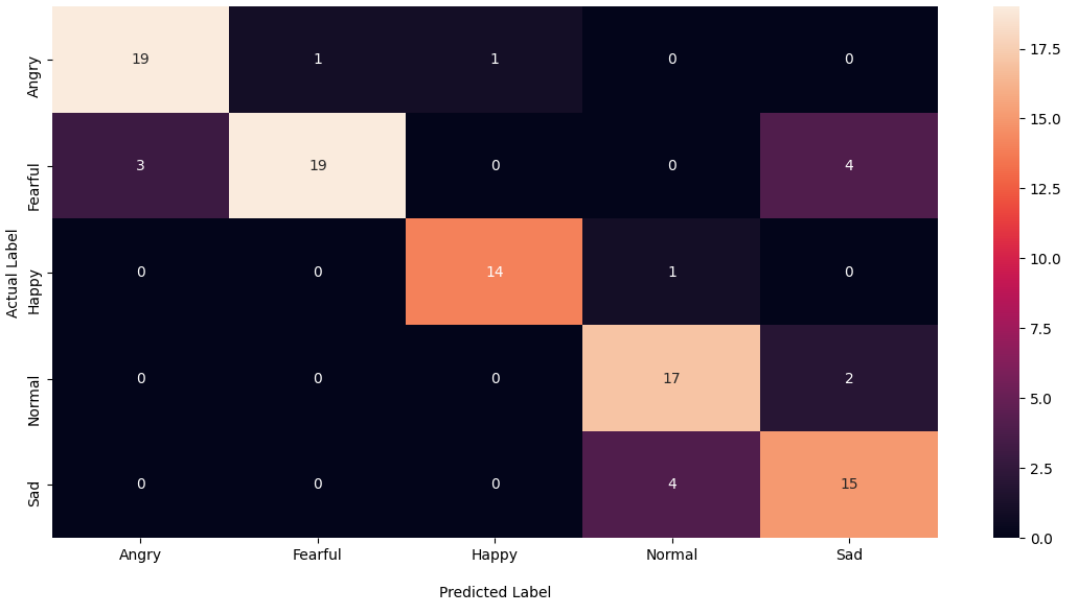


Figure 24. Confusion matrix for the S001 considering MEL, MFCC features using TPOT classifier

Figure 25 shows the confusion matrix after training the model on S001 users' dataset extracting MEL, MFCC, MWDPC features using TPOT. The model used 500 samples split into training and test data following 80%-20% train-test ratio. The training dataset consists of 400 samples and test dataset consists of 100 samples. The confusion matrix shows that among 100 test samples, 19 samples were classified correctly as angry data, 21 samples were classified correctly as fearful data, 14 samples were classified correctly as happy data, 17 samples were classified correctly as normal data, and 15 samples were classified correctly as sad data. The rest of the test samples were misclassified and results in 87% classification accuracy, which is 3% more than extracting MEL, MFCC features. I performed a chi-squared to determine if the difference in the classification accuracy is statistically significant for user S001 and found the resultant p value as 0.046 (i.e. < 0.05), yielding a significant improvement in the proposed methodology.

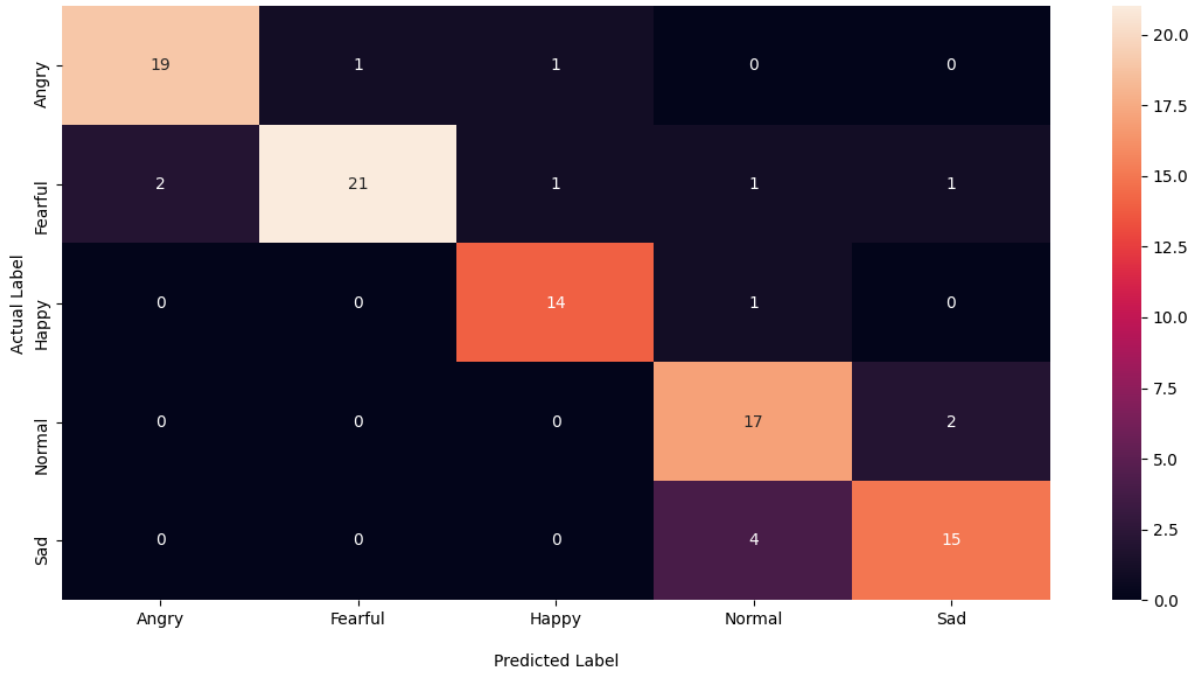


Figure 25. Confusion matrix for the S001 considering MEL, MFCC, MWDPC features using TPOT classifier

Figure 26 shows the confusion matrix after training the model on S002 users' dataset extracting MEL, MFCC features using TPOT. The actor is a male of 21-25 age and a native English speaker. The model used 500 samples split into training and test data following 80%-20% train-test ratio. The training dataset consists of 400 samples and test dataset consists of 100 samples. The confusion matrix shows that among 100 test samples, 18 samples were classified correctly as angry data, 22 samples were classified correctly as fearful data, 14 samples were classified correctly as happy data, 18 samples were classified correctly as normal data, and 18 samples were classified correctly as sad data. The rest of the test samples were misclassified and results in 90% classification accuracy.

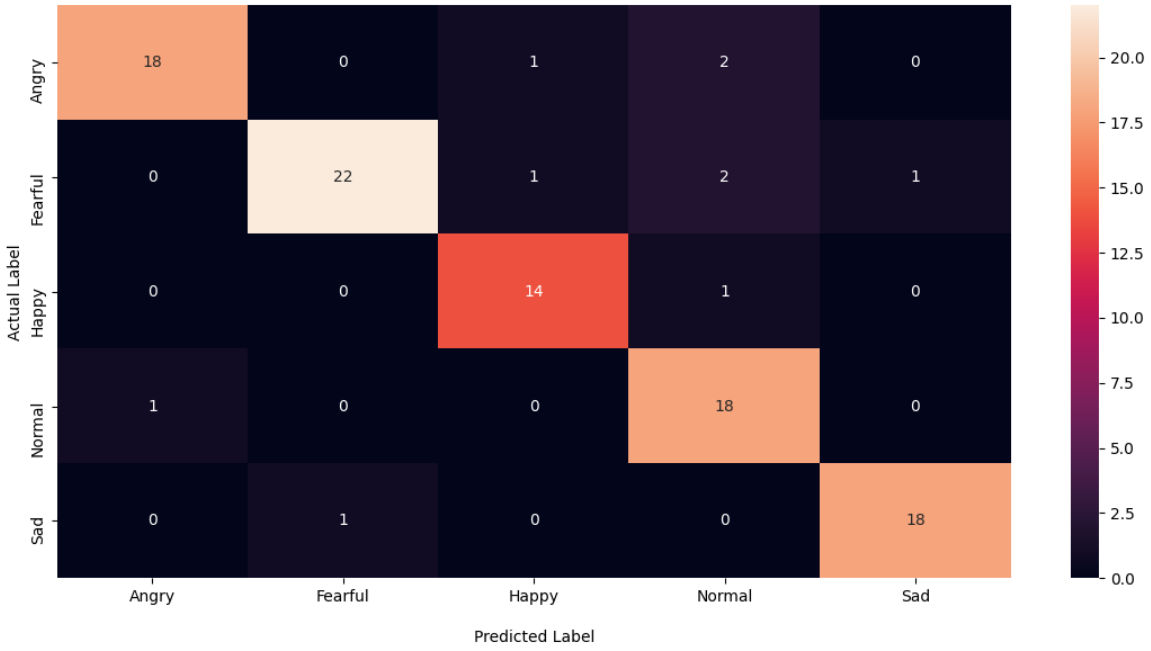


Figure 26. Confusion matrix for the S002 considering MEL, MFCC features using TPOT classifier

Figure 27 shows the confusion matrix after training the model on S002 users' dataset extracting MEL, MFCC, PCWC features using TPOT. The model used 500 samples split into training and test data following 80%-20% train-test ratio. The training dataset consists of 400 samples and test dataset consists of 100 samples. The confusion matrix shows that among 100 test samples, 19 samples were classified correctly as angry data, 23 samples were classified correctly as fearful data, 15 samples were classified correctly as happy data, 19 samples were classified correctly as normal data, and 19 samples were classified correctly as sad data. The rest of the test samples were misclassified and results in 92% classification accuracy, which is 2% more than extracting MEL, MFCC features.

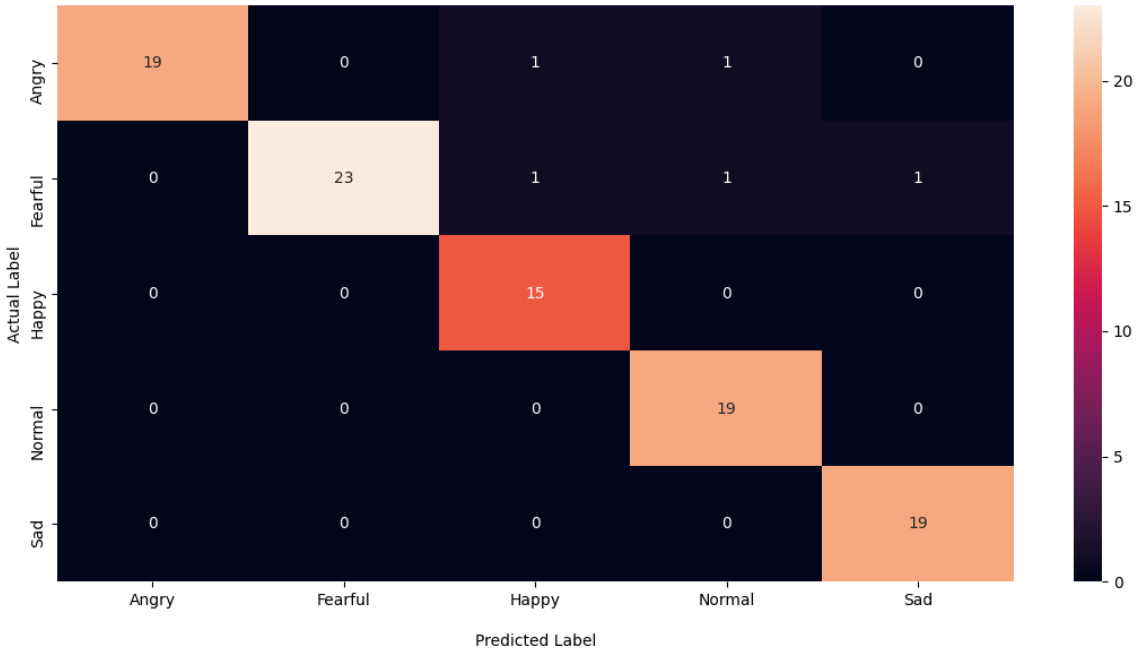


Figure 27. Confusion matrix for the S002 considering MEL, MFCC, PCWC features using TPOT classifier

Figure 28 shows the confusion matrix after training the model on S002 users' dataset extracting MEL, MFCC, MWDPC, PCWC features using TPOT. The model used 500 samples split into training and test data following 80%-20% train-test ratio. The training dataset consists of 400 samples and test dataset consists of 100 samples. The confusion matrix shows that among 100 test samples, 18 samples were classified correctly as angry data, 23 samples were classified correctly as fearful data, 14 samples were classified correctly as happy data, 19 samples were classified correctly as normal data, and 18 samples were classified correctly as sad data. The rest of the test samples were misclassified and results in 92% classification accuracy, which is 2% more than extracting MEL, MFCC features. However, this improvement in accuracy is not statistically significant as it resulted in a *p* value of 0.09926 in a chi-squared test.

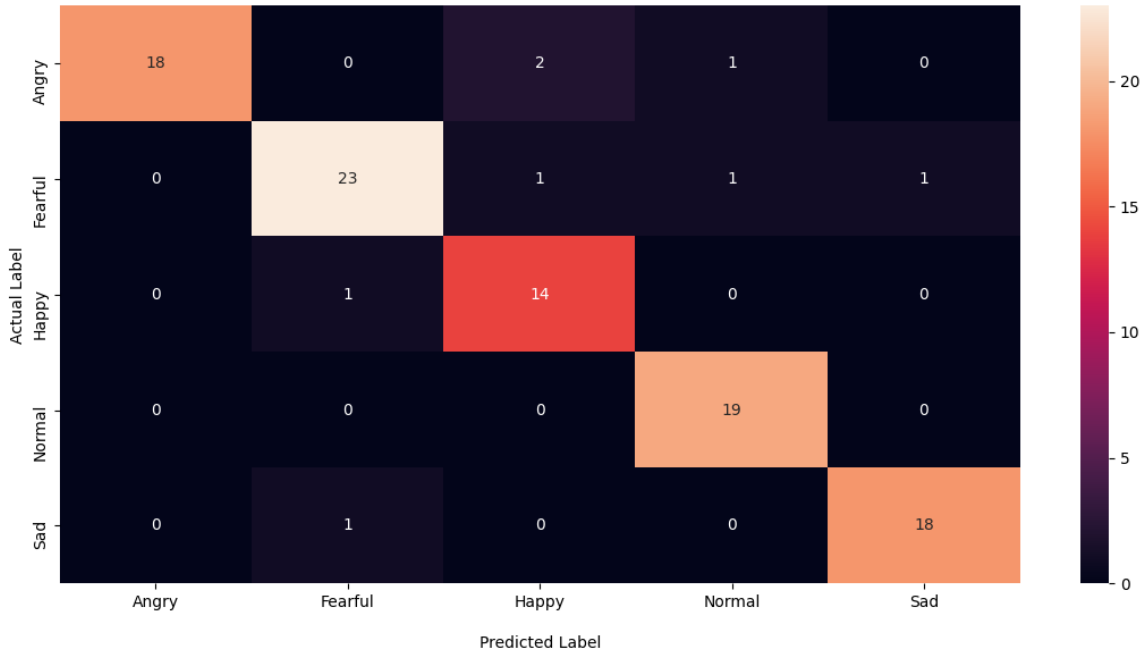


Figure 28. Confusion matrix for the S002 considering MEL, MFCC, MWDPC, PCWC features using TPOT classifier

Figure 29 shows the confusion matrix after training the model on S003 users' dataset extracting MEL, MFCC features using TPOT. The actor is a male of 21-25 age and a non-native English speaker. The model used 500 samples split into training and test data following 80%-20% train-test ratio. The training dataset consists of 400 samples and test dataset consists of 100 samples. The confusion matrix shows that among 100 test samples, 21 samples were classified correctly as angry data, 24 samples were classified correctly as fearful data, 15 samples were classified correctly as happy data, 19 samples were classified correctly as normal data, and 18 samples were classified correctly as sad data. The rest of the test samples were misclassified and results in 69% classification accuracy.

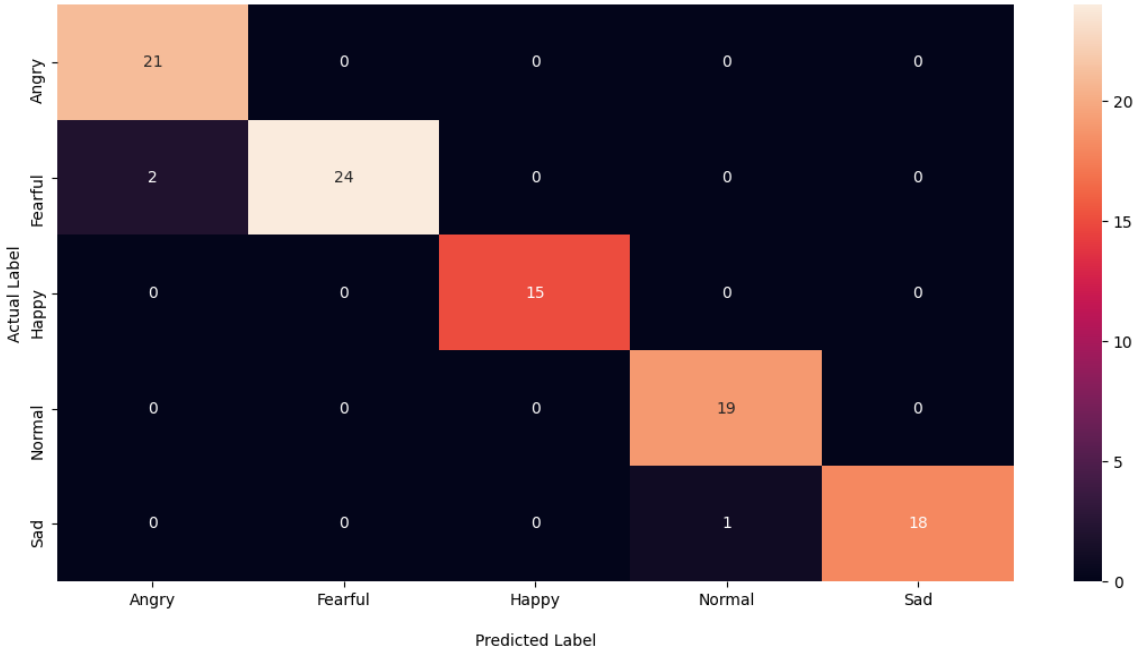


Figure 29. Confusion matrix for the S003 considering MEL, MFCC features using TPOT classifier

Figure 30 shows the confusion matrix after training the model on S003 users' dataset extracting MEL, MFCC, MCD, CGL features using TPOT. The model used 500 samples split into training and test data following 80%-20% train-test ratio. The training dataset consists of 400 samples and test dataset consists of 100 samples. The confusion matrix shows that among 100 test samples, 21 samples were classified correctly as angry data, 26 samples were classified correctly as fearful data, 15 samples were classified correctly as happy data, 19 samples were classified correctly as normal data, and 19 samples were classified correctly as normal data. The rest of the test samples were misclassified and results in 72% classification accuracy, which is 3% more than extracting MEL, MFCC features. This is not a statistically significant improvement given that the correct and incorrect classification resulted in a p value of 0.13516 in chi-squared test.

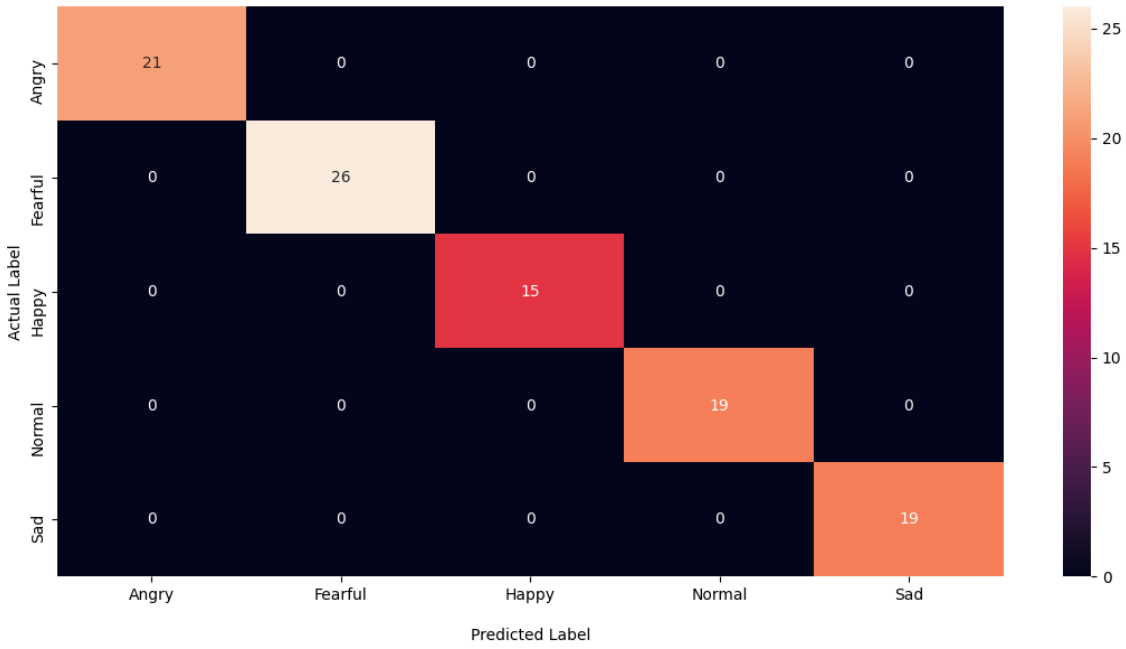


Figure 30. Confusion matrix for the S003 considering MEL, MFCC, MCD, CGL features using TPOT classifier

Figure 31 shows the confusion matrix after training the model on S004 users' dataset extracting MEL, MFCC features using TPOT. The actor is a non-binary of age 21-25 and a native English speaker. The model used 500 samples split into training and test data following 80%-20% train-test ratio. The training dataset consists of 400 samples and test dataset consists of 100 samples. The confusion matrix shows that among 100 test samples, 12 samples were classified correctly as angry data, 14 samples were classified correctly as fearful data, 14 samples were classified correctly as happy data, 14 samples were classified correctly as normal data, and 15 samples were classified correctly as sad data. The rest of the test samples were misclassified and results in 97% classification accuracy.

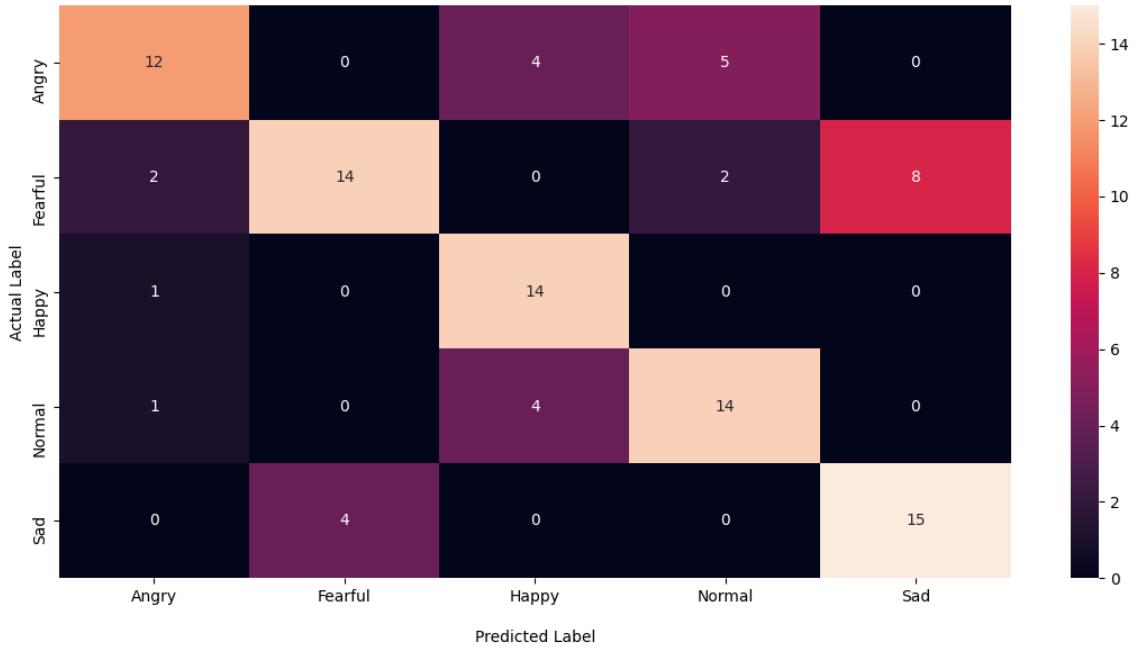


Figure 31. Confusion matrix for the S004 considering MEL, MFCC features using TPOT classifier

Figure 32 shows the confusion matrix after training the model on S004 users' dataset extracting MEL, MFCC, MWDPC features using TPOT. The model used 500 samples split into training and test data following 80%-20% train-test ratio. The training dataset consists of 400 samples and test dataset consists of 100 samples. The confusion matrix shows that among 100 test samples, 12 samples were classified correctly as angry data, 17 samples were classified correctly as fearful data, 14 samples were classified correctly as happy data, 13 samples were classified correctly as normal data, and 16 samples were classified correctly as sad data. The rest of the test samples were misclassified and results in 99% classification accuracy, which is 2% more than extracting MEL, MFCC features. This is statistically significant improvement based on the p value of 0.044 from a chi-squared test.

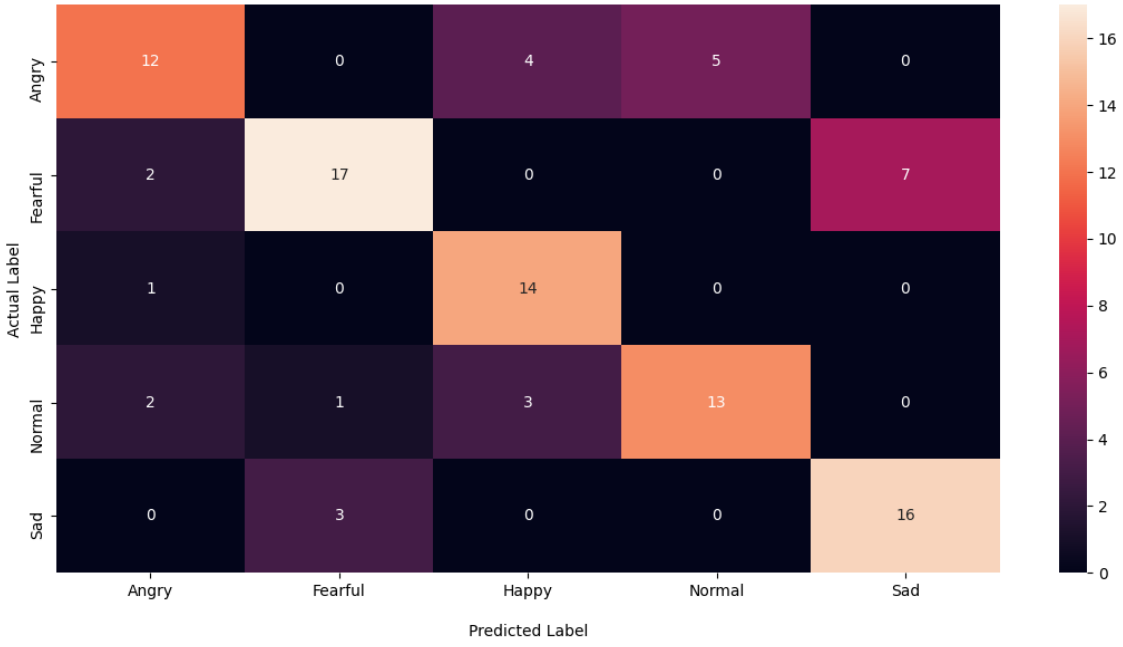


Figure 32. Confusion matrix for the S004 considering MEL, MFCC, MWDPC features using TPOT classifier

Figure 33 shows the confusion matrix after training the model on S005 users' dataset extracting MEL, MFCC features using TPOT. The actor is a female of 15-20 age and a native English speaker. The model used 500 samples split into training and test data following 80%-20% train-test ratio. The training dataset consists of 400 samples and test dataset consists of 100 samples. The confusion matrix shows that among 100 test samples, 18 samples were classified correctly as angry data, 22 samples were classified correctly as fearful data, 13 samples were classified correctly as happy data, 18 samples were classified correctly as normal data, and 19 samples were classified correctly as sad data. The rest of the test samples were misclassified and results in 90% classification accuracy.

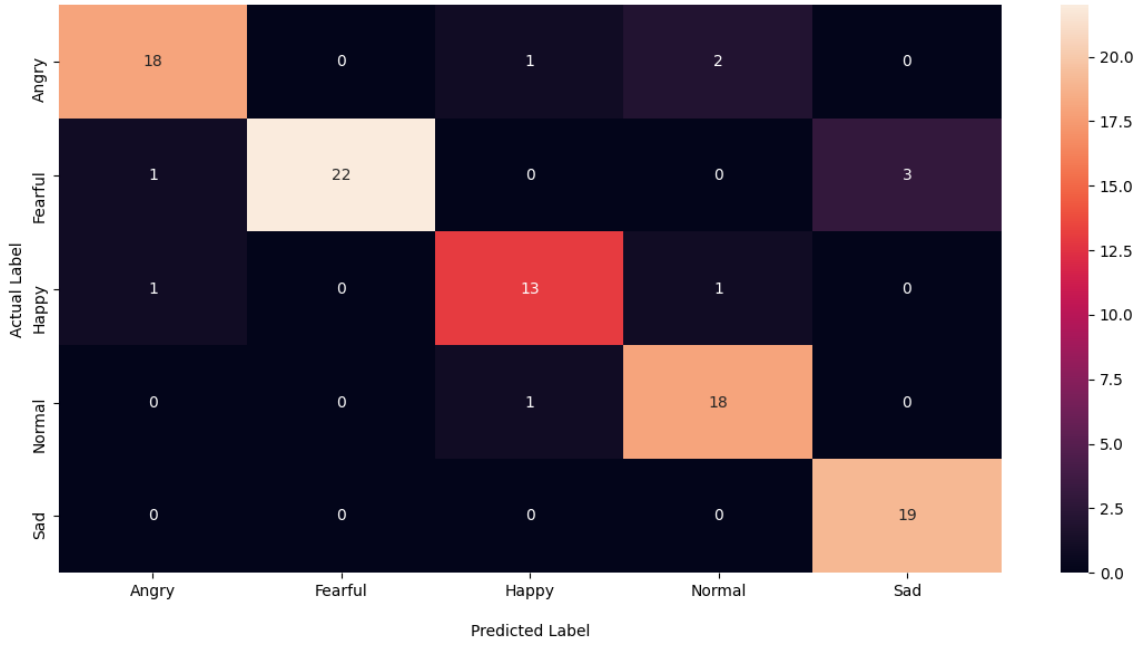


Figure 33. Confusion matrix for the S005 considering MEL, MFCC features using TPOT classifier

Figure 34 shows the confusion matrix after training the model on S005 users' dataset extracting MEL, MFCC, CGL features using TPOT. The model used 500 samples split into training and test data following 80%-20% train-test ratio. The training dataset consists of 400 samples and test dataset consists of 100 samples. The confusion matrix shows that among 100 test samples, 19 samples were classified correctly as angry data, 22 samples were classified correctly as fearful data, 13 samples were classified correctly as happy data, 18 samples were classified correctly as normal data, and 19 samples were classified correctly as sad data. The rest of the test samples were misclassified and results in 94% classification accuracy, which is 4% more than extracting MEL, MFCC features. The statistical significance for this user is p = 0.01722, which is statistically significant improvement.

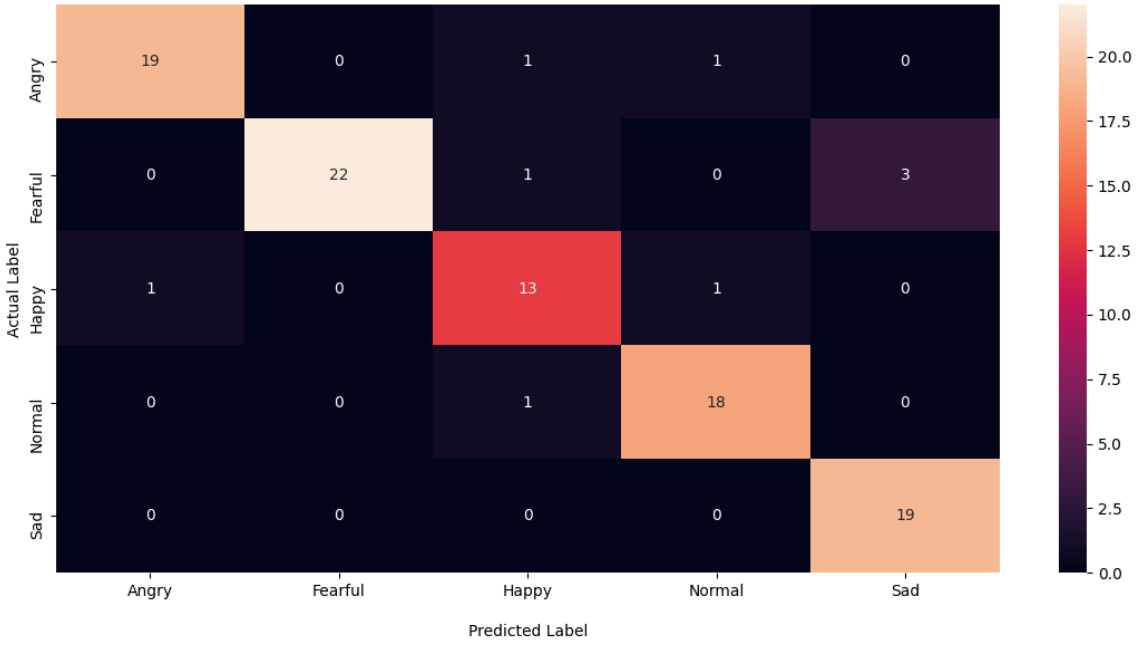


Figure 34. Confusion matrix for the S005 considering MEL, MFCC, CGL features using TPOT classifier

Figure 35 shows the confusion matrix after training the model on S006 users' dataset extracting MEL, MFCC features using TPOT. The actor is a male of 31-35 age and a non-native English speaker. The model used 500 samples split into training and test data following 80%-20% train-test ratio. The training dataset consists of 400 samples and test dataset consists of 100 samples. The confusion matrix shows that among 100 test samples, 19 samples were classified correctly as angry data, 26 samples were classified correctly as fearful data, 14 samples were classified correctly as happy data, 14 samples were classified correctly as normal data, and 19 samples were classified correctly as sad data. The rest of the test samples were misclassified and results in 92% classification accuracy.

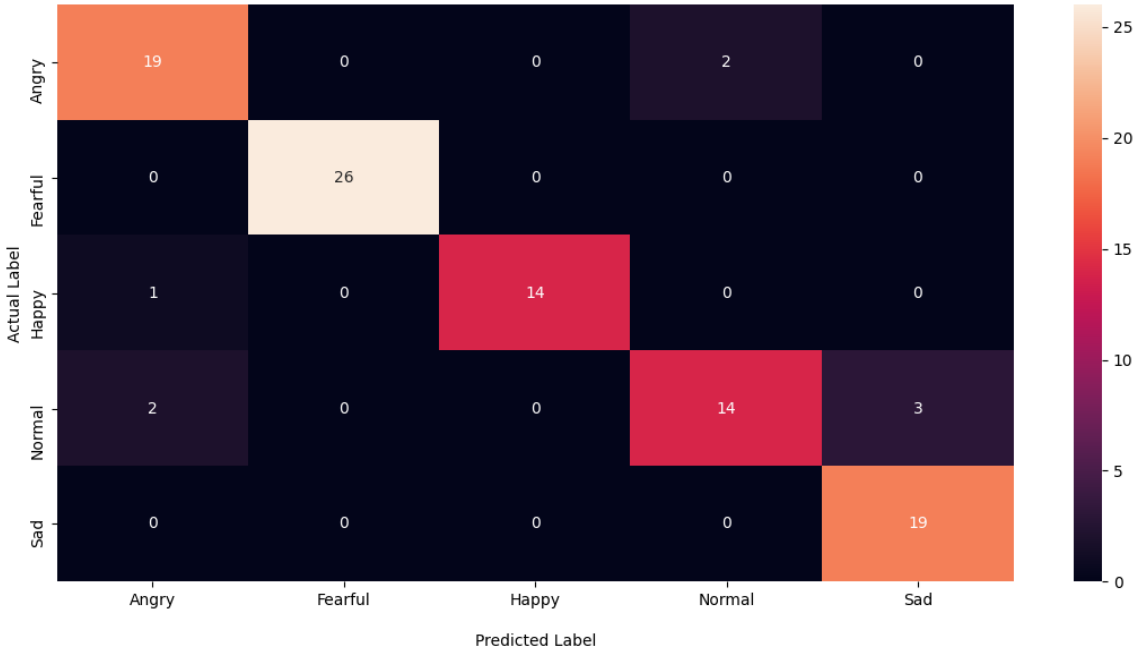


Figure 35. Confusion matrix for the S006 considering MEL, MFCC features using TPOT classifier

Figure 36 shows the confusion matrix after training the model on S006 users' dataset extracting MEL, MFCC, MCD features using TPOT. The model used 500 samples split into training and test data following 80%-20% train-test ratio. The training dataset consists of 400 samples and test dataset consists of 100 samples. The confusion matrix shows that among 100 test samples, 19 samples were classified correctly as angry data, 26 samples were classified correctly as fearful data, 14 samples were classified correctly as happy data, 18 samples were classified correctly as normal data, and 18 samples were classified correctly as sad data. The rest of the test samples were misclassified and results in 95% classification accuracy, which is 3% more than extracting MEL, MFCC features. Under a chi-squared test, I found the probability value to be $p = 0.002$, yielding a statistically significant improvement in the performance evaluation.

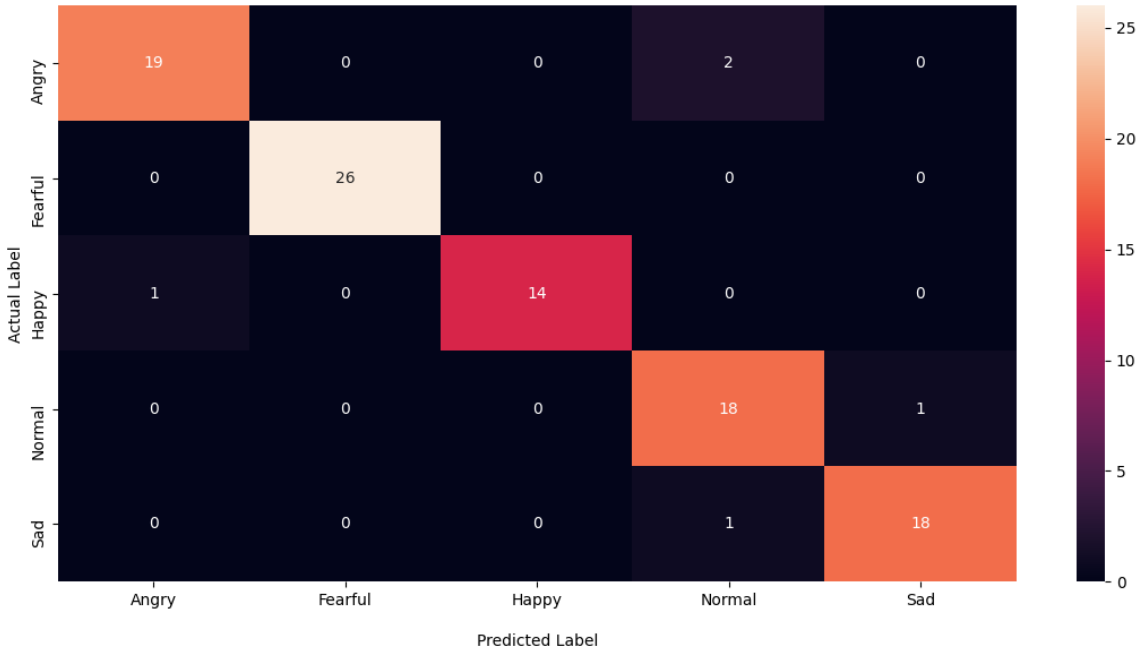


Figure 36. Confusion matrix for the S006 considering MEL, MFCC, MCD features using TPOT classifier

Figure 37 shows the confusion matrix after training the model on S007 users' dataset extracting MEL, MFCC features using TPOT. The actor is a male of 36-40 age and a non-native English speaker. The model used 500 samples split into training and test data following 80%-20% train-test ratio. The training dataset consists of 400 samples and test dataset consists of 100 samples. The confusion matrix shows that among 100 test samples, 21 samples were classified correctly as angry data, 25 samples were classified correctly as fearful data, 15 samples were classified correctly as happy data, 19 samples were classified correctly as normal data, and 19 samples were classified correctly as sad data. The rest of the test samples were misclassified and results in 99% classification accuracy.

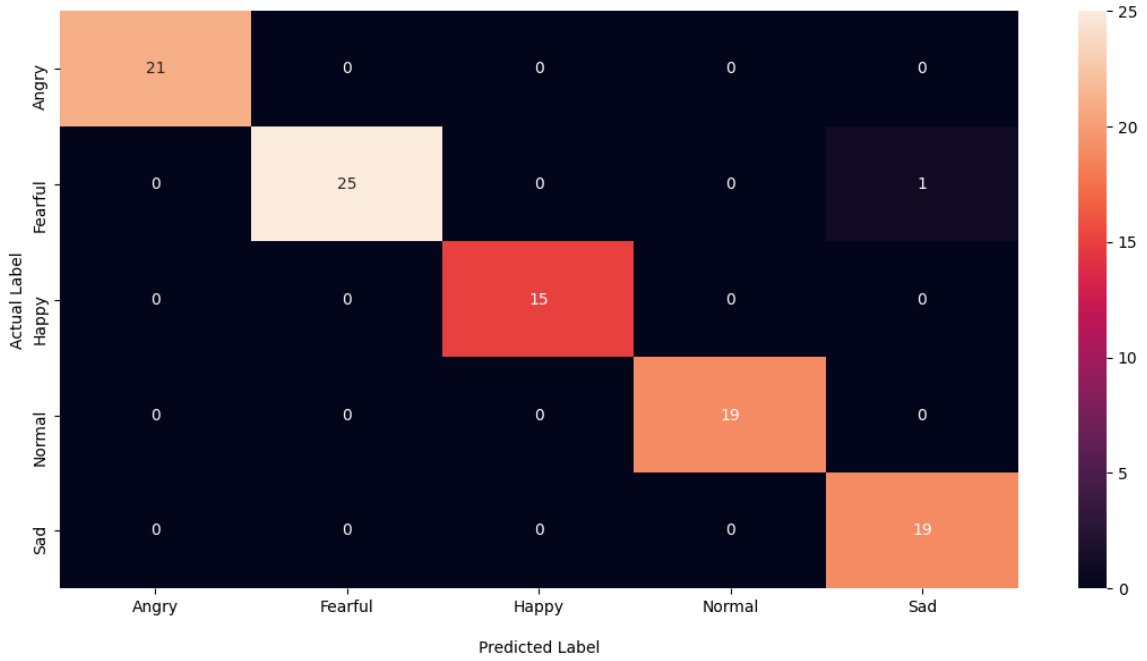


Figure 37. Confusion matrix for the S007 considering MEL, MFCC features using TPOT classifier

Figure 38 shows the confusion matrix after training the model on S007 users' dataset extracting MEL, MFCC, MCD, CGL features using TPOT. The model used 500 samples split into training and test data following 80%-20% train-test ratio. The training dataset consists of 400 samples and test dataset consists of 100 samples. The confusion matrix shows that among 100 test samples, 21 samples were classified correctly as angry data, 26 samples were classified correctly as fearful data, 15 samples were classified correctly as happy data, 19 samples were classified correctly as normal data, and 19 samples were classified correctly as sad data. The rest of the test samples were misclassified and results in 100% classification accuracy, which is 1% more than extracting MEL, MFCC features. This is a statistically significant improvement as the proposed methodology yields maximum possible accuracy.

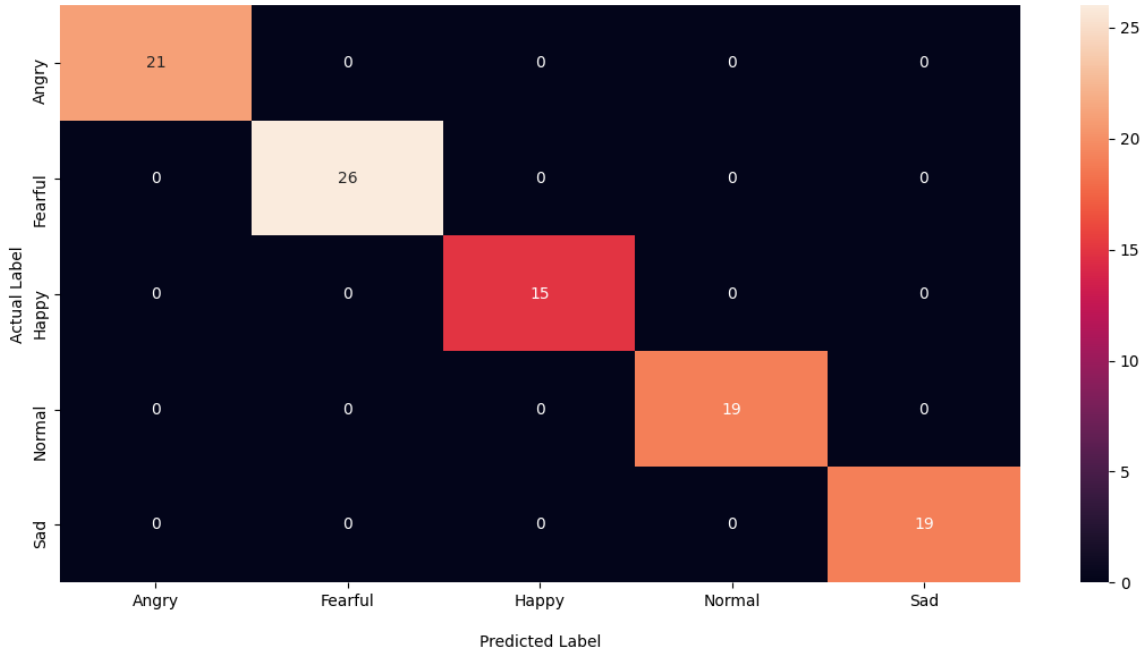


Figure 38. Confusion matrix for the S007 considering MEL, MFCC, MCD, CGL features using TPOT classifier

Figure 39 shows the confusion matrix after training the model on S008 users' dataset extracting MEL, MFCC features using TPOT. The actor is a female of 26-30 age and a non-native English speaker. The model used 500 samples split into training and test data following 80%-20% train-test ratio. The training dataset consists of 400 samples and test dataset consists of 100 samples. The confusion matrix shows that among 100 test samples, 18 samples were classified correctly as angry data, 22 samples were classified correctly as fearful data, 15 samples were classified correctly as happy data, 19 samples were classified correctly as normal data, and 15 samples were classified correctly as sad data. The rest of the test samples were misclassified and results in 89% classification accuracy.

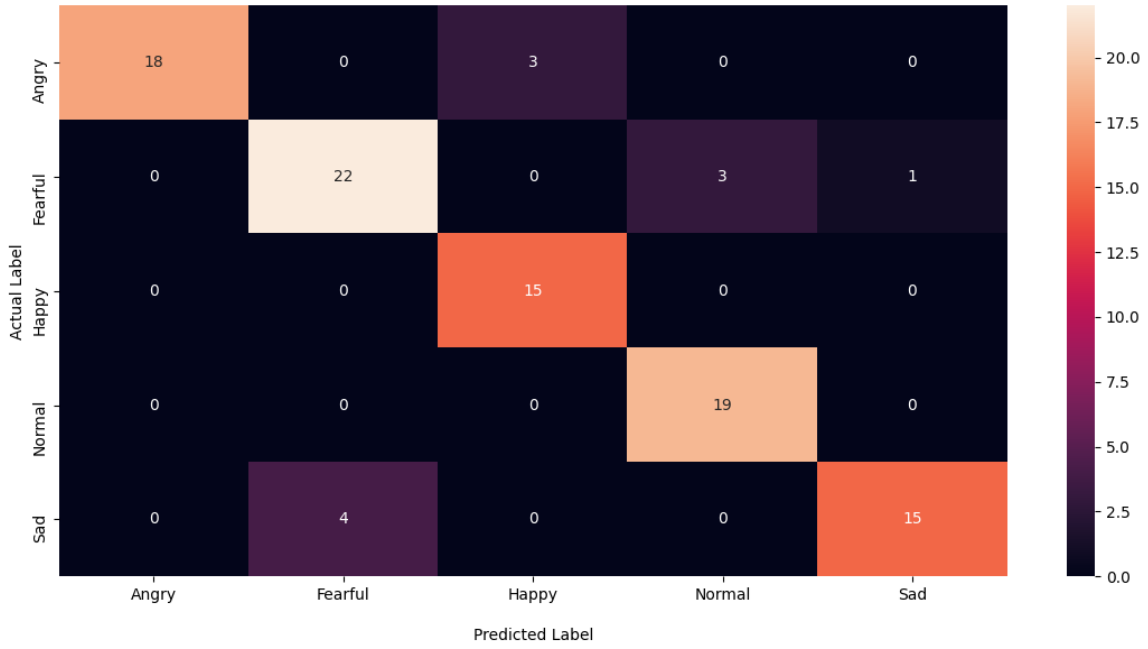


Figure 39. Confusion matrix for the S008 considering MEL, MFCC features using TPOT classifier

Figure 40 shows the confusion matrix after training the model on S008 users' dataset extracting MEL, MFCC, MWDPC features using TPOT. The model used 500 samples split into training and test data following 80%-20% train-test ratio. The training dataset consists of 400 samples and test dataset consists of 100 samples. The confusion matrix shows that among 100 test samples, 18 samples were classified correctly as angry data, 25 samples were classified correctly as fearful data, 15 samples were classified correctly as happy data, 19 samples were classified correctly as normal data, and 17 samples were classified correctly as sad data. The rest of the test samples were misclassified and results in 95% classification accuracy, which is 6% more than extracting MEL, MFCC features. Based on the same chi-squared test, I found the improvement yielding a *p* value of 0.0059, which is statistically significant.

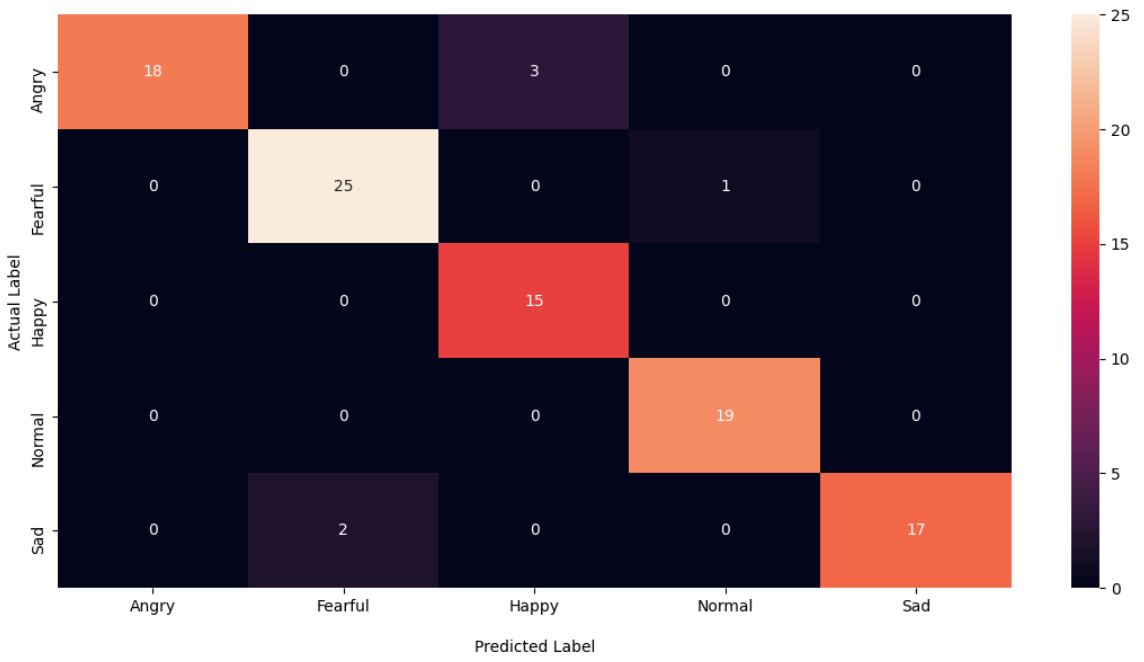


Figure 40. Confusion matrix for the S008 considering MEL, MFCC, MWDPC features using TPOT classifier

Figure 41 shows the confusion matrix after training the model on S009 users' dataset extracting MEL, MFCC features using TPOT. The actor is a male of 21-25 age and a native English speaker. The model used 500 samples where 400 training samples and 100 testing samples after splitting the dataset into 80%-20% train test ratio. The confusion matrix shows that among 100 test samples, 21 samples were classified correctly as angry data, 26 samples were classified correctly as fearful data, 15 samples were classified correctly as happy data, 19 samples were classified correctly as normal data, and 19 samples were classified correctly as sad data. The rest of the test samples were misclassified and results in 98% classification accuracy.

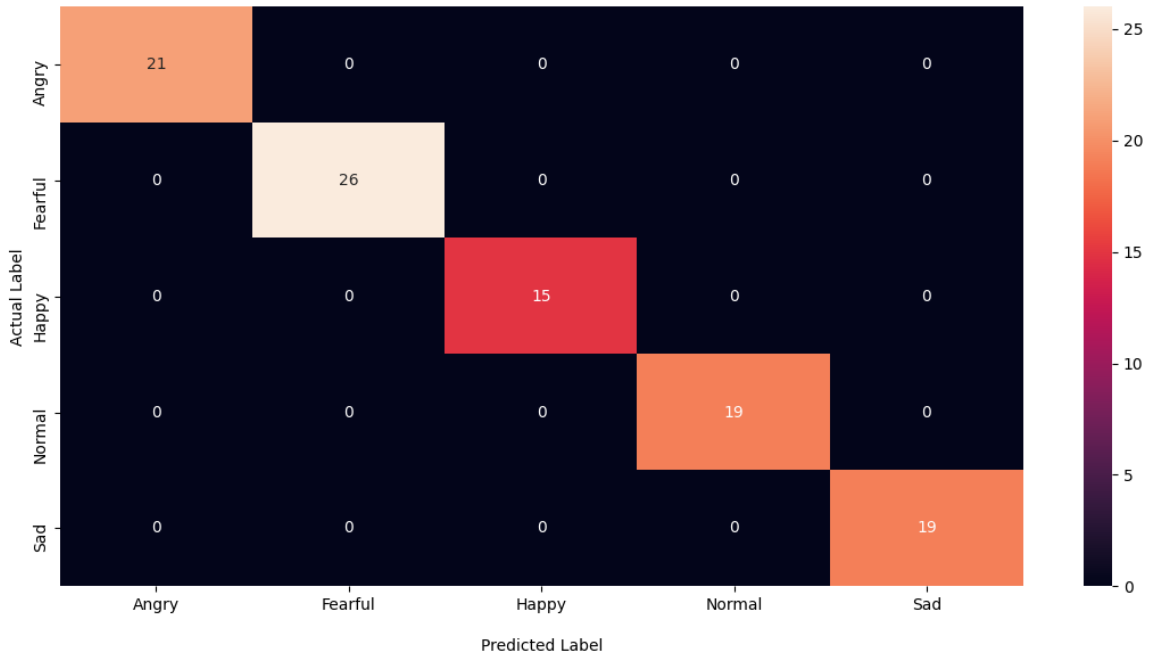


Figure 41. Confusion matrix for the S009 considering MEL, MFCC features using TPOT classifier

Figure 42 shows the confusion matrix after training the model on S009 users' dataset extracting MEL, MFCC, CGL features using TPOT. The model used 500 samples split into training and test data following 80%-20% train-test ratio. The training dataset consists of 400 samples and test dataset consists of 100 samples. The confusion matrix shows that among 100 test samples, 20 samples were classified correctly as angry data, 26 samples were classified correctly as fearful data, 15 samples were classified correctly as happy data, 19 samples were classified correctly as normal data, and 19 samples were classified correctly as sad data. The rest of the test samples were misclassified and results in 100% classification accuracy, which is 2% more than extracting MEL, MFCC features. This is a statistically significant improvement as the proposed methodology yields maximum possible accuracy.

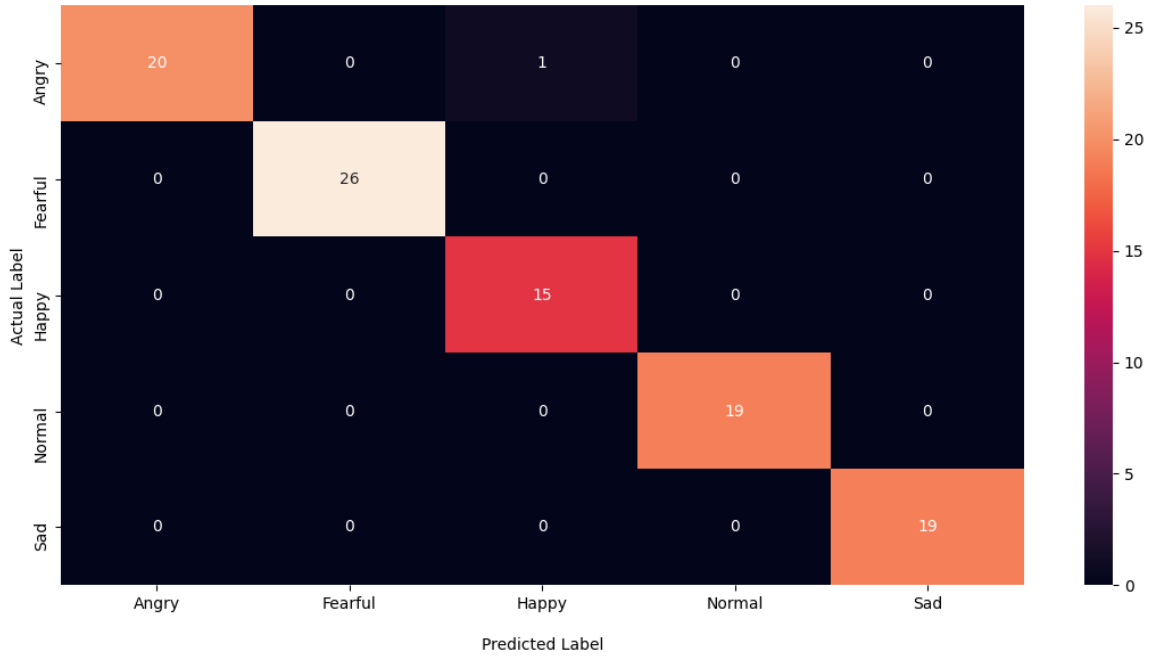


Figure 42. Confusion matrix for the S009 considering MEL, MFCC, CGL features using TPOT classifier

Figure 43 shows the confusion matrix after training the model on S010 users' dataset extracting MEL, MFCC features using TPOT. The actor is a male of 21-25 age and a native English speaker. The model used 500 samples split into training and test data following 80%-20% train-test ratio. The training dataset consists of 400 samples and test dataset consists of 100 samples. The confusion matrix shows that among 100 test samples, 21 samples were classified correctly as angry data, 23 samples were classified correctly as fearful data, 15 samples were classified correctly as happy data, 19 samples were classified correctly as normal data, and 19 samples were classified correctly as sad data. The rest of the test samples were misclassified and results in 98% classification accuracy.

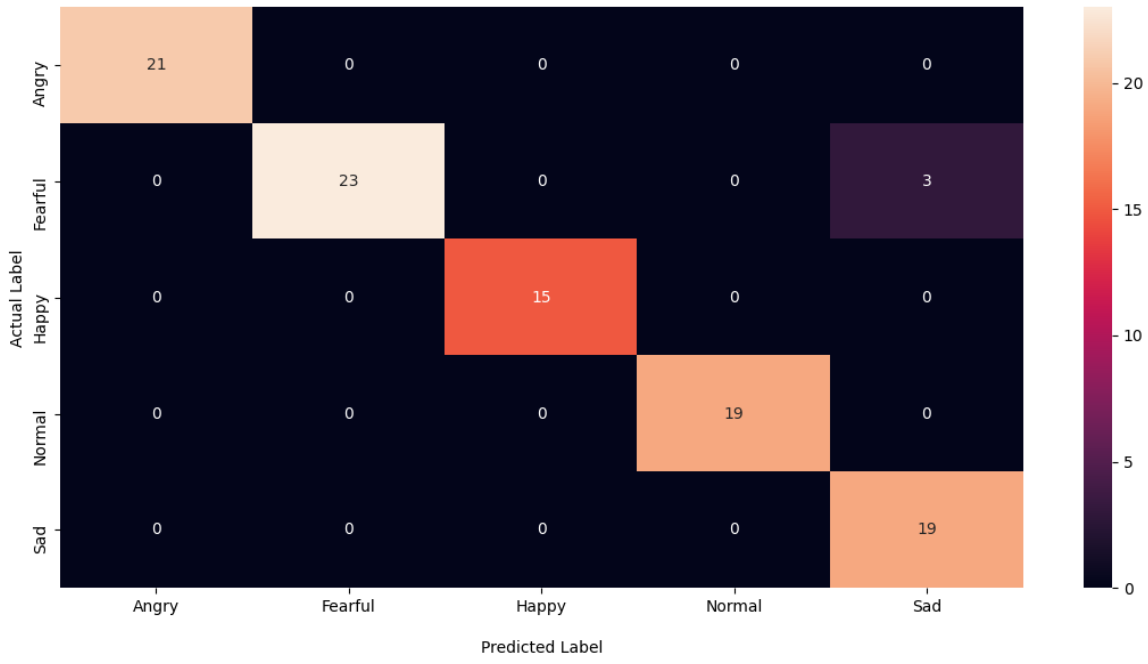


Figure 43. Confusion matrix for the S010 considering MEL, MFCC features using TPOT classifier

Figure 44 shows the confusion matrix after training the model on S010 users' dataset extracting MEL, MFCC, MCD features using TPOT. The model used 500 samples split into training and test data following 80%-20% train-test ratio. The training dataset consists of 400 samples and test dataset consists of 100 samples. The confusion matrix shows that among 100 test samples, 21 samples were classified correctly as angry data, 23 samples were classified correctly as fearful data, 15 samples were classified correctly as happy data, 19 samples were classified correctly as normal data, and 19 samples were classified correctly as sad data. The rest of the test samples were misclassified and results in 99% classification accuracy, which is 1% more than extracting MEL, MFCC features.

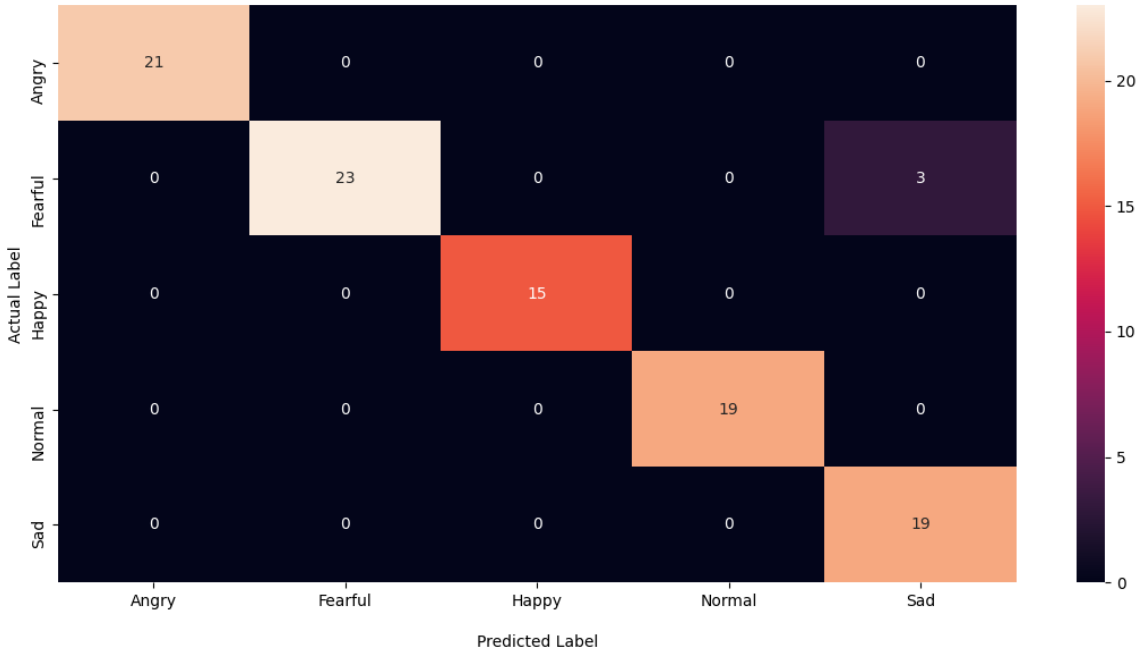


Figure 44. Confusion matrix for the S010 considering MEL, MFCC, MCD features using TPOT classifier

Figure 45 shows the confusion matrix after training the model on S010 users' dataset extracting MEL, MFCC, MWDPC features using TPOT. The model used 500 samples split into

training and test data following 80%-20% train-test ratio. The training dataset consists of 400 samples and test dataset consists of 100 samples. The confusion matrix shows that among 100 test samples, 21 samples were classified correctly as angry data, 23 samples were classified correctly as fearful data, 15 samples were classified correctly as happy data, 19 samples were classified correctly as normal data, and 19 samples were classified correctly as sad data. The rest of the test samples were misclassified and results in 99% classification accuracy, which is 1% more than extracting MEL, MFCC features. Under a chi-squared test, I found the probability value to be $p = 0.0246$, yielding a statistically significant improvement in the performance evaluation.

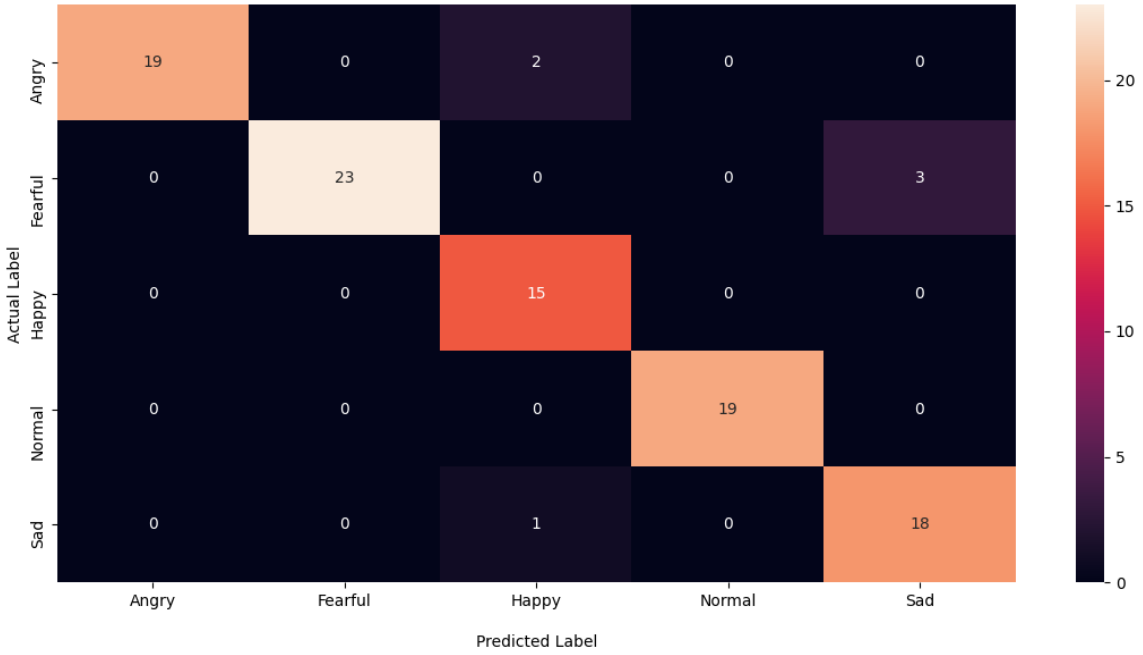


Figure 45. Confusion matrix for the S010 considering MEL, MFCC, MWDPC features using TPOT classifier

The accuracy obtained from Figure 24 to Figure 45 is summarized in Table 11. In summary, if the custom features are combined with MEL and MFCC, the accuracy gets better.

Table 11. Results for TPOT classifier using different participants of the smart home dataset

User	Features	TPOT
S001	MEL, MFCC	84%
	MEL, MFCC, MWDPC	87%
S002	MEL, MFCC	90%
	MEL, MFCC, PCWC	92%
	MEL, MFCC, MWDPC, PCWC	92%
S003	MEL, MFCC	69%
	MEL, MFCC, MCD, CGL	72%
S004	MEL, MFCC	97%
	MEL, MFCC, MWDPC	99%
S005	MEL, MFCC	90%
	MEL, MFCC, CGL	94%
S006	MEL, MFCC	92%
	MEL, MFCC, MCD	95%
S007	MEL, MFCC	99%
	MEL, MFCC, MCD, CGL	100%
S008	MEL, MFCC	89%
	MEL, MFCC, MWDPC	95%
S009	MEL, MFCC	98%
	MEL, MFCC, CGL	100%
S010	MEL, MFCC	98%
	MEL, MFCC, MCD	99%
	MEL, MFCC, MWDPC	99%

For user S001, the accuracy gets 3% better if MWDPC is combined with MEL and MFCC feature extraction. We can see the accuracy gets better for S002, S004, S008, and S010 users if MEL and MFCC are combined with MWDPC. We can observe better accuracy for S002 by combining PCWC feature extraction. The accuracy even gets better from 90% to 94% if CGL is combined with the existing features, for the S005 dataset. Also, we see the impact of CGL for

users S003, S007, and S009 whereas the accuracy gets better in extracting MCD along with MEL, MFCC for S003, S006, S007, and S010. Thus, Table 11, shows that we can detect emotion more accurately by combining MEL, and MFCC features with the custom features for the smart home dataset. Besides, I also showed the improvement in performance evaluation is statistically significant in most of the user cases based on the p value from chi-squared test.

Based on the observation depicted in this section, we can conclude that is there is no one universal custom feature that works better for all of them rather different custom features worked better for different users. Also, some users have really high accuracy compared to other users. Another observation is that native speakers have more accuracy than nonnative speakers. Moreover, some users are not different emotional states are not distinguishable enough.

Apart from testing the system by splitting the dataset into training and testing datasets, the model was also tested on unlabeled data in real-time. For this, MEL, MFCC, and one custom feature each time was used. Table 12 shows the result observed from applying different models with different combinations of features.

Table 12. Emotion detection for unlabeled data using the different combinations of features

Model	**Actual Emotion**	**Predicted Emotion**	**Accurately Detected**
MEL, MFCC	Happy	Happy	Yes
MEL, MFCC, CGL	Normal	Normal	Yes
MEL, MFCC, MCD	Fearful	Fearful	Yes
MEL, MFCC, MWDPC	Sad	Sad	Yes
MEL, MFCC, PCWC	Angry	Angry	Yes

CONCLUSIONS

Speech plays a pivotal role in the classification of emotions besides multiple factors, e.g., facial expressions as well as body language. Assessing the emotions of a human being based on these emotions through artificial intelligence is a promising research domain. The information related to facial expression and body language can be extracted through video and image processing and is already being employed in different applications. However, the scope of this research is concentrated on the smart home device applications, and interaction in the smart home environment is mostly dictated by speech. The rise in popularity of smart home devices (e.g., Amazon Alexa or Google Home) has incentivized the development of real-time emotion detection. Therefore, motivated by the need for smart devices to suggest appropriate actions based on the users' current emotional state, I have explored methods to determine the emotional state of a smart home user based on the short audio conversations with smart assistants. In this thesis, I presented my novel initiative to classify happy, normal, sad, fearful, and angry emotions from smart commands. Potential uses include tailoring the smart home services and the functionalities of smart devices to the user's current emotion or tracking the emotional states to improve the mental well-being of the user.

Voice commands given to smart home devices are short relative to typical human conversations, as they usually consist of 2-6 words. Because of this, short voice commands lack contextual data (i.e., lexical cues) that can be used to estimate the emotion of smart home device users. Therefore, emotion detection in the smart home environment is mostly dependent on the exact extraction of audio features combined with an appropriate classification algorithm. In this thesis, performed a comparison among several classifiers and observed that the XGBoost

classifier from TPOT AutoML performs best. I also proposed four novel audio features and combined them with the widely used MFCC and Mel Spectrogram features for emotion classification. Combining MEL, MFCC with CGL, MCD, MWDPC and PCWC features showed improvement in the emotion detection accuracy. Moreover, this accuracy improvement indicates that emotion prediction may increase if we consider a larger dataset containing smart home voice commands and audio conversations with the smart assistants which the publicly available datasets like RAVDESS as well as TESS do not provide. Therefore, I built a custom smart home audio dataset to confirm the results. My thesis highlights the importance of investigating more dynamic features from smart home audio conversations to ensure accurate emotion detection for individual users. The future scope of this research can be focused on increasing the custom dataset size and planning to incorporate advanced machine learning classifiers, such as Bi-LSTM and Deep Belief Network.

REFERENCES

[1] J. Hall and R. Iqbal, "Compes: A command messaging service for iot policy enforcement in a heterogeneous network," in *Proceedings of the Second International Conference on Internet-of-Things Design and Implementation (IoTDI)*, 2017, pp. 37–43.

[2] A. M. Badshah *et al.*, "Deep features-based speech emotion recognition for smart affective services," *Multimedia Tools and Applications*, vol. 78, no. 5, pp. 5571–5589, 2019.

[3] S. Z. Bong, K. Wan, M. Murugappan, N. M. Ibrahim, Y. Rajamanickam, and K. Mohamad, "Implementation of wavelet packet transform and non linear analysis for emotion classification in stroke patient using brain signals," *Biomedical Signal Processing and Control*, vol. 36, pp. 102–112, 2017.

[4] G. Shashidhar, K. Koolagudi, and R. Sreenivasa, "Emotion recognition from speech: a review," *International Journal of Speech Technology*, vol. 15, pp. 99–117, 2012.

[5] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things (IoT): A vision, architectural elements, and future directions," *Future Generation Computer Systems*, vol. 29, no. 7, pp. 1645–1660, 2013.

[6] F. K. Aldrich, "Smart Homes: Past, Present and Future," in *Inside the Smart Home*, R. Harper, Ed. London: Springer, 2003, pp. 17–39.

[7] A. S. Lago, J. P. Dias, and H. S. Ferreira, "Managing non-trivial internet-of-things systems with conversational assistants: A prototype and a feasibility experiment," *Journal of Computational Science*, vol. 51, p. 101324, 2021.

[8] E. Stefanidi, M. Foukarakis, D. Arampatzis, M. Korozi, A. Leonidis, and M. Antona, "ParlAmI: A Multimodal Approach for Programming Intelligent Environments," *Technologies*, vol. 7, no. 1, Art. no. 1, 2019.

[9] A. K. Dey, R. Hamid, C. Beckmann, I. Li, and D. Hsu, "a CAPpella: programming by demonstration of context-aware applications," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, New York, NY, USA, Apr. 2004, pp. 33–40.

[10] P. K. D. Pramanik, S. Pal, and P. Choudhury, "Beyond Automation: The Cognitive IoT. Artificial Intelligence Brings Sense to the Internet of Things," in *Cognitive Computing for Big Data Systems Over IoT: Frameworks, Tools and Applications*, Springer International Publishing, 2018, pp. 1–37.

[11] D. B. Shank, C. Graves, A. Gott, P. Gamez, and S. Rodriguez, "Feeling our way to machine minds: People's emotions when perceiving mind in artificial intelligence," *Computers in Human Behavior*, vol. 98, pp. 256–266, 2019.

[12] G. Kortuem, F. Kawsar, V. Sundramoorthy, and D. Fitton, "Smart objects as building blocks for the Internet of things," *IEEE Internet Computing*, vol. 14, no. 1, pp. 44–51, 2010.

[13] C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos, "Context Aware Computing for The Internet of Things: A Survey," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 1, pp. 414–454, 2014.

[14] M. W. Kraus, "Voice-only communication enhances empathic accuracy.," *American Psychologist*, vol. 72, no. 7, p. 644, 2017.

[15] Z. Zeng, M. Pantic, G. I. Roisman, and T. S. Huang, "A survey of affect recognition methods: Audio, visual, and spontaneous expressions," *IEEE Transactions on Pattern Analysis and Machine Intelligence.*, vol. 31, no. 1, pp. 39–58, 2008.

[16] K. S. Rao, S. G. Koolagudi, and R. R. Vempada, "Emotion recognition from speech using global and local prosodic features," *International Journal of Speech Technology*, vol. 16, no. 2, pp. 143–160, 2013.

[17] R. Cowie and E. Douglas-Cowie, "Automatic statistical analysis of the signal and prosodic signs of emotion in speech," in *Proceeding of Fourth International Conference on Spoken Language Processing. (ICSLP)*, Oct. 1996, vol. 3, pp. 1989–1992 vol.3.

[18] A. Bombatkar, G. Bhoyar, K. Morjani, S. Gautam, and V. Gupta, "Emotion recognition using Speech Processing Using k-nearest neighbor algorithm," *International Journal of Engineering Research and Applications (IJERA)*, pp. 2248–9622, 2014.

[19] M. Borchert and A. Dusterhoft, "Emotions in speech-experiments with prosody and quality features in speech for use in categorical and dimensional emotion recognition environments," in *International Conference on Natural Language Processing and Knowledge Engineering*, 2005, pp. 147–151.

[20] A. B. Kandali, A. Routray, and T. K. Basu, "Emotion recognition from Assamese speeches using MFCC features and GMM classifier," in *TENCON 2008-2008 IEEE Region 10 conference*, 2008, pp. 1–5.

[21] L. Cen, H. L. Z. L. Yu, M. Dong, and P. Chan, "Machine learning methods in the application of speech emotion recognition", *INTECH Open Access Publisher*, 2010.

[22] H. Meng, T. Yan, F. Yuan, and H. Wei, "Speech emotion recognition from 3D log-mel spectrograms with deep learning network," *IEEE Access*, vol. 7, pp. 125868–125881, 2019.

[23] Z. Tariq, S. K. Shah, and Y. Lee, "Speech emotion detection using iot based deep learning for health care," in *IEEE International Conference on Big Data (Big Data)*, 2019, pp. 4191–4196.

[24] A. K. Samantaray, K. Mahapatra, B. Kabi, and A. Routray, “A novel approach of speech emotion recognition with prosody, quality and derived features using SVM classifier for a class of North-Eastern Languages,” in *IEEE 2nd International Conference on Recent Trends in Information Systems (ReTIS)*, 2015, pp. 372–377.

[25] 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.

[26] M. K. Pichora-Fuller and K. Dupuis, “Toronto emotional speech set (TESS),” *Sch. Portal Dataverse*, 2020.

[27] T. S. Polzin and A. H. Waibel, “Recognizing emotions in speech,” in *Proceeding of Fourth International Conference on Spoken Language Processing*, 1996, vol. 3, pp. 1970–1973.

[28] A. Milton, S. S. Roy, and S. T. Selvi, “SVM scheme for speech emotion recognition using MFCC feature,” *International Journal of Computer Applications*, vol. 69, no. 9, 2013.

[29] N. Majumder, S. Poria, D. Hazarika, R. Mihalcea, A. Gelbukh, and E. Cambria, “Dialoguernn: An attentive rnn for emotion detection in conversations,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2019, vol. 33, no. 01, pp. 6818–6825.

[30] D. Hazarika, S. Poria, A. Zadeh, E. Cambria, L.-P. Morency, and R. Zimmermann, “Conversational memory network for emotion recognition in dyadic dialogue videos,” in *Proceedings of the Conference. Association for Computational Linguistics. North American Chapter. Meeting*, vol. 2018, p. 2122.

[31] F. Eyben, M. Wöllmer, and B. Schuller, “Opensmile: the munich versatile and fast open-source audio feature extractor,” in *18th ACM International Conference on Multimedia*, 2010, pp. 1459–1462.

[32] S. Poria, D. Hazarika, N. Majumder, G. Naik, E. Cambria, and R. Mihalcea, “Meld: A multimodal multi-party dataset for emotion recognition in conversations,” *ArXiv Preprint. ArXiv181002508*, 2018.

[33] S. Mirsamadi, E. Barsoum, and C. Zhang, “Automatic speech emotion recognition using recurrent neural networks with local attention,” in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017, pp. 2227–2231.

[34] C. Busso *et al.*, “IEMOCAP: Interactive emotional dyadic motion capture database,” *Language Resources and Evaluation*, vol. 42, no. 4, pp. 335–359, 2008.

[35] M. Sarma, P. Ghahremani, D. Povey, N. K. Goel, K. K. Sarma, and N. Dehak, “Emotion Identification from Raw Speech Signals Using DNNs.,” in *Interspeech*, 2018, pp. 3097–3101.

[36] W. Lim, D. Jang, and T. Lee, “Speech emotion recognition using convolutional and recurrent neural networks,” in *2016 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA)*, 2016, pp. 1–4.

[37] M. Lech, M. Stolar, C. Best, and R. Bolia, “Real-Time Speech Emotion Recognition Using a Pre-trained Image Classification Network: Effects of Bandwidth Reduction and Companding,” *Frontiers in Computer Science*, vol. 2, 2020.

[38] Z. Huang, M. Dong, Q. Mao, and Y. Zhan, “Speech Emotion Recognition Using CNN,” in *Proceedings of the 22nd ACM International Conference on Multimedia*, 2014, pp. 801–804.

[39] I. Shahin, A. B. Nassif, and S. Hamsa, “Emotion Recognition Using Hybrid Gaussian Mixture Model and Deep Neural Network,” *IEEE Access*, vol. 7, pp. 26777–26787, 2019.

[40] B. Javaheri, “Speech & Song Emotion Recognition Using Multilayer Perceptron and Standard Vector Machine,” *ArXiv210509406 Cs Eess*, 2021.

[41] A. M. Badshah, J. Ahmad, N. Rahim, and S. W. Baik, “Speech Emotion Recognition from Spectrograms with Deep Convolutional Neural Network,” in *2017 International Conference on Platform Technology and Service (PlatCon)*, Feb. 2017, pp. 1–5.

[42] B. M. Nema and A. A. Abdul-Kareem, “Preprocessing signal for Speech Emotion Recognition,” *Al-Mustansiriyah Journal of Science*, vol. 28, no. 3, Art. no. 3, 2017.

[43] P. B. Dasgupta, “Detection and analysis of human emotions through voice and speech pattern processing,” *ArXiv Preprint. ArXiv171010198*, 2017.

[44] Y. Shi and W. Song, “Speech emotion recognition based on data mining technology,” in *Sixth International Conference on Natural Computation*, 2010, vol. 2, pp. 615–619.

[45] J. Shen *et al.*, “Natural tts synthesis by conditioning wavenet on mel spectrogram predictions,” in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 4779–4783.

[46] K. Venkataramanan and H. R. Rajamohan, “Emotion recognition from speech,” *ArXiv Preprint. ArXiv191210458*, 2019.

[47] F. Zheng, G. Zhang, and Z. Song, “Comparison of different implementations of MFCC,” *Journal of Computer Science and Technology*, vol. 16, no. 6, pp. 582–589, 2001.

[48] V. Tiwari, “MFCC and its applications in speaker recognition,” *International Journal on Emerging Technologies*, vol. 1, no. 1, pp. 19–22, 2010.

[49] L. Muda, M. Begam, and I. Elamvazuthi, “Voice recognition algorithms using mel frequency cepstral coefficient (MFCC) and dynamic time warping (DTW) techniques,” *ArXiv Preprint. ArXiv10034083*, 2010.

[50] W. Han, C.-F. Chan, C.-S. Choy, and K.-P. Pun, “An efficient MFCC extraction method in speech recognition,” in *IEEE International Symposium on Circuits and Systems*, 2006, pp. 4-pp.

[51] P. M. Chauhan and N. P. Desai, “Mel frequency cepstral coefficients (MFCC) based speaker identification in noisy environment using wiener filter,” in *International Conference on Green Computing Communication and Electrical Engineering (ICGCCEE)*, 2014, pp. 1–5.

[52] C. Praksah and V. Gaikwad, “Analysis of emotion recognition system through speech signal using KNN, GMM & SVM classifier,” *IOSR J Electron Commun Eng IOSR-JECE*, vol. 10, no. 2, pp. 55–67, 2015.

[53] P. Auer, H. Burgsteiner, and W. Maass, “A learning rule for very simple universal approximators consisting of a single layer of perceptrons,” *Neural Networks.*, vol. 21, no. 5, pp. 786–795, 2008.

[54] S. Kim, K. Haruma, M. Ito, S. Tanaka, M. Yoshihara, and K. Chayama, “Magnifying gastroendoscopy for diagnosis of histologic gastritis in the gastric antrum,” *Digestive and Liver Disease*, vol. 36, no. 4, pp. 286–291, 2004.

[55] M. W. Gardner and S. Dorling, “Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences,” *Atmospheric Environment*, vol. 32, no. 14–15, pp. 2627–2636, 1998.

[56] O.-W. Kwon and T.-W. Lee, “Optimizing speech/non-speech classifier design using adaboost,” in *IEEE International Conference on Acoustics, Speech, and Signal Processing, 2003. Proceedings (ICASSP).*, 2003, vol. 1, p. I–I.

[57] S. Hamsa, I. Shahin, Y. Iraqi, and N. Werghi, “Emotion recognition from speech using wavelet packet transform cochlear filter bank and random forest classifier,” *IEEE Access*, vol. 8, pp. 96994–97006, 2020.

[58] F. Hutter, J. Lücke, and L. Schmidt-Thieme, “Beyond manual tuning of hyperparameters,” *KI-Künstliche Intelligenz*, vol. 29, no. 4, pp. 329–337, 2015.

[59] R. S. Olson and J. H. Moore, “TPOT: A tree-based pipeline optimization tool for automating machine learning,” in *Workshop on Automatic Machine Learning*, 2016, pp. 66–74.

[60] R. S. Olson, N. Bartley, R. J. Urbanowicz, and J. H. Moore, “Evaluation of a tree-based pipeline optimization tool for automating data science,” in *Proceedings of the Genetic and Evolutionary Computation Conference*, 2016, pp. 485–492.

[61] M. A. Ali and P. M. Shemi, “An improved method of audio denoising based on wavelet transform,” in *2015 International Conference on Power, Instrumentation, Control and Computing (PICC)*, 2015, pp. 1–6.

[62] R. Jannat, I. Tynes, L. L. Lime, J. Adorno, and S. Canavan, “Ubiquitous emotion recognition using audio and video data,” in *Proceedings of the 2018 ACM International Joint Conference and 2018 International Symposium on Pervasive and Ubiquitous Computing and Wearable Computers*, 2018, pp. 956–959.

[63] A. Giese and J. Seitzer, “Using a Genetic Algorithm to Evolve a D* Search Heuristic”. In *MAICS,* 2011, p. 67-72.

[64] U. Garg, S. Agarwal, S. Gupta, R. Dutt, and D. Singh, “Prediction of Emotions from the Audio Speech Signals using MFCC, MEL and Chroma,” in *12th International Conference on Computational Intelligence and Communication Networks (CICN)*, 2020, pp. 87–91.

[65] D. Issa, M. F. Demirci, and A. Yazici, “Speech emotion recognition with deep convolutional neural networks,” *Biomedical Signal Processing and Control*, vol. 59, p. 101894, 2020.