POLITECNICO DI MILANO

School of Industrial and Information Engineering
Master of Science in Computer Science and Engineering

Title: **Emotion Recognition from Video Using Transfer Learning and Stacking**

Supervisor: prof. Licia Sbattella
Co-Supervisor: Ing. Roberto Tedesco/PHD/,
Francesco Cottone

Master Thesis of:

Samsom Tsegay Beyene Matr. 894380

Academic Year: 2018-2019

Abstract

Human beings express emotions through different types of modalities. Analyzing these different modalities helps to create a system in which the emotional state is expressed more clearly and hence easier to understand. Exploring the focus to various expression channels can fasten research on emotion recognition in addition to human-machine interaction. In this paper, we propose a multi-channel hybrid fusion architecture which classifiers an emotion expression into one of the most common categories: anger, disgust, fear, happiness, neutral, sadness, and surprise. The proposed video (audio-visual) emotion recognition method has experimented on the different labels of human emotions from both audio and video datasets. The classification is done into two phases. first, audio and visual classifiers are trained separately on audio data and visual data respectively. In the second phase, audio and video features are combined to train another final classifier based on stacking generalization technique. Audio data are transformed into a spectrogram and the spectrogram is given to a VGGsih, a CNN pre-trained model, as a base model for feature extraction. Similarly, visual features we are computed from a sequence of frames representing each video clip. Visual features are extracted using CNN based pre-trained model called VGG16. After the base classifiers for each of the channels are trained separately, they are used as feature extractors for another fused final classifier. The second last layer output values of the base classifiers are fused together to train the stacked classifier, which is a final classification predictor. The performance of the experimented stacked audio-visual classifier results with an overall accuracy of 87%, which is more comparable than to the recognition by humans.

Keywords: VGGish, VGG16, CNN, Stacking generalization, Emotion recognition, Pre-trained model, feature extraction, spectrogram, Fusion.

Acknowledgement

First of all, I would like to thank my company supervisor, Mr Francesco Cottone, for his tremendous help with my research design, thesis writing and continuous comments. Without his support, my thesis might not be possible. I also would like to thank Ing. Roberto Tedesco/PHD/ for his incisive comments on my research design and thesis writing. I thank my thesis advisor Prof. Licia Sbattella with her support to do in the area of emotion recognition application and her help for completion of this work.

Lastly, I thank Vidiemme Consulting for providing me with a harmonious environment in doing my thesis work internship.

Contents

1 Introduction **10**

1.1 Motivations 11

1.2 Aims 11

1.3 Thesis Structure 12

2 Background: Emotion Recognition **13**

2.1 Emotion 13

2.1.1 Emotion definitions 13

2.1.2 Emotion expression types 14

2.1.3 Emotion Usage in Computer Technology 15

2.1.4 Emotion recognition Models and Techniques 16

2.1.5 Emotion Recognition Applications 17

2.2 Research paper findings in emotion recognition 19

2.3 Discussion: Pros and Cons 23

3 Background: Neural Networks **26**

3.1 Introduction to Neural Networks 26

3.2 Deep learning 29

3.2.1 Convolutional Neural Network: 29

3.2.2 Pre-trained models 32

3.2.3 Transfer Learning 33

3.3 LSTM 34

3.3.1 Introduction to LSTM 34

3.3.2 LSTM structure: Cell state 35

3.3.3 LSTM structure: Gates 35

4 Methodology **38**

4.1 RAVDASS dataset 38

4.1.1 RAVDASS structure 38

4.1.2 actors 40

4.1.3 Ground truth annotations 40

4.1.4 Adaptation of the dataset 41

4.2 Model feature Extraction 41

4.2.1 Audio feature extraction 41

4.2.2 Visual feature extraction 42

4.2.3 Stacked feature extraction 43

4.2.4 Why Transfer learning for Feature selection 43

4.3 Model structure 46

4.3.1 Model overview 46

4.3.2 Audio model 48

4.3.3 Visual Model 49

4.3.4 Stacked model 51

5 Model Implementation **54**

5.1 Software used 54

5.1.1 Programming language 54

5.1.2 Tools and packages 55

5.2 Training modules 58

5.2.1 Hyper-parameters 59

5.2.2 Loss function 60
5.2.3 Optimizer 61
5.3 Testing modules 63

6 Results and Discussion **65**
6.1 Audio Model results 65
6.1.1 Accuracy on training 66
6.1.2 Accuracy on testing 67
6.2 Visual model results 68
6.2.1 Accuracy on training 68
6.2.2 Accuracy on testing 69
6.3 Stacked model results 70
6.3.1 Accuracy on training 71
6.3.2 Accuracy on testing 73
6.4 Final comments 75

7 Conclusion and future work **77**
7.1 Conclusions 77
7.2 Feature Work 77

Appendix **79**

References **87**

List of Figures

1 Principle structure of the fuzzy classification 19

2 Framework for emotion recognition 20

3 The architecture of CNN used 21

4 Overview of the paper approach 22

5 Video affective analysis architecture 23

6 HHT based audio feature extraction algorithm 25

7 HHT based visual feature based feature extraction 25

8 Artificial Neuron architecture 26

9 Activation functions for Neurons 27

10 Feedforward Neural Network 27

11 Recurrent Neural Network: Elman Network 28

12 CNN Architecture 30

13 Convolution operation example 30

14 pooling operation 31

15 LSTM repeating module 35

16 LSTM cell 35

17 LSTM gates 36

18 Forget gate 36

19 Update gate 36

20 Output gate 37

21 Selected samples from RAVDESS database 40

22 Proposed Architecture for Emotion Recognition 47

23 Audio module architecture 48

24 Audio model classifier. 49

25 Visual Module Architecture 50

26 Visual Classifier Model 51

27 Stacked Classifier Model 52

28 VGG16 Architecture 53

29 VGGish Architecture 53

30 categorical cross-entropy loss. 61

31 Audio classifier on the training dataset 66

32 Audio Classifier on Testing data 67

33 Visual Classifier on Training data 69

34 Visual Classifier on Testing data 70

35 Stacked model's Accuracy for training and validation dataset . . 71

36 Stacked model's loss for training and validation dataset 71

37 Stacked Classifier on training data 72

38 Stacked Classifier performance on testing data 73

39 Three models Comparison on the test dataset 74

List of Tables

1 audio classifier model summary 64

2 Stacked model's Heatmap on training data 72

3 Stacked model's Heatmap on testing data 73

Acronyms

AI —— Artificial Intelligence
CNN —— Convolution Neural Network
COG —— Center of Gravity
DNN —— Deep Neural Network
ELM —— Extreme Learning Machine
HHT —— Hilbert Huang Transform
LBP —— Local Binary Pattern
LSTM —— Long Short Term Memory
MFCC —— Mel Frequency Cepstral Coefficient
PLS —— Partial least squares
RBM —— Restricted Boltzmann machine
RDM —— Run Difference Method
RNN —— Recurrent Neural Network
SVM —— Support Vector Machine

1 Introduction

Human communication is either intrapersonal communication, communication with oneself, or interpersonal communication, communication with two or more people. Interpersonal communication is expressed in different ways such as non-verbal communication, speech, conversation, visual communications and others. These communication types collectively or individually determine the person's feeling, intention, and emotion while interacting with other people. For this particular paper, we are interested in how emotion can be recognized from human conversations.

As a general Emotion Recognition problem and human-computer interaction perspective, there are generally six different emotion descriptors. Theses are facial expression, speech and vocal intonation, physiological signals, body gesture and pose, text and combination of two or more of the approaches. Facial expressions provide the most informative data for computer awareness of emotions [27]. However, software applications that use facial expressions have a number of restrictions that mostly limit their accuracy and applicability. Usually, they can only manage a small set of expressions from a frontal view of faces without facial hair and glasses, and they require good and stable lighting conditions. Another emotion descriptor called vocal intonations produces less accurate emotion recognition results than facial expression approaches. Vocal intonation analysis can only manage a subset of basic emotions from the recorded audio files or from the speech streams that come from microphones. These require post-processing through various speech analysis methods [30].

Other types of emotion descriptor called Physiological sensors allow for capturing a variety of physiological responses such as body temperature, heart rate, blood volume, and skin conductance of an individual [17]. These sensors are sometimes offered in the form of wearable devices [23]. Although such technologies show promising results in emotion recognition, it is scarcely applied, because it is obtrusive to learners and requires expensive and dedicated equipment [18]. Moreover, body movements and gestures are an additional source of emotion recognition descriptors. Even though they are supposed to be more emotional descriptors, they are not capable of extracting the user's emotions [28]. Lastly, Text and speech are also as a means for expressing human emotions. Since all of the above-mentioned approaches are language-dependent, it is difficult and challenging to develop a universal world-wide application for international society. Another pitfall for emotion recognition is the users may not necessarily express

their own emotions, but describes somebody else's emotions or sometimes they do not express the real emotion feeling in a sentence explicitly.

1.1 Motivations

During human dialogue conversation, people communicate with each other by sending and receiving information in different common modalities such as voice, body movement, facial expressions, text, gestures and psychological changes. The modalities mentioned above are broadly categorized into verbal and non-verbal components. According to Ko, Byoung [15] verbal components convey one-third of human communication whereas non-verbal components convey two-thirds. Hence, understanding the attitude and moods of one's behaviour is very crucial to have a well mannered, fruitful and consistent conversation among people in general and for human-robot (chatbot) interaction in particular. In both of the verbal and nonverbal categories, by carrying emotional meaning, emotion expressions are one of the main information channels in interpersonal communication. Therefore, it is natural that research on emotion recognition is gaining a lot of attention in recent years than the past in different computer and human interaction applications. So this motivates us to implement and see the performance on emotion recognition with a currently hot and demanding technology called deep learning.

1.2 Aims

In recent years human emotion recognition is becoming a hot demand in different commercial fields such as psychiatric counselling for mental health care services [22], intelligent tutoring systems [1], job interviews [12], entertainment[26], robotics [19, 21] and even monitoring agents in call centres. Since most of, if not all, these applications are developed and trained to a specific database, they face the issues of bias in ethnicity, age, sex, culture, contextual meaning of the sentences on other nature circumstances. In addition, the emotion recognizer system developed so far are trained using one specific type of data: they use one channel data source.

So this paper focuses on the problem of emotion recognition from video(Audio-visual) dataset. To put it in another way, its main purpose is to classify a given emotion expression sample to one of the common universal categories: anger, disgust, fear, happiness, neutral, sadness, and surprise. In this particular case,

a single video clip has only a single type of emotion for both the audio and visual part. For the visual part, a facial expression is used as a means of input data. From the video, frames are extracted at a rate of 45 fps followed by face detection, cropping and resizing. In the audio data, each audio clip is converted into its corresponding spectrogram formant. For both data sources, a separate deep learning state-of-the-art pre-trained models are used as feature extractors. Stable and good models are trained to learn each feature space independently, and a new brand final classifier model is trained, based on stacking generalization principle, on new feature space composed of the second last layer outputs of the individual classifiers. The conducted experiment on Ryerson Audio-Visual Database of Emotional Speech and Song(RAVDESS) shows that the third stacked classifier has a significant outperformance in comparison to the individual model classifiers.

1.3 Thesis Structure

The rest of the paper is organized as the following outline: Section II, background on emotion recognition and related work on audio-visual emotion recognition is discussed. In section III, a brief background on the neural network is covered. In section IV, the proposed method is explained in detail and is followed by section V that describes the model implementation. Experimental results and discussions are presented in section VI. Finally, paper conclusion and future work are provided in section VII.

2 Background: Emotion Recognition

2.1 Emotion

2.1.1 Emotion definitions

Emotion has a variety of definition based on different studies that are conducted at different time zones. According to [24] emotions can be defined as a positive or negative experience that is associated with a particular pattern of physiological activity. Emotions also produce different physiological, behavioural and cognitive changes. The study says the original role of emotions was to motivate adaptive behaviours that in the past would have contributed to the passing on of genes through survival, reproduction, and kin selection. In other theories , cognition is an important aspect of emotion.

Those acting primarily on the emotions they are feeling may seem as if they are not thinking, but mental processes are still essential, particularly in the interpretation of events. For example, the realization of our believing that we are in a dangerous situation and the subsequent arousal of our body's nervous system (rapid heartbeat and breathing, sweating, muscle tension) is integral to the experience of our feeling afraid. Other theories, however, claim that emotion is separate from and can precede cognition. Consciously experiencing an emotion is exhibiting a mental representation of that emotion from a past or hypothetical experience, which is linked back to a content state of pleasure or displeasure [31].

Another study theories even state emotions are states of feeling that result in physical and psychological changes that influence our behaviour [9]. The physiology of emotion is closely linked to arousal of the nervous system with various states and strengths of arousal relating, apparently, to particular emotions. Emotion is also linked to a behavioural tendency. Extroverted people are more likely to be social and express their emotions, while introverted people are more likely to be more socially withdrawn and conceal their emotions.

Emotion is also often perceived the driving force behind motivation, negative and positive behavioural aspects. The different components of emotion are categorized somewhat differently depending on the academic discipline. In psychology and philosophy, emotion typically includes a subjective, conscious experience characterized primarily by psychophysiological expressions, biological reactions, and mental states. A similar multi componential description of emotion is found in

sociology. For example, Peggy Thoits [29] described emotions as involving physiological components, cultural or emotional labels (anger, surprise, fear, happiness, etc.), expressive body actions, and the appraisal of situations and contexts. In this paper, we consider the emotion labels of the physiological with seven distinct labels (anger, disgust, fear, happiness, neutral, sadness and surprise).

2.1.2 Emotion expression types

Human beings express their emotion or feeling in a number of different ways using verbal or non-verbal communication. Body language such as a slouched posture or crossed arms can be used to send different emotional signals. Even though facial expressions and audio signals are assumed to be more universal, they still lack them universality which is affected by innate and cultural behavioural adaptations. Being said this, expressions used for expressing emotion could have many emotion labels according to the culture and context of time and occasion. But expressions that used to convey basic emotions can be categorized into anger, disgust, fear, happiness, neutral, sadness and surprise. These seven basic emotions are more universal usually expressed using face and audio data which can be detected by people across the world. Based on [2] the basic seven emotions are described below so as to have a brief understanding of how they differ from each other and their unique behaviour.

Anger: an emotion that is often associated with the range of minor irritation to intense rage. Physically, anger causes someone to experience an increased heart rate, heightened blood pressure, and abnormal levels of adrenaline and noradrenaline. Anger is characterized by a facial expression that causes someone to lower their brows, press their lips together firmly, and bulge their eyes.

Disgust: this is an emotion that is usually associated with things that are unsanitary, inedible, infectious, or offending. Disgust is also characterized by a facial expression that causes someone to raise their upper lip, wrinkle their nose bridge, and raise their cheeks.

Fear: it is one of the emotions that is always associated with a threatening or dangerous stimulus. In other way saying, fear is a basic survival mechanism that occurs in response to a traumatic presence, such as a pain or the impending threat of pain. When experiencing fear, it is said to cause a person to raise their brows, open their mouth slightly, and open their eyes in a manner that is wider than normal.

Happiness: an emotion which is often associated with a state of mind that reflects contentment, satisfaction, pleasure or joy. Happiness is one of the most popular emotions, and it has been studied throughout different philosophical, religious, and biological approaches. Happiness is characterized by a facial expression that causes someone to raise the corners of their mouth upwards.

Neutral: one of the universal emotions that are always associated with a state of mind that reflects a normal feeling. Neutral is characterized by a facial expression that causes someone to keep the mouth closed.

Sadness: an emotion that is often associated with feelings of disadvantage, loss, and helplessness. Usually, humans react to being sad by getting quiet, and they experience a lack of energy and a need to be withdrawn. Sadness is also characterized by a facial expression that causes someone to lower corners of their mouths, and raise the inner portion of their brows.

Surprise: it is an emotion that is often associated with a brief state of being. This brief state of being is invoked by an unexpected, relevant event. The surprise is characterized by a facial expression that causes someone to arch their brows, open their eyes widely, and drop their jaw.

2.1.3 Emotion Usage in Computer Technology

There are enormous alternative means that human beings can interact with computers. The means and type of communication between humans and computers are crucial to facilitate the interaction and process information accordingly. One important aspect of Human-computer interaction is user satisfaction or affective satisfaction. To achieve this user satisfaction many human-machine interaction studies suggests that supporting knowledge on both the machine and human side is required. On the machine side, computer graphics, operating system, programming language and development environment is relevant. Linguistics, social sciences, cognitive psychology, social psychology, and human factors such as computer user are relevant from the human side. Therefore, capturing this required knowledge are sometimes a little bit difficult when it comes to humans because human psychology is a very complex system that needs to be learnt by machines. Here, we try to explain how emotion expression affects the human to computer interaction and its usage.

Emotion is cognitive psychology which is one of the most communication channels for a human to human interactions. It basically influences how humans commu-

nicate and connect with each other. Humans with high human emotional intelligence lead more successful professional and personal lives are more likeable and more persuasive, tend to be more effective leaders, and generally lead healthier, happier and even longer lives [25]. In this fourth industrial revolution, also known as digital world and artificial intelligence, researchers are trying to integrate the business devices or applications to have the essence of emotion recognizer with the entity they are interacting with. This implies that the applications (computers or chatbots) have to be developed to analyze emotion expressions, interpret the emotions and respond accordingly with a certain accuracy. As a result, researchers suggested that in the coming few years most digital products and applications will incorporate with an emotion module that will help for understanding the feeling and emotion of a client.

2.1.4 Emotion recognition Models and Techniques

In the growing field of affective computing, there are so many algorithm techniques that are under use or are emerging as hot research topics. These techniques vary based on a type of application people are intending to use it and the type of data source that the application is going to function. Being said this , the more general term that is appealing for affecting computing in general and emotion recognition, in particular, is the power of Artificial intelligence. For this broad field machine learning and computer vision techniques are the one which is commonly used. Therefore, deep exploration and exploitation of these fascinating methods are indispensable for emotion recognition system.

Emotion recognition can be modelled using different channel sources. The main types of source channels currently undergoing are Emotion from Text, Emotion from Speech and Emotion from Video. Emotion recognition from text needs input as a sequence of words which are provided from a linguistic part of a speech. The model that is designed to recognize the emotion of a person needs the linguistic meaning of the words to classify the sentiment analysis as positive, negative and neutral. This method is limited not because it is hard to find a good capture for more basic emotion label classifications like sad, anger, surprise, and others but because it is language-dependent. The second type of emotion model type is a model which accepts audio data only. This type of model receives an acoustic data form the user and tries to recognize emotion from the speech. The problem of this emotional recognizer model is that it is not capturing more information from a facial expression which is the most emotion expresser part of the human body. The last model is the one which uses video as it's input source. This model

takes visual and audio from the given video. The model trained from the video is now more robust than a model trained either from audio-only or text only because it is obtaining more informative emotion data from the most informative emotion part called facial expression. Having these different types of models which are receiving different data source, let us explain one by one the techniques used to these models to train and test.

Whatever the dataset or the model type we choose, classical machine learning and deep learning are the current top leading fields that are getting more attention for developing an emotion recognizer application. In the classical part, the user is expected to explicitly specify the descriptive features in a handcrafted way. This method may perform well in a situation where the data has not complex spatial dimensionality and they do not need a domain expert. On the contrary, deep learning does not need handcrafted features to train their model. What they do is they try to learn from the data eventually and keep the key features that generalize the data properties. This helps developers to develop a robust and consistent emotion recognizer. So it is up to the developer according to use case provided. In this paper, a deep learning methodology is used.

2.1.5 Emotion Recognition Applications

Apparently, there are enormous applications that are smarter enough regarding autoimmunity. Starting from simple personal assistance devices up to large commercialized in giant high tech companies, they are trying their product to be smarter by capturing human emotions so as to have a consistent dialogue conversation. Most researchers and business observers related to this field are speculating that by the early 20^{th} 10% of personal devices will have an emotion AI capabilities either on device or cloud services [3]. Consequently, new use case offers are demanding from industries and organizations to create a better customer experience, preferences and unlock real cost savings. The following use cases are among the most upfront area of fields that are stated by the paper [3] where emotion recognition is going to play a great role.

Video gaming: Using computer vision, the game console/video game detects emotions via facial expressions during the game and adapts to it. This detection of emotion helps to know the player's feeling while playing the game. Once the emotion is known the application tunes itself that hook people in and keeps them playing.

Medical diagnosis: Software can help doctors with the diagnosis of diseases such as depression and dementia by using voice analysis. The emotion part also plays a great role in reading some information that the person express visually.

Education: Learning software prototypes have been developed to adapt to kids' emotions. When the child shows frustration because a task is too difficult or too simple, the program adapts the task so it becomes less or more challenging. Another learning system helps autistic children recognize other people's emotions.

Employee safety: Based on Gartner client inquiries, demand for employee safety solutions are on the rise. Emotion AI can help to analyze the stress and anxiety levels of employees who have very demanding jobs such as first responders.

Patient care: A 'nurse bot' not only reminds older patients on long-term medical programs to take their medication but also converses with them every day to monitor their overall wellbeing.

Car safety: Automotive vendors can use computer vision technology to monitor the driver's emotional state. An extreme emotional state or drowsiness could trigger an alert for the driver.

Autonomous car: In the future, the interior of autonomous cars will have many sensors, including cameras and microphones, to monitor what is happening and to understand how users view the driving experience.

Fraud detection: Insurance companies use voice analysis to detect whether a customer is telling the truth when submitting a claim. According to independent surveys, up to 30% of users have admitted to lying to their car insurance company in order to gain coverage.

Recruiting: Software is used during job interviews to understand the credibility of a candidate. Call centre intelligent routing: An angry customer can be detected from the beginning and can be routed to a well-trained agent who can also monitor in real-time how the conversation is going and adjust.

Connected home: A VPA-enabled speaker can recognize the mood of the person interacting with it and respond accordingly.

Public service: Partnerships between emotion AI technology vendors and surveillance camera providers have emerged. Cameras in public places in the United Arabic Emirates can detect people's facial expressions and, hence, understand the general mood of the population. This project was initiated by the country's

Ministry of Happiness.

Retail: Retailers have started looking into installing computer vision emotion AI technology in stores to capture demographic information and visitors' mood and reactions.

2.2 Research paper findings in emotion recognition

There are several available papers conducted on emotion recognition that use audio only, visual only or both. They follow different algorithms and methodologies for feature extraction, feature selection, model selection and other procedures to implement their work. [6] proposed an emotion classifier named Fuzzy Emotion Recognition in Natural Speech Dialogue which uses fuzzy logic as a classifier that reduces the classification error. This paper is specifically designed for robot head MEXI which often communicates with well-known persons. Here, the rule-based approach is used to extracted features to classify emotion based on the given rule. It is more training for the dependent speaker than a universal application. It performs less in the universal case.

It takes a single sentence as input and classifies it into the given emotion labels in this case five: happiness, sadness, anger, fear and neutral, with a certain degree. The paper uses the centre of gravity (COG) as a defuzzification method. The computed degrees for each emotion are then compared by PROSPER and the strongest emotion is returned as recognized. Figure 1 shows the principal structure of the fuzzy classification.

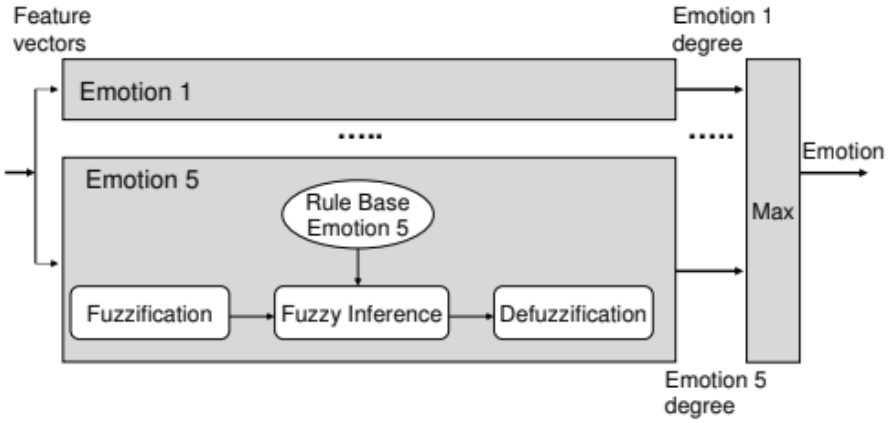


Figure 1: Principle structure of the fuzzy classification

Another research by Kyo-Joong Oh, DongKun Lee, ByungSoo Ko, Ho-Jin Choi [22] proposed a multimodal interface model titled Chatbot for Psychiatric Coun-

seling in Mental Healthcare Service Based on Emotional Dialogue Analysis and Sentence Generation. In this architecture, the emotion recognizer is integrated with other modules of the counselling application. It uses the RNN and LSTM model for learning the classifier models. Here, multimodal refers to the application can receive in different input format like text, audio, video or voice but they are not combined together to detect the emotion at the same time.

The paper first collects corpora of target languages. The different models for their text, audio, and video are trained independently according to their specific configuration of the corpus collected and feature representation. The emotion module of the application is shown in figure 2

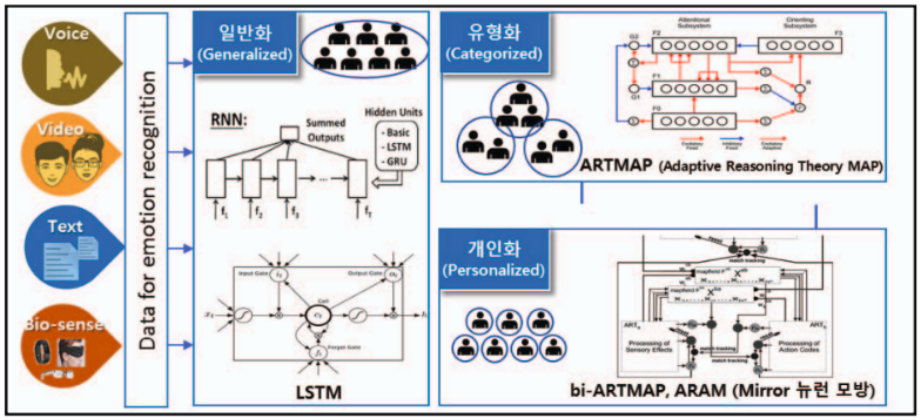


Figure 2: Framework for emotion recognition

A research paper by Bertero, Dario and Siddique, Farhad Bin and Wu, Chien-Sheng and Wan, Yan and Chan, Ricky Ho Yin and Fung, Pascale [8] titled Real-Time Speech Emotion and Sentiment Recognition for Interactive Dialogue Systems also conducted an experiment using a CNN deep learning a model classifier. In this, the CNN is used both as feature extractor in the convolution and max pool layer and followed by fully connected layer classifier Performs well compared to SVM one. This paper is limited to audio features only and does not consider any visual part.

Another article Bahreini, Kiavash and van der Vegt, Wim and Westera, Wim [7] for capturing school learners emotion titled 'Emotions are a significant influential factor in the process of learning' is developed based mixed algorithm expressions. It uses FURIA algorithm for unordered fuzzy rule induction to offer timely and appropriate feedback based on learners' facial expressions. It uses fuzzy logic as a classifier model. In addition, Duncan, Dan and Shine, Gautam and English, Chris [10] Facial Emotion Recognition in Real-Time is implemented based on CNN in

real-time streaming. Here they use transfer learning on the fully connected layers of CNN and trained the model over a variety of datasets. Figure 3 shows the whole architecture of this paper.

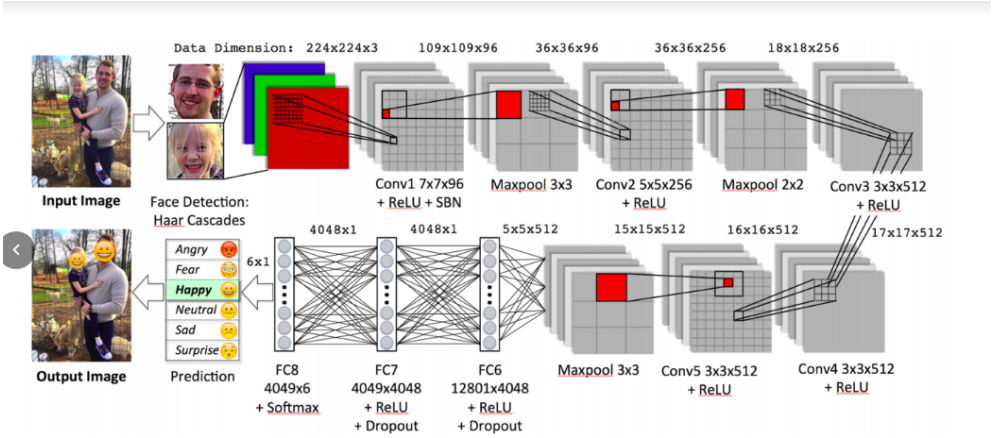


Figure 3: The architecture of CNN used

After the model was trained an image from a live video using a face detector is fed and the model eventually classifies the detected emotion to one of the emotion categories. Classifying emotions can be difficult depending on whether the input image is static or a transition frame into a facial expression. Static (frame-based) depends on static facial features obtained by handcrafted features selected from peak expression frames of image sequences. Dynamic(video-based) utilities Spatio-temporal features to capture dynamics in facial expression sequence. So this paper is lacking robustness since it only considers static facial features. There is still a need for dynamic (transitional state) features to be addressed.

There is another paper done by Zhao, Kaili and Chu, Wen-Sheng and Zhang, Honggang [32] that integrates both the spatial features as well as temporal variations using CNN for the first one and LSTM for the latter one. the CNN is used as a feature extractor and LSTM helps the model to capture the emotion of the user. Here the LSTM is used to correlate the emotion between different timestamps not how the sequence of frames is changing to predict the emotion.

According to Kaya, Heysem and Gürpınar, Furkan and Salah, Albert Ali [14] Video-based emotion recognition in the wild using deep transfer learning and score fusion paper, emotion recognition is a very challenging method because of noise, large idiosyncratic variance, and sensor-related differences. In this article, the steps are face detection, face alignment-based certain model, feature extraction and classification. It uses a CNN based feature extraction method via transfer learning. The paper adopts an approach that combines audio and visual features with kernel ELM PLS based classifiers followed by weighted score level fusion.

The general design model of this paper is shown in figure 4

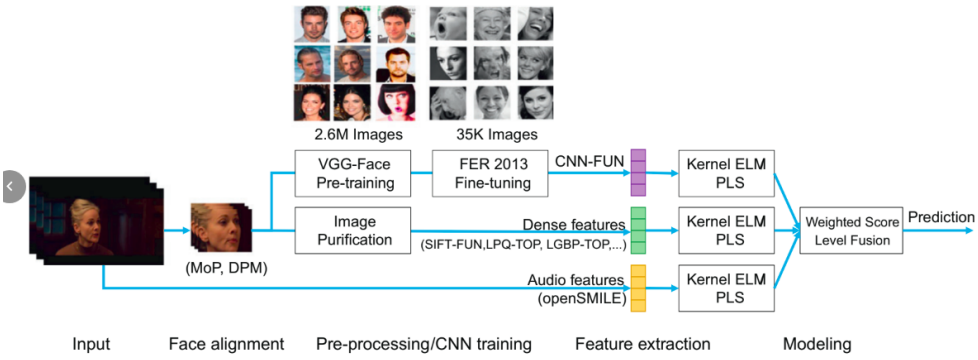


Figure 4: Overview of the paper approach

Based on the article Mo, Shasha and Niu, Jianwei and Su, Yiming and Das, Sajal K [20] titled A novel feature set for video emotion recognition that considers both audio and visual features. Auditory features include zero-crossing rate, sound energy, pitch, onset, beat strength, tempo, bandwidth, spectral centroid, spectral roll-off, spectral flux, loudness, and Mel-frequency Cepstrum Coefficients (MFCC) et al., And also visual features mainly include lighting, saturation, color energy, color heat, short-change rate, shot duration, shot type transition rate, motion intensity, and motion dynamics et al. In short, HHTC features based on the combination of Hilbert–Huang Transform (HHT) based visual features, HHT-based audio features, and cross-correlation features is proposed in this paper. An SVM model is also used as a classifier model that is trained on the extracted features. The main limitation of this model is it uses a hand-crafted feature engineering extraction method and hence more computation and might lead to an inaccurate selection of features.

Another more convincing paper is Ashwin, TS and Saran, Sai and Reddy, G Ram Mohana [5] approach, which is Video Affective Content Analysis Based on Multimodal Features Using a Novel Hybrid SVM-RBM. The classifier of this model uses both audio and visual features(from video streaming or stored) to categorize the emotions. It uses the MFCC method and Active Appearance Model as feature extraction methodology for acoustic and visual data respectively. In this methodology, it uses a simultaneous selection of both acoustic and visual features for the discrete emotion category and hybrid of SVM-RBM classifier for each of the feature spaces. The more description of this model is shown in figure 5

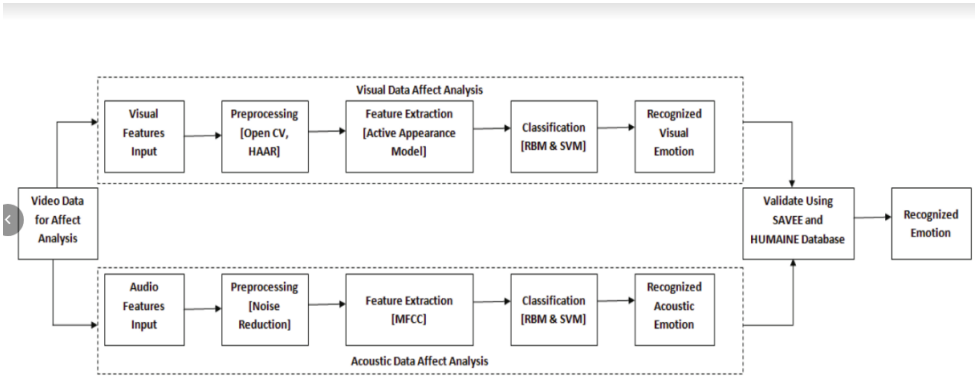


Figure 5: Video affective analysis architecture

In all of the above paperwork, the researchers proposed the state-of-the-art on either using audio-visual or one of them without taking time variation into account. They use a single static frame mapped to an emotion category rather than considering a sequence of frames to help a model to learn a pattern of how frames are changing through given time for each emotion expression. Moreover, during the fusion of the acoustic-visual, they used confidence output of each model and averaging according to a given weight and take this as final emotion predictor.

In this work, we focus on alleviating these limitations mentioned above by proposing a new architecture for each audio and visual dataset followed by a third Stacked classifier as final emotion recognizer. For the visual model classifier, we used a CNN-LSTM hybrid model. We used a pre-trained CNN model called VGG16 for feature extraction and LSTM as classifier through transfer learning. Likewise, a visual model a similar approach is used for the audio case i.e a VGGish pre-trained audio model is used for feature extraction and a simple Neural Network is used as a classifier. Finally, the second layer output of the two models for a separate data that is not used for training the two models is used as a new feature vector representation to train a stacked classifier in order to obtain final emotion prediction.

2.3 Discussion: Pros and Cons

The comparison of our proposed method with other papers described in section 2.3 can be seen into two categories: papers with a single input channel and papers with a multi-input channel. We present the pros and cons of both category papers with our proposed architecture for emotion recognition.

Articles [6], [22], [8], [7], [10] and [32] are based on a single dataset source channel

that can be either text, audio or visual, which are used to train their designed model. The paper [6] uses a fuzzy rule as a classification algorithm. It is particularly designed for a MEXI application. The positive side of this application is its performance good for a dependent user but it is not a robust model compared to our model and others as it uses a manual feature extraction method and deep learning techniques. Similarly, papers [22], [8], [7], [10], and [32] proposed a classification model that is trained from a single input channel. [22] has an interface to accept text, audio or visual but one at a time. Moreover, article [8] uses audio database and [7], [10], [32] articles use facial expressions (visual) dataset for training and testing. These models use the convolutional neural network(CNN) followed by a dense layer techniques for features extraction and classification purpose, which is good from feature extraction and classification perspective. The limitation of these papers is that they train their model from scratch and they do not try to combine features of different input channels as our model combines both audio and visual data.

The other side of papers we have reviewed are those which use more than one input channel for training and testing their emotion classifier models. Papers like [14], [20], [5] use both audio and visual input channels. Article [14] uses both visual as well as audio data to classify a given emotion. It uses open smile methodology for audio feature extraction and CNN for visual feature extraction. Kernel extreme learning machine (ELM) based classifier is trained for both audio and visual model classifiers. This model uses a partial least square (PCS) as a measure of the model performance. The combinations of the two data features is done at score level particularly it uses weighted score level fusion. Another paper that considers both audio and visual datasets titled with a novel feature set for video emotion recognition [20] applies a Hilbert Huang Transform (HHT) based approach for both audio and visual feature extraction. The HHT based follows cross-correlation, Hilbert transform, cross-correlation features and sum. The combined sum features are used to train the SVM classifier. The HHT procedures for both audio and visual are shown in figure 6 and 7 respectively. This model needs more calculation precision during the feature extraction part.

Last but not least research paper which is similar to our model is the one with title video affective content analysis based on multi-model features using hybrid SVM-RBM [5]. This architecture processes the two datasets in parallel and fuses them at the end. The visual part uses Active Appearance Model as a feature extraction methodology. Then the extracted visual features are fed into an SVM classifier. Similarly, the audio part applies the MFCC representation feature extraction approach. Moreover, RBM classifier model is trained for the audio

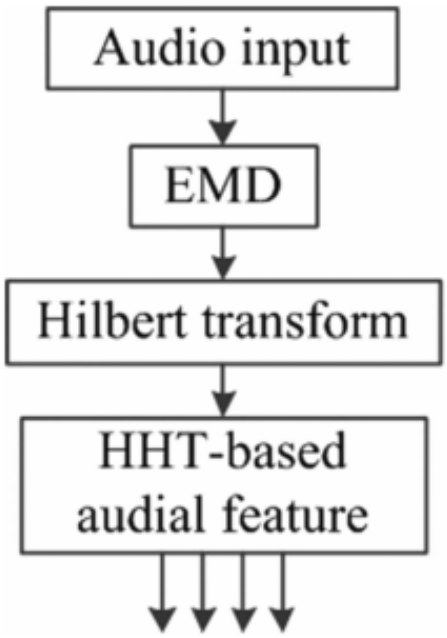


Figure 6: HHT based audio feature extraction algorithm

classification. The scores of these two models are fused together for final emotion recognition.

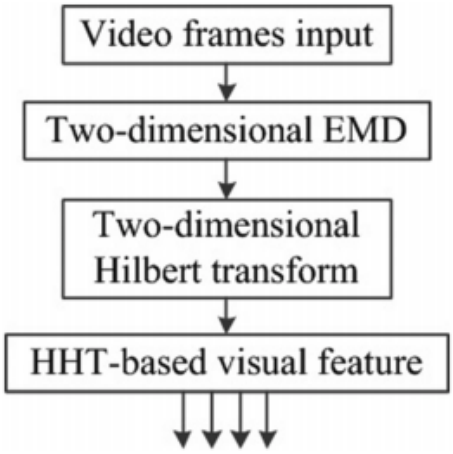


Figure 7: HHT based visual feature based feature extraction

The important part of all the above video based multi-channel input architectures ([14], [20], [5]) is they combine both audio and visual features for their emotion recognition. The limitation of theses architectures compare to our model is they do not entertain the ensembles predictive force called stacking generalization. They simply fuse the score of the base models to predict rather than using these fused features to train another meta-classifier which is usually called stacked classifier. Our model entertains this kind of predictive force scalability and the meta-classifier is used as our final emotion classifier.

3 Background: Neural Networks

3.1 Introduction to Neural Networks

Neural networks are literally a network or circuit of neurons that are computational models which work similarly to the functioning of a human nervous system. The neural can be a biological neural or artificial neural. A biological neural network is made of real biological neurons whereas the artificial neural is for solving artificial intelligence problems. Artificial intelligence, cognitive modelling, and neural networks are information processing paradigms inspired by the way biological neural systems process data. Artificial intelligence and cognitive modelling try to simulate some properties of biological neural networks. In the artificial intelligence field, artificial neural networks have been applied successfully to speech recognition, image analysis and adaptive control, in order to construct software agents (in computer and video games) or autonomous robots.

The building block of a neural network is called neuron. A neural network which contains a single neuron is called perceptron. It takes some inputs, do some math with them and produces one output. Figure 8 shows this very basic neural network (perceptron) architecture.

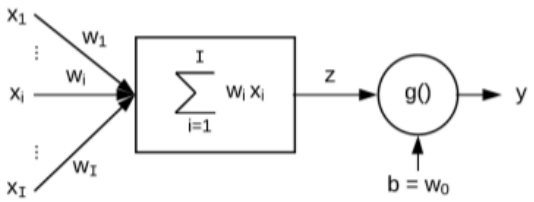


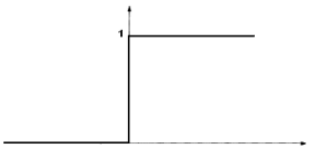
Figure 8: Artificial Neuron architecture

In the above model, it contains the number of inputs, weights, activation functions and the output. The inputs are the dataset fed into the network for processes. Other parameters are the weights of the artificial neural which are sometimes called synapses. The activation function is the one which translates the computed value by the neuron to a specific bound. And the last one is the output of the network.

For the activation of the network, someone can use different common activation functions based on the specific application he/she is interested. The most common are shown in figure 9 with their generative formula.

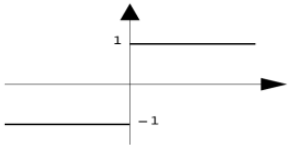
Step function

$$g(z) = \begin{cases} 1 & \text{if } z \geq 0 \\ 0 & \text{if } z < 0 \end{cases}$$



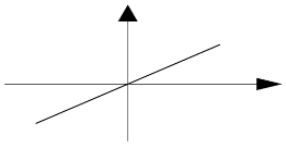
Sign function

$$g(z) = \begin{cases} 1 & \text{if } z > 0 \\ 0 & \text{if } z = 0 \\ -1 & \text{if } z < 0 \end{cases}$$



Linear function

$$g(z) = z$$



Sigmoid function

$$g(z) = \frac{1}{1 + e^{-z}}$$

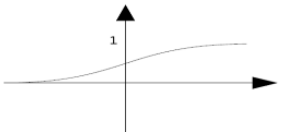


Figure 9: Activation functions for Neurons

The simple model shown in figure 3.1 is the basic architecture for any complex neural network as well as deep learning architectures. There are different types of neural networks not only based on the way they learn and data flow pipeline among the neurons but also mathematical operations and a set of parameters required to determine the output. Some of the common neural network types [1] are described below.

Feedforward Neural Network: This neural network is one of the simplest forms of artificial neural network(ANN), where the data or the input travels in one direction. The data passes through the input nodes and exit on the output nodes. This neural network may or may not have the hidden layers. Figure 10 shows a feedforward neural network with 2 hidden layers with a single unit output layer.

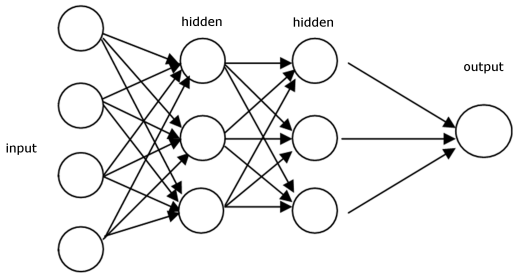


Figure 10: Feedforward Neural Network

Radial basis function Neural Network(RBF): Radial basis functions consider the distance of a point with respect to the center. RBF functions have two layers, first where the features are combined with the Radial Basis Function in the inner layer and then the output of these features are taken into consideration while computing the same output in the next time-step which is basically a memory.

Kohonen Self Organizing Neural Network: The objective of a Kohonen map is to input vectors of arbitrary dimension to discrete map comprised of neurons. The map needs to be trained to create its own organization of the training data. It comprises of either one or two dimensions. When training the map the location of the neuron remains constant but the weights differ depending on the value. This self organization process has different parts, in the first phase every neuron value is initialized with a small weight and the input vector. In the second phase, the neuron closest to the point is the winning neuron and the neurons connected to the winning neuron will also move towards the point.

Recurrent Neural Network(RNN): The Recurrent Neural Network works on the principle of saving the output of a layer and feeding this back to the input to help in predicting the outcome of the layer. Here, the first layer is formed similar to the feed forward neural network with the product of the sum of the weights and the features. The recurrent neural network process starts once this is computed. This means that from one time step to the next each neuron will remember some information it had in the previous time-step. This makes each neuron act like a memory cell in performing computations. One type of recurrent neural network called elman networks is shown in figure 11.

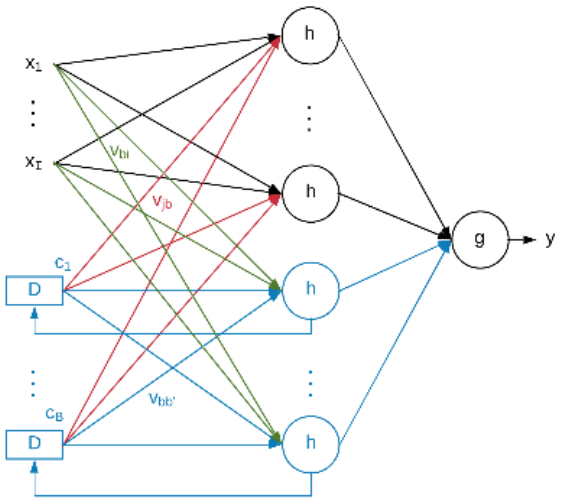


Figure 11: Recurrent Neural Network: Elman Network

Convolutional Neural Network: Convolutional neural networks are similar to feed forward neural networks , where the neurons have learn-able weights and biases. Its application have been in signal and image processing which takes over OpenCV in the field of computer vision.

Modular Neural Network: Modular Neural Networks have a collection of different networks working independently and contributing towards the output. Each neural network has a set of inputs which are unique compared to other networks constructing and performing sub-tasks. These networks do not interact or signal each other in accomplishing the tasks.

From the different types of neural networks described above, we only use the recurrent neural network and convolutional neural network architectures. These are the most common practices in deep neural network applications and have so many utilities. One of convolutional neural networks utilities that are used in this paper is transfer learning. Similarly in RNN particularly LSTM has the capability of storing previous states in a gate and cells for later use. Since this paper uses these architectures, the two architectures are described in detail below.

3.2 Deep learning

Deep learning is one of the major modern machine learning methods based on artificial neural networks. There are many types of deep learning based on the architecture they use. To list some of them: convolutional neural network, deep neural network, deep belief networks, Recurrent neural networks and others. These architectures have produced results comparable to in some cases superior to human experts in different applications. A brief explanation for some common terms and architectures of deep learning used in this paper are discussed below.

3.2.1 Convolutional Neural Network:

A Convolutional Neural Network, also known as CNN or ConvNet, is a class of deep neural networks that specializes in processing data that has a grid-like topology, such as an image. The architecture of a ConvNet is analogous to that of the connectivity pattern of Neurons in the Human Brain and was inspired by the organization of the Visual Cortex. Individual neurons respond to stimuli only in a restricted region of the visual field known as the Receptive Field. A collection of such fields overlap to cover the entire visual area. The role of the

ConvNet is to reduce the images or matrix of pixels into a form which is easier to process, without losing features which are critical for getting a good prediction. Typically, a CNN has three main layers: convolutional layer, pooling layer and fully-connected layer as shown in figure 12. Each of the layers of CNN is described below according to [4].

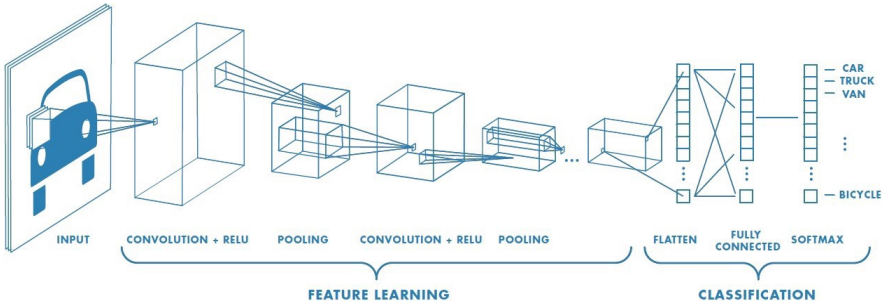


Figure 12: CNN Architecture

Convolutional Layer:

The convolution layer is the core building block of CNN. It carries the main portion of the network's computational load. This layer performs a dot product between two matrices, where one matrix is the set of learnable parameters which is also known as a kernel, and the other matrix is the restricted portion of the receptive field. The kernel is spatially smaller than an image but is more in-depth. This means that, if the image is composed of three (RGB) channels, the kernel height and width will be spatially small, but the depth extends up to all three channels.

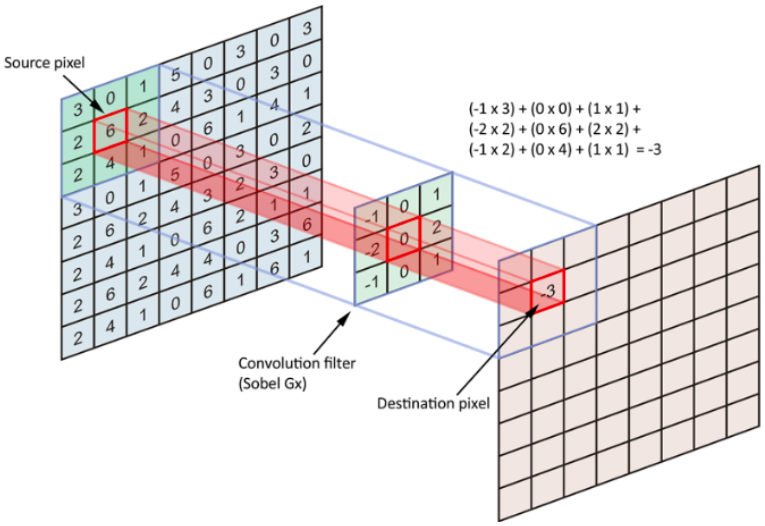


Figure 13: Convolution operation example

During the forward pass, the kernel slides across the height and width of the

image-producing the image representation of that receptive region. This produces a two-dimensional representation of the image known as an activation map that gives the response of the kernel at each spatial position of the image.

The sliding size of the kernel is called a stride. Figure 13 shows how convolution works for a single kernel iteration. This process is repeated on all the pixels and iterates until it reached the required number of features.

Pooling Layer:
The pooling layer replaces the output of the network at certain locations by deriving a summary statistic of the nearby outputs. This helps in reducing the spatial size of the representation, which decreases the required amount of computation and weights. The pooling operation is processed on every slice of the representation individually.

There are several pooling functions such as the average of the rectangular neighbourhood, L2 norm of the rectangular neighbourhood, and a weighted average based on the distance from the central pixel. However, the most popular process is max pooling, which reports the maximum output from the neighbourhood. This operation is shown in Figure 14.

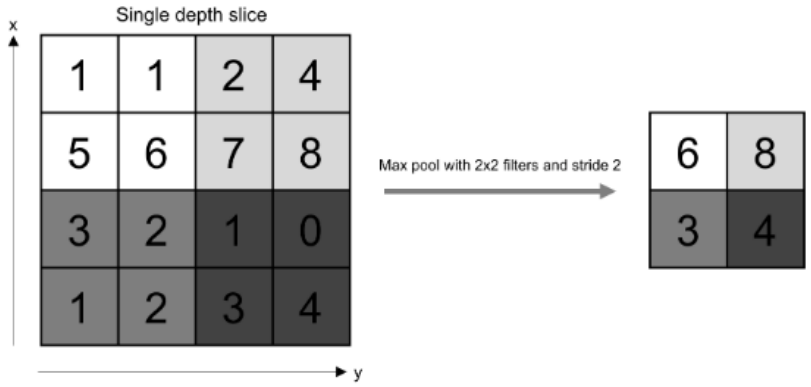


Figure 14: pooling operation

Fully connected Layer:
Neurons in this layer have full connectivity with all neurons in the preceding and succeeding layer as seen in regular CNN. This is why it can be computed as usual by a matrix multiplication followed by a bias effect. The FC layer helps map the representation between the input and the output. Since convolution is a linear operation and images are far from linear, non-linearity layers are often placed directly after the convolutional layer to introduce non-linearity to the activation map. There are several types of non-linear operations, the ones which are popular

are Sigmoid, tanh, Relu.

3.2.2 Pre-trained models

A pre-trained model is a saved model that was trained on a large benchmark dataset to solve a particular problem and can be customizable(reusable) in similar applications. the pre-trained model can be used either as it is or use transfer learning to customize this model to a specific task. Typically, pre-trained models can be used for two purposes: feature extraction and fine-tuning.

Feature extraction:
Use the representations learned by a previous network to extract meaningful features from new samples. we simply add a new classifier, which will be trained from scratch, on top of the pre-trained model so that we can repurpose the feature maps learned previously for our dataset. We do not need to retrain the entire model. The base convolutional network already contains features that are generically useful for classifying pictures. However, the final classification part of the pre-trained model is specific to the original classification task and subsequently specific to the set of classes on which the model was trained.

Fine-Tuning:
Unfreezing a few of the top layers of a frozen model base and jointly training both the newly-added classifier layers and the last layers of the base model. This allows us to fine-tune the higher-order feature representations in the base model in order to make them more relevant for the specific task. Currently, there are many pre-trained models that can be used as one of the above-mentioned functionalities based on a particular purpose. Each of these models is trained on different large-scale datasets and they have different architectures and way of learning. Below is a short brief description of common pre-trained models.

Xception
It is a convolutional neural network that is trained on more than a million images from the ImageNet database The network is 71 layers deep and can classify images into 1000 object categories, such as keyboard, mouse, pencil, and many animals. As a result, the network has learned rich feature representations for a wide range of images. The network has an image input size of 299-by-299.

VGG16: It is a convolutional neural network model proposed for Very Deep Convolutional Networks for Large-Scale Image Recognition. It is trained on ImageNet, which is more than 14 million images dataset belonging to 1000 categories.

Similarly, there is an upgraded version of VGG16 called VGG19.

There are more pre-trained models that are trained in the same Imagenet but with different architectures and number of layers such as ResNet50, InceptionV3, InceptionResNetV2, MobileNet, MobileNetV2, DenseNet, NASNet. Another pre-trained model trained with weight on AudioNet developed and released by Google research group is called **VGGish**. This is developed for the purpose of audio classification which has a similar design architecture like the VGG pre-trained model for ImageNet.

3.2.3 Transfer Learning

Transfer learning is a machine learning method where a model developed for a task is reused as the starting point for a model which is used for a similar task. It is a popular approach in deep learning where pre-trained models are used as the starting point on computer vision, speech analysis and natural language processing tasks given the vast computation and time resources required to develop neural network models on these problems and from the huge jumps in a skill that they provide on related problems.

In transfer learning, we first train a base network on a base dataset and task, and then we repurpose the learned features or transfer them to a second target network to be trained on a target dataset and task. This process will tend to work if the features are general, meaning suitable to both base and target tasks, instead of specific to the base task. Transfer learning differs from traditional machine learning is that it is the use of pre-trained models that have been used for another task to jump-start the development process on a new task or problem.

When we are repurposing a pre-trained model for our own needs, we start by removing the original classifier, then we add a new classifier that fits our purposes, and finally, we have to fine-tune, if applicable, our model according to one of three strategies:

Train the entire model:
In this case, we use the architecture of the pre-trained model and train it according to our dataset. We are learning the model from scratch, so we will need a large dataset and a lot of computational power.

Train some layers and leave the others frozen:
Lower layers refer to general features (problem independent), while higher layers

refer to specific features (problem-dependent). Here, we play with that dichotomy by choosing how much we want to adjust the weights of the network (a frozen layer does not change during training). Usually, if we have a small dataset and a large number of parameters, we will leave more layers frozen to avoid overfitting. By contrast, if the dataset is large and the number of parameters is small, we can improve the model by training more layers to the new task since overfitting is not an issue.

Freeze the convolutional base:
This case corresponds to an extreme situation of the train/freeze trade-off. The main idea is to keep the convolutional base in its original form and then use its outputs to feed the classifier. We are using the pre-trained model as a fixed feature extraction mechanism, which can be useful if we are short on computational power, our dataset is small, and/or pre-trained model solves a problem very similar to the one you want to solve.

3.3 LSTM

3.3.1 Introduction to LSTM

Long Short Term Memory networks usually just called LSTMs – are a special kind of recurrent neural network(RNN), capable of learning long-term dependencies. They were introduced by Hochreiter Schmidhuber (1997), and were refined and popularized by many people in the following work. They work tremendously well on a large variety of problems, and are now widely used in different time series and similar applications. LSTMs are explicitly designed to avoid long-term dependency problems. Remembering information for long periods of time is practically their default behavior, not something they struggle to learn.

All recurrent neural networks have the form of a chain of repeating modules of neural network. In standard RNNs, this repeating module will have a very simple structure, such as a single tanh layer. These modules are shown in figure 15.

In figure 15, each line carries an entire vector, from the output of one node to the inputs of others. The pink circles represent pointwise operations, like vector addition, while the yellow boxes are learned neural network layers. Lines merging denote concatenation, while a line forking denote its content being copied and the copies going to different locations. The components of LSTM architecture shown in figure 15 are categorized as cell and gate.

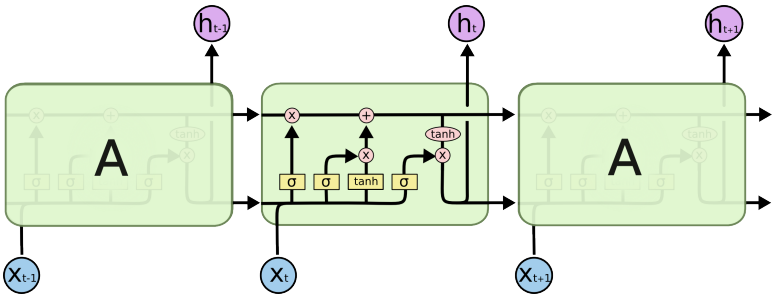


Figure 15: LSTM repeating module

3.3.2 LSTM structure: Cell state

The key to LSTMs is the cell state, the horizontal line running through the top of the diagram. The cell state is kind of like a conveyor belt. It runs straight down the entire chain, with only some minor linear interactions. It is very easy for information to just flow along it unchanged. The cell part of the LSTM is depicted in figure **??**.

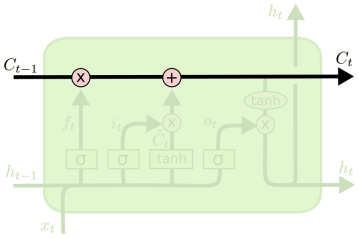


Figure 16: LSTM cell

The LSTM does have the ability to remove or add information to the cell state, carefully regulated by structures called gates. The main task of a cell is to control how much previous data should be reserved to compute the current computation.

3.3.3 LSTM structure: Gates

Every LSTM module has three main gates: Forget gate, update/input gate and output gate. These gates help the LSTM to work coordinately and in a convenient way for appropriate applications like time series applications. The gates of LSTM are shown in figure 17 followed by a short description.

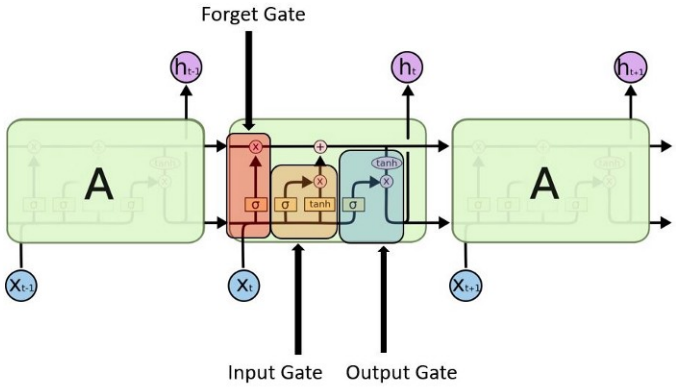
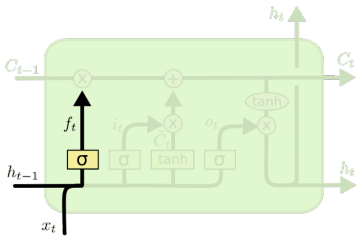


Figure 17: LSTM gates

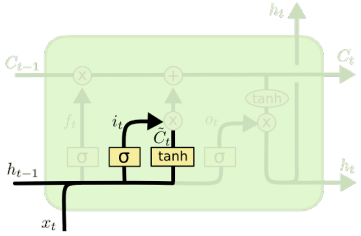
Forgate Gate:
describes how much of the past we have to consider. This gate decides which information to be omitted from the cell in that particular timestamp. It is decided by the sigmoid function. it looks at the previous state(ht-1) and the content input(Xt) and outputs a number between 0(omit this)and 1(keep this)for each number in the cell state Ct1. forget gate structure is shown in figure 18.



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

Figure 18: Forget gate

Update/input gate:
This gate decides how much of this unit is added to the current state. Sigmoid function decides which values to let through 0 or 1 and tanh function gives weight age to the values which are passed deciding their level of importance ranging from -1 to 1. The update gate of the LSTM is presented in figure 19.



$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

Figure 19: Update gate

Output gate: It decides which part of the current cell makes it to the output. Sigmoid function decides which values to let through 0,1. and tanh function gives weight age to the values which are passed deciding their level of importance ranging from -1 to 1 and multiplied with output of Sigmoid. This gate structure is shown in figure 20.

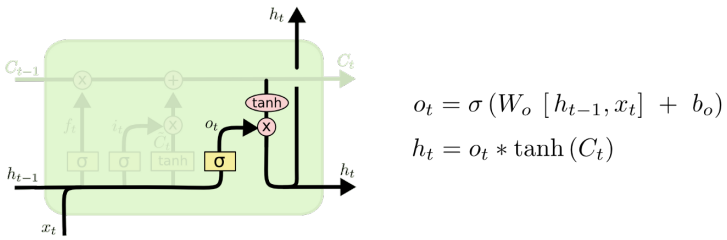


Figure 20: Output gate

4 Methodology

In this chapter, the main topics of the thesis are discussed in a clear, precise and concise way. This includes a description of the dataset, how features are extracted, the model design with its submodels.

4.1 RAVDASS dataset

The database is an open source file which is prepared by Ryerson University and released on April 5, 2018. In this subtopic, it deals with how the database is structured, number actors, annotation means and how it is adapted to our thesis.

4.1.1 RAVDASS structure

The Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS) by Russo, Frank A and Livingstone, Steven [25] is published on April 5, 2018, which contains 7356 files. It is a dynamic, multimodal set of facial and vocal expressions in North American English. The files are structured and categorized as the following structure.

Audio-only files:
Audio-only files of all actors (01-24) are available as two separate zip files (200 MB each):

- Speech file (Audio_Speech_Actors_01-24.zip, 215 MB) contains 1440 files: 60 trials per actor x 24 actors = 1440.
- Song file (Audio_Song_Actors_01-24.zip, 198 MB) contains 1012 files: 44 trials per actor x 23 actors = 1012.

Audio-Visual and Video-only files:
Video files are provided as separate zip downloads for each actor (01-24, 500 MB each), and are split into separate speech and song downloads:

- Speech files (Video_Speech_Actor_01.zip to Video_Speech_Actor_24.zip)

collectively contains 2880 files: 60 trials per actor x 2 modalities (AV, VO) x 24 actors = 2880.

- Song files (Video_Song_Actor_01.zip to Video_Song_Actor_24.zip) collectively contains 2024 files: 44 trials per actor x 2 modalities (AV, VO) x 23 actors = 2024.

The naming of the files are represented as a collection of identifiers :

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

As mentioned above, the name is a combination of eight identifiers each with a different description. To be more clear on the naming part an example is given below.

Filename example: 02-01-06-01-02-01-12.mp4

1. Video-only (02)
2. Speech (01)
3. Fearful (06)
4. Normal intensity (01)
5. Statement "dogs" (02)

6. 1st Repetition (01)
7. 12th Actor (12)
8. Female, as the actor ID number is even

4.1.2 actors

The database contains 24 professional actors (12 females,12 male), vocalizing two lexically-matched statements in a neutral North American accent. Speech includes calm, happiness, sadness, anger, fear, surprise, and disgust expressions, and the song contains calm, happiness, sadness, anger, and fear emotions. Each expression is produced at two levels of emotional intensity (normal, strong) with an additional neutral expression. The actors have expressed the different acted emotions in a, particularly designed lab studio.

4.1.3 Ground truth annotations

Each file has been rated ten times on emotional validity, intensity, and genuineness. Ratings were provided by 247 individuals who were characteristic of untrained adult research participants from North America. A further set of 72 participants provided test-retest data. High levels of emotional validity, interrater reliability, and test-retest intrarater reliability were reported.

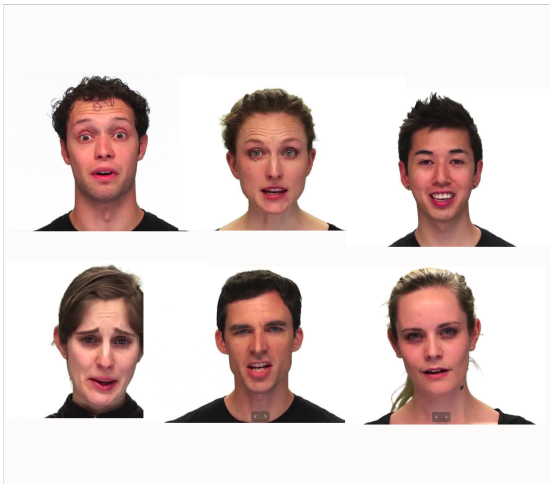


Figure 21: Selected samples from RAVDESS database

4.1.4 Adaptation of the dataset

For this paper, we disregard the song part and took only speech part of the video part with only seven emotions which are angry, disgust, fear, happiness, neutral, sadness and surprise. Selected sample images from audio-video database is shown in figure 21.

4.2 Model feature Extraction

Feature extraction involves reducing the number of resources required to describe a large set of data. In other words, Feature extraction is a process of dimensionality reduction by which an initial set of raw data is reduced to more manageable groups for processing. A characteristic of these large data sets is a large number of variables that require a lot of computing resources to process. Feature extraction is the name for methods that select and /or combine variables into features, effectively reducing the amount of data that must be processed, while still accurately and completely describing the original data set. Feature extraction in two-dimensional space like images is very crucial because it requires a large amount of memory and high computational power during the training of a complex model that needs a huge amount of data for learning. Hence, below is a description of what and how feature extraction is used for each of the three classifiers used in our paper independently.

4.2.1 Audio feature extraction

The audio part of the dataset is provided as a collection of a single clip of audio with the length between 3 and 5 minutes. Each audio clip is fed into a preprocessing function that accepts a raw audio file and is converted into a two-dimensional vector (496 by 64). This preprocessing function is a function which is similarly used in training a model on an audio set called VGGish. The preprocessing function outputs a single vector with the size of 496 by 64 for an audio file which has a length of at least 5 seconds. If the length of an audio clip in the dataset is less than five seconds, the clip is first padded with zeros to fill out the length before the function is invoked. This task is applied repetitively until every audio dataset in the database is converted into a two-dimensional array.

The preprocessing module of generating a spectrogram works 25ms window and 10ms shift. To generate one example or sample from the audio, the module

expects an audio clip with a length of 4960 and processes it with zero overlappings to generate more examples or samples if the clip is long enough. The feature extracted of the spectrogram is 64.

After processing the 4960ms audio data we get 496 x 64-dimensional vector. To extract the features from this vector we used a pre-trained model called VGGish. VGGish is a pre-trained model trained on audio set data which is released by Google. The audio set is a large amount of data collected from Youtube based on the tags provided when people upload videos into youtube. The audio set has around 6000hrs of audio data for 567 audio classes. Google has released its model which is trained on audioset dataset. The model is VGG like CNN architectures which operate on spectrograms at lower layers and uses many convolution layers and fully connected layers as we go upper layers. This allows us to extract representations from higher layers for our own audio data. As a result, we forward pass that reshaped vector into VGGish and obtain a higher dimensional embedding of dimension 512. This is repeated for all the available training data as well as testing data sets.

4.2.2 Visual feature extraction

From the RAVDESS database, the visual (video) part is considered to extract the visual features. Frist the sequence of frames are generated for each video clip using a tool called FFmpeg at 45 fps. For each frame, a face detection techniques are applied to crop the face from the other parts of the frame. Then frames are resized followed by colour converting, reshaping and normalization.

After having these normalized and reshaped data frames, they are fed into a pre-trained CNN model called VGG16 for extracting an important pattern that exactly describes the original frames. VGG16 is a convolutional neural network model proposed for Very Deep Convolutional Networks for Large-Scale Image classification. VGG16 is trained on Imagenet which is a dataset of over 15 million labelled high-resolution images belonging to roughly 22,000 categories. The images were collected from the web and labelled by human labellers using Amazon's Mechanical Turk crowd-sourcing tool. VGG16 was trained for weeks and was using NVIDIA Titan Black GPU. This model outputs a set of embeddings with the size of 512 features for each frame.

4.2.3 Stacked feature extraction

The features used to train the third classifier (final predictor) of the emotion recognizer are obtained from the two base classifier of the whole model architecture. The two base classifier one for audio and the second for visual are trained independently from a prepared dataset for training. Then the raw data which is reserved for training the stacked classifier is given into corresponding base classifiers and outputs their prediction. Then from each of the base classifiers, we take the second last layer output of the model. Lastly, the two extracted outputs are merged to form a new feature in a synchronous way. This process is repeated for all the training data used for the stacked classifier.

4.2.4 Why Transfer learning for Feature selection

As it is apparent that Transfer learning is a machine learning method where a model developed for a task is reused as the starting point for a model on a second task. In our case, we use it as a feature extractor in both audio and video part. So let us dive into briefing why transfer learning is chosen other than the classical method of machine learning both audio and visual.

For visual part:
The classical feature extraction methods for a given use-case are designed by the handcrafted way by the programmer. Generally, even though this is a time-consuming process for application, they have more performance issue problem depending on the particular system applied. For example, when we use them in an application that extracts features from the image they are quite influenced by uneven illumination, size, colour, and orientation of the image. Although there are no strong background theories on why their performance is low compared to deep learning algorithms, the assumption behind this is that they are static and does not learn from the data for the feature extraction part. In another side, deep learning algorithms such as auto coders and convolution neural network (CNN) are able to learn from the input data. These deep learning algorithms are more powerful spatial feature extraction, particularly for images.

Furthermore, in order to use a good feature extractor algorithm other than deep learning we have to make sure that certain conditions such as, we have to have domain knowledge on the particular application, input's Time variant invariant, number of features to be used, (variation) of the data are captured efficiently and systematically.

From the classical machine learning algorithms that are used as image feature extraction are basically categorized into Spatial Gray level difference method (SGLDM), Local binary pattern (LBP) Run difference method(RDM) and the most common for audio is MFCC. These algorithms use a statistical approach to extract the values of the parameters from the given image. The parameters are selected based on the specific application like contrast, energy, homogeneity, sharpness et al. In general they choose a few parameter sets which is more and more low dimensionality compared to the curse dimensionality of the original input. So they lack the appropriate selection of parameters as well as they are prone to approximation errors. Even though such kind of handcrafted feature extraction is computationally fast during training, they are far away in terms of accuracy from modern algorithms such as deep learning.

In addition, article [30] did a comparison among the following four main algorithms from different learning methodology analysis.

Colour-texture based
In these methods, image is characterized(perceived) by three features: colour, textual and shape.

SIFT
This is mainly designed for applications that are more affected by orientation, location and scale variance.so here first the SIFT descriptors should be extracted from the image.

HMAX
Here the algorithm is based on visual information extraction method of our brain and has five steps for feature extraction.

CNN
Based on local receptive fields, shared weights and spatial or temporal supersampling to ensure scale, shit and orientation distortions.

Hence, the selection of the above algorithms depends on the number of training data and image category. Color-texture performs well for a small dataset and for an application that uses colour-related classification. In SIFT it performs great in local regions that are based on the shape and orientation .the other two uses some learning means. The main difference between HMAX and CNN is the HMAX learns the patches randomly for uniform distribution purpose. Whereas CNN learns patches based on the most informative one by iterating on the training data.

Therefore, the reason why deep learning (CNN) is chosen for feature extractor for this work is due to the following reasons: Best in class performance in image input, scale effectively with data, no need feature engineering, adaptable and transferable, and best fits for inputs with high dimensionality input i.e like image and strong enough on discriminating spectrum temporal patterns. This means that they are able to capture patterns across time and frequency for giving input spectrograms.

For the audio part:
There are so many methods for feature extraction from audio files that are used for describing an audio clip in a reduced dimension. Numerous research studies have been conducted on audio feature extraction due to the importance of audio features in characterizing the emotion of videos. Typical audio features include zero-crossing rate, sound energy, pitch, onset, beat strength, tempo, bandwidth, spectral centroid, spectral roll-off, spectral flux, loudness, and Mel-frequency Cepstrum Coefficients (MFCC) et al.

Other new methodologies are emerging in recent years to extract features from the audio which perform better performance in terms of describing the audio to be used in machine learning (or deep learning) techniques. One of these techniques is using a pre-trained convolutional neural network(CNN) trained on a large scale audio dataset. These deep learning algorithms preprocess the audio according to their window and shift parameters settings to convert into two-dimensional space vector just like an image and usually, it is called spectrogram. From the available pre-trained models that are trained on, we used a recently released model by Google research group in 2017 called VGGish.

The reason to choose a pre-trained model than using the classical features extraction methods is that it outperforms in extracting important features compared to the classical ones. According to [8] CNN based feature extraction achieves better results in performance compared to the conventional feature-based approach. Moreover, [13] states that training a classifier starting from a similar pre-trained model outperforms than training the network from scratch. To confirm these two statements are valid in our case, we have trained the classifier with MFCC features of the audio files. The result of the classifier underperforms as of using VGGish features. As a result, using a pre-trained audio model for audio feature extraction is a convenient option for performance as well as resource efficiency.

4.3 Model structure

4.3.1 Model overview

The emotion recognition from the video has two main modules namely audio(acoustic) data analysis and visual data analysis. The general architecture of the proposed solution is shown in figure 4.2. The modules are trained and validated using the standard database RAVDESS [25]. The paper performs a stacked generalization method, which requires the processing of the classification results from both the visual and video module modalities. The second last layer outputs of audio and visual classifiers are used as an input to the new brand stacked network final emotion predictor. Next, we explain how features are extracted for both audio and visual classifiers followed by a brief description of how classifier layer outputs are combined and which networks are considered for each model. Figure 22 shows the general proposed architecture.

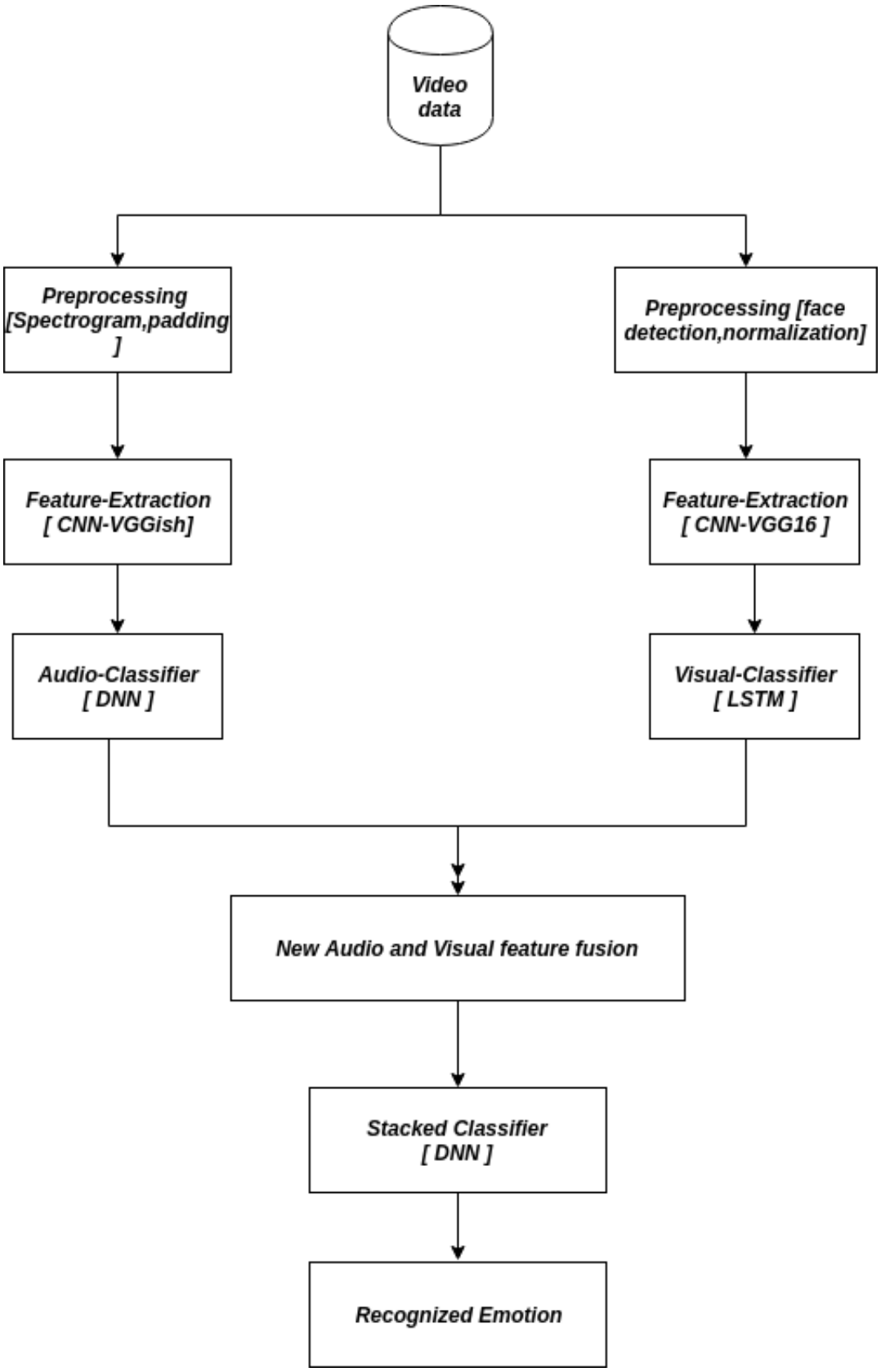


Figure 22: Proposed Architecture for Emotion Recognition

4.3.2 Audio model

Audio files can express different emotions based on the values they contain to the audio parameter measures such as speech rate, pitch frequency, high energy, intensity duration, spectral energy distribution, MFCC and filter-bank energy parameters. In our case, we choose our approach on spectrogram representation form of the audio since they are independent of the lexical meaning of the spoken words. Each audio clip which represents a single emotion expression is given to the pre-processes algorithm used by VGGish developed by Hershey, Shawn and Chaudhuri, Sourish and Ellis, Daniel PW and Gemmeke, Jort F and Jansen, Aren and Moore, R Channing and Plakal, Manoj and Platt, Devin and Saurous, Rif A and Seybold, Bryan and others [13] pre-trained model. The VGGish pre-processing algorithm generate spectrogram for every 4960ms audio data. The spectrogram works with 25ms window with 10ms shift. The feature dimension of the spectrogram is 64. After processing 4960ms audio data we get 496x64 spectrogram(image). To fit our audio clips dataset, which are almost all less than 4960ms, for each clip, we padded at the end of the clip with zero before passing to the audio preprocessing algorithm.

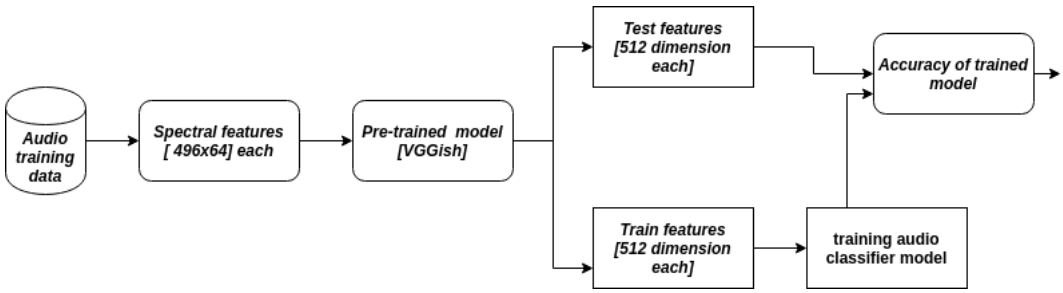


Figure 23: Audio module architecture

After the above preprocessing algorithm is done for all of the available audio datasets, we forward pass these preprocessed audio data into pre-trained CNN called VGGish trained on an audionet created by Gemmeke, Jort F and Ellis, Daniel PW and Freedman, Dylan and Jansen, Aren and Lawrence, Wade and Moore, R Channing and Plakal, Manoj and Ritter, Marvin [11] and obtain a higher dimensional embedding of dimension 512. Once we extract the features for all the training data we used this ready data for training our audio model classifier. The audio classifier is a simple two-layer Deep Neural Network(DNN) with 256 neurons each followed by a softmax classifier layer. The complete pipeline for the whole audio module architecture and an audio classifier model are shown in Figure 23 and Figure 24 respectively.

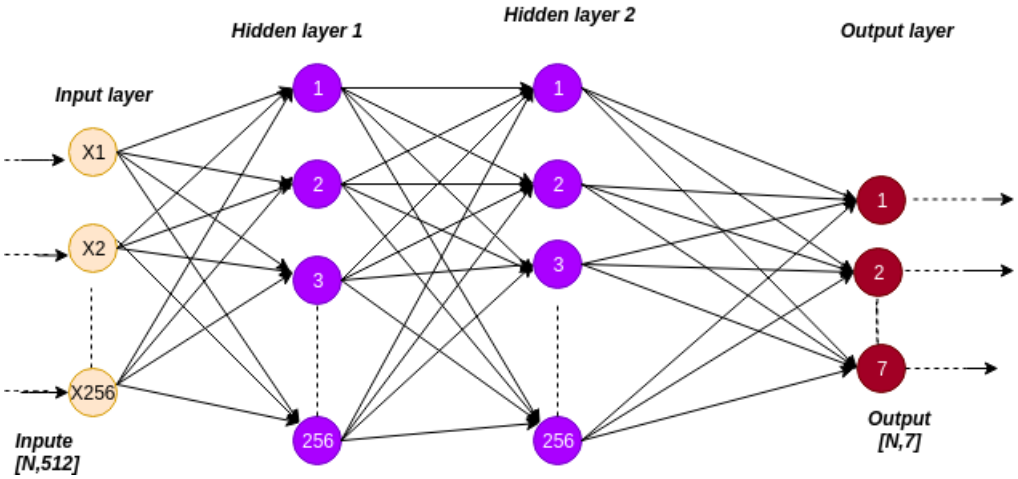


Figure 24: Audio model classifier.

4.3.3 Visual Model

Apparently using different visual human body expressions such as face and gesture gives a certain degree of emotion. From these body parts facial expressions provide the most informative data for computer awareness of emotions. Therefore, in this paper, we consider the facial expression as our visual data for recognizing emotions. From the video part of the dataset, we extracted a varying number of frames for each clip at a rate of 45 fps. Then we applied a face detection algorithm called Haar Cascade[site] to crop the frames. this step is followed by resizing, reshape the face image to which the pre-trained model is built-in and finally normalization. Here since we want our model to learn a pattern of how the frames are changing with time series for each clip, the frames are arranged in sequence for a sample clip of the visual dataset. This helps to learn the network how frames of clips are varying for the different emotions expressed by the 24 professional actors.

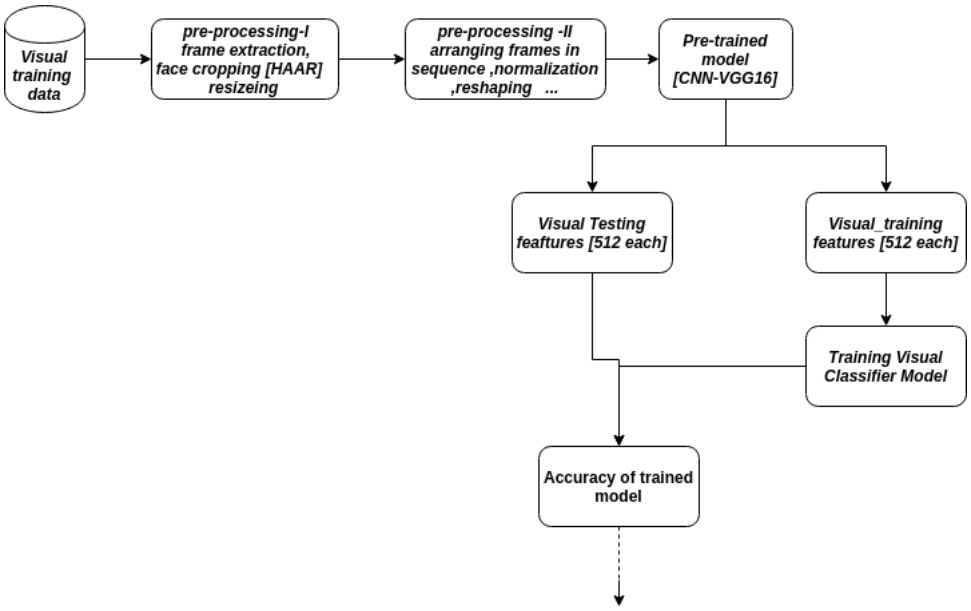


Figure 25: Visual Module Architecture

Having the pre-processed data, we used a CNN pre-trained model called VGG16 developed by Krizhevsky, Alex and Sutskever, Ilya and Hinton, Geoffrey E [16] trained on Imagenet as feature extractor through transfer learning. The pre-trained model gives 512 features for a single frame. So in total, we have the number of frames generated by 512 dimensions for each sample of the visual dataset. Then we used a recurrent neural network particularly LSTM as our visual classifier. The LSTM is a simple 2 layers with 200 and 100 units each respectively followed by softmax classifier. The full description in the graph for both the visual module pipeline and visual classifier is shown in figure 26 and figure 26 respectively.

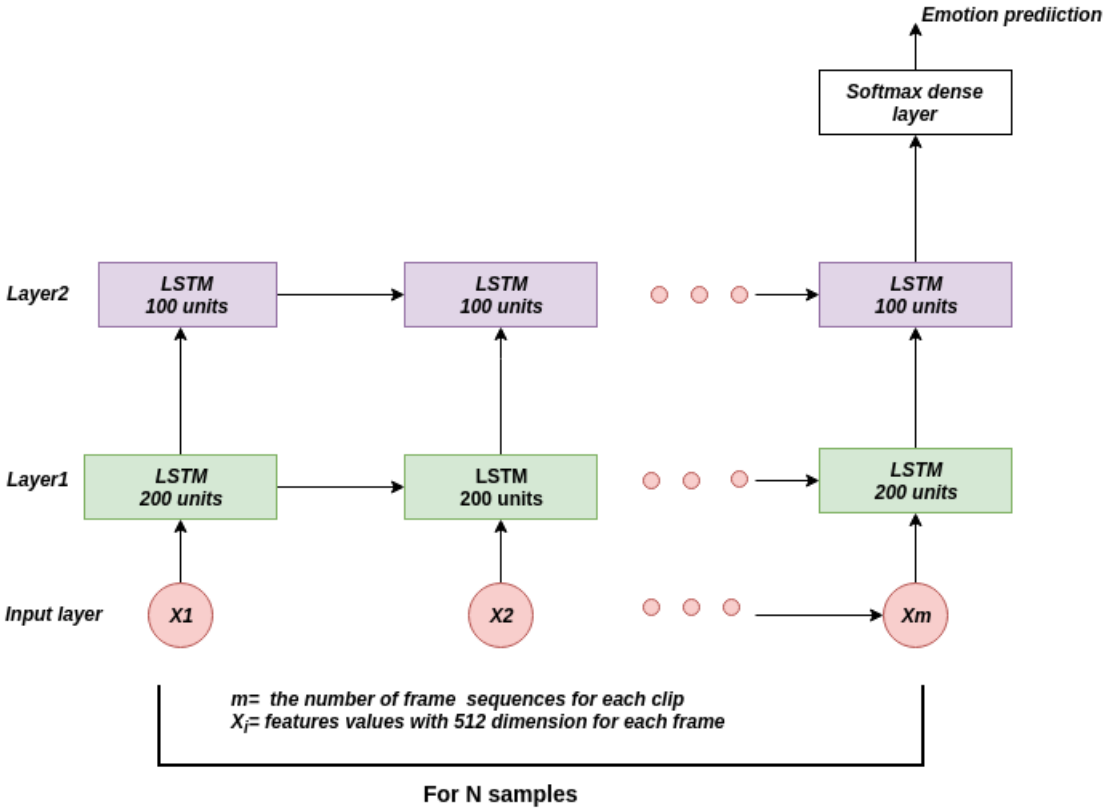


Figure 26: Visual Classifier Model

4.3.4 Stacked model

This is a model trained for the final classification task. This new brand model network is used because recent trends in machine learning, particularly in deep learning are showing better performance in predicting accuracy with another stacked model(level 1) as compared to the individual models(level 0).

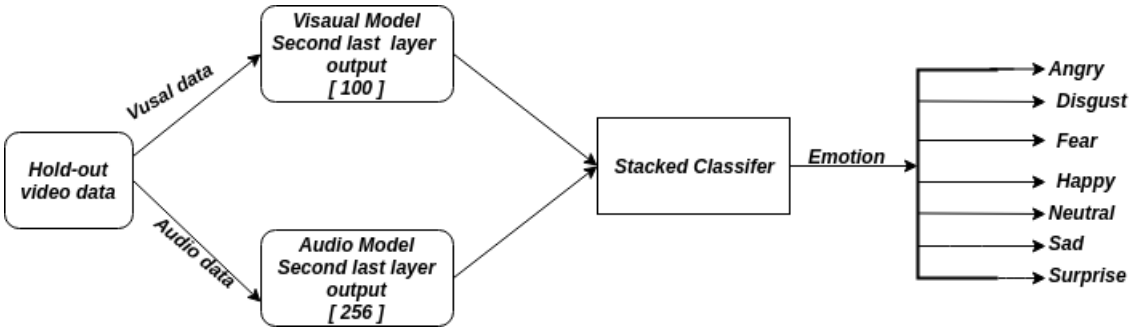


Figure 27: Stacked Classifier Model

The model is trained with a separate hold out data that has never been seen before for training of the individual models: audio classifier and visual classifier. The features used to train this network are obtained by merging the outputs of the second last layer of the audio and visual classifier. The audio model extracts 256 dimensions and similarly, 100 dimensions are extracted from the visual classifier which their summation gives 356 dimensions for a single sample. This process is repeated for all of the training data provided to train this network. The network size for this network is a single DNN with 256 neurons annexed with a softmax layer. The layout of this model and VGGish and VGG16 architecture are depicted in figure 27, 28 and 29 respectively.

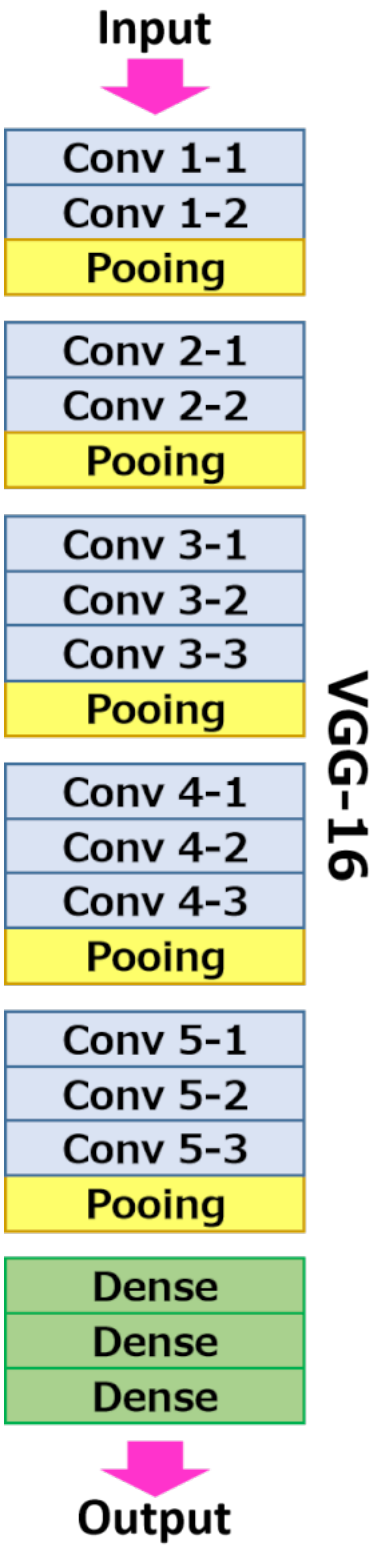


Figure 28: VGG16 Architecture

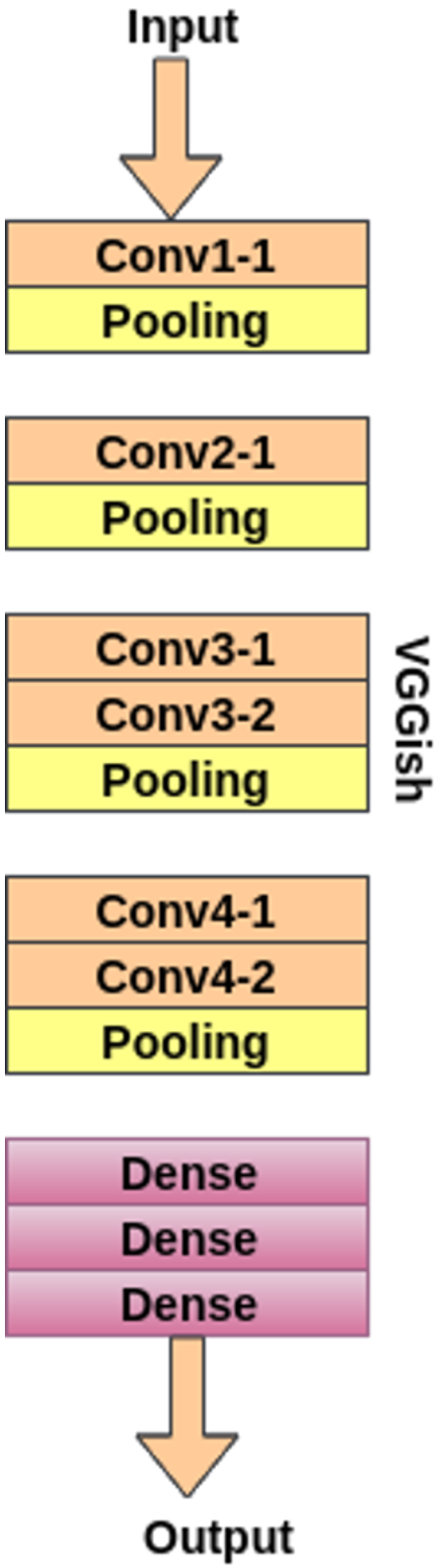


Figure 29: VGGish Architecture

5 Model Implementation

5.1 Software used

Software is a set of programs or instructions and other operating information that has been built in a single bundle to be used by a digital device (computer) for accomplishing a specific task. The software used for the purpose of this thesis development is something that includes the programming language used, tools and packages installed. These are explained one by one for having clear understanding of what and for what they are used.

5.1.1 Programming language

A programming language is a formal language, which comprises a set of instructions that produce various kinds of output. Programming languages are used in computer programming to implement algorithms. The sole programming language used in this emotion recognition system is Python3.

Python programming language:
Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently whereas other languages use punctuation, and it has fewer syntactic constructions than other languages. The main properties of python are:

- **Python is Interpreted**
  Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- **Python is Interactive**
  we can actually sit at a Python prompt and interact with the interpreter directly to write your programs.
- **Python is Object-Oriented**
  Python supports Object-Oriented style or technique of programming that encapsulates code within objects.
- **Python is a Beginner's Language**
  Python is a great language for the beginner-level programmers and supports

the development of a wide range of applications from simple text processing to WWW browsers to games.

Python is a rich language with many features like easy to learn, easy to read , easy to maintain, a broad standard library, interactive mode, portable, extendable, databases and scalable. Apart from the above-mentioned features, Python has a big list of good features, few are listed below :

- It supports functional and structured programming methods as well as OOP.
- It can be used as a scripting language or can be compiled to byte-code for building large applications.
- It provides very high-level dynamic data types and supports dynamic type checking.
- It supports automatic garbage collection.
- It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

5.1.2 Tools and packages

The tools and packages used are listed and discussed in brief for what purpose they are used.

FFmpeg:
FFmpeg is very fast audio and video converter that also grab from a live audio/video source . It can also convert between arbitrary sample rates and resize video on the fly with a high-quality polyphase filter. Moreover, it is a free and open-source project consisting of a vast software suite of libraries and programs for handling video, audio, and other multimedia files and streams. At its core is the FFmpeg program itself, designed for command-line-based processing of video and audio files, and widely used for format transcoding. In our case, we used it to convert the visual part of the dataset into a sequence of 45 frames per second. The tool is used until all of the dataset clips are converted to their corresponding sequence of frames. Some common applications that are supported by FFmpeg are mentioned as follows:

Extraction frames from video:
FFmpeg helps for extraction a series of frames from a given video automatically in the form of image formats such as jpg, jpeg and others. The tool extracts the images frame by frame at a given rate of frames per second or it takes the default rate of the video. The general syntax of the FFmpeg command is given by:

FFmpeg -i input_video output_name.jpg/jpeg

Cutting Video/audio :
FFmpeg can cut a video or audio for a certain interval. The syntax command used for cutting is :

FFmpeg -i input_file -to -ss starting_time -t ending_time copy output_file

This command cuts an audio or video file from the given starting time to the ending time and place it in a file named output_file. Other common applications and functions of FFmpeg like concatenating audios or videos, knowing properties of audio or video, and creating video have similar syntax and can be done using their particular command.

HAAR CASCADE:
Haar Cascade is a machine learning object detection algorithm used to identify objects in an image or video and based on the concept of features proposed by Paul Viola and Michael Jones in their paper "Rapid Object Detection using a Boosted Cascade of Simple Features" in 2001. It is a machine learning-based approach where a cascade function is trained from a lot of positive and negative images. It is then used to detect objects in other images.

Haar cascade is well known for being able to detect faces and body parts in an image, but can be trained to identify almost any object. It is used here as a face detector followed by cropping of a frame until all of the frames are cropped. The sample code used in our development looks like :

```
face_detector = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
for filename in glob.glob(path + '/' + folder +'/'+ subfolder +'/*.jpg'):
    image = cv2.imread(filename)
    img = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
```

```
face_view = face_detector.detectMultiScale(img,1.3,5)
for (x,y,w,h) in face_view:
    face_img = img[y:y+h, x:x+w]
    cv2.imwrite(filename,face_img)
    break
```

Jupyter Notebook:
The Jupyter Notebook is an open-source web application that allows us to create and share documents that contain live code, equations, visualizations and narrative text. Some of its functions include data cleaning and transformation, numerical simulation, statistical modelling, data visualization, machine learning, and much more. We use this tool as our working environment like integrated development environment (IDE).

Scikit learn:
Scikit is a free machine learning library for Python. It features various algorithms like support vector machine, random forests, and k-neighbours, and it also supports Python numerical and scientific libraries like NumPy and SciPy. It was created to make doing machine learning in Python easier and more robust. This module has rich functionality like pre-processing, splitting data into training and testing, result visualization and evaluation tools and more. All these functionalities of scikit are used during the development of our system.

Pandas:
Pandas is a popular Python package for data science and it offers powerful, expressive and flexible data structures that make data manipulation and analysis easy, among many other things. The DataFrame is one of these structures. The DataFrame consists of three main components: the data, the index, and the columns. It is used to read data which are in different formats like csv, xlsx and others and store in the format of DataFrame.

TensorFlow:
TensorFlow is created by the Google Brain team and is an open-source library for numerical computation and large-scale machine learning. TensorFlow bundles together a slew of machine learning and deep learning (aka neural networking) models and algorithms and makes them useful by way of a common metaphor. It uses Python to provide a convenient front-end API for building applications with the framework while executing those applications in high-performance C++.

TensorFlow can train and run deep neural networks for handwritten digit classification, image recognition, word embeddings, recurrent neural networks, sequence-to-sequence models for machine translation, natural language processing, and PDE (partial differential equation) based simulations. Best of all, TensorFlow supports production prediction at scale, with the same models used for training. Our application is developed and tested using TensorFlow as a backend.

Keras:
Keras is an open-source neural network library written in Python. It is capable of running on top of TensorFlow, Microsoft Cognitive Toolkit, Theano, CNTK or PlaidML. It is designed to enable fast experimentation with deep neural networks and focuses on being user-friendly, modular, and extensible. Keras was conceived to be an interface rather than a standalone machine learning framework. It offers a higher-level, more intuitive set of abstractions that make it easy to develop deep learning models regardless of the computational backend used.

Keras Applications are deep learning models that are made available alongside pre-trained weights. These models can be used for prediction, feature extraction, and fine-tuning. Models that are available for image classification with weight trained on ImageNet are : Xception ,VGG16, VGG19, ResNet, ResNetV2, ResNeXt, InceptionV3 ,MobileNet , MobileNetV2, DensNet and more. Moreover, there is a pre-trained model trained on audio set called VGGish released by a google research group in 2017. From these pre-trained models, we used VGG16 and VGGish. Both VGG16 and GGish are used for the sole purpose of visual and audio feature extractions respectively.

5.2 Training modules

Training in machine learning is fitting the dataset to learn for performing a specific task like classification, regression, clustering and others. It involves providing the machine learning algorithm with training data to learn from. the training data should contain the correct answer, which is known as a target or target attribute. The learning algorithm finds patterns in the training data that map the input data attributes to the target, which is the answer that we want to predict, and it outputs a machine learning model that captures these patterns. In our case, the machine learning model (a deep neural network model) is designed to classify an expressed emotion from the data into the seven most common emotion category, which are anger, disgust, fear, happiness, neutral, sadness, surprise.

Before fitting our model pre-fitting steps are done for appropriate and good procedure. These steps include exploring data, cleaning data, choosing the best feature extraction, and others. Exploring data is finding a video database that aligns and can be customized with our use-case. Similarly, cleaning data is preparing the data to be used in a way that is acceptable by the shape of the feature extractor and classifier model. Then, the data is split into three sections: training one(50%) for the base models, training two(25%) for the stacker model and testing (25%) for evaluation and testing. The base models gets extracted features from pre trained VGG16 whereas the stacked classifier receives from the trained based classifier in a combined way.

After the dataset are cleared and features are extracted for each of the models, the model can be trained based on its settings. The setting is different from one model to another model because of the difference in configuration during building the model. the configurations can be on the number of layers, a number of epochs, batch size, amount data needed for training, how data is split into training and testing, types of the optimizer, type of loss function, learning rate used , type of performance measurement used, and others. These configurations have a great effect on building a successful and robust neural network model. The setting variations mentioned above and others can be grouped into hyper-parameters, optimizers and loss functions. In the subtopics below, we discussed what hyper-parameters, optimizer and loss functions are used with detailed description for training the emotion classifier model.

5.2.1 Hyper-parameters

In machine learning, hyperparameters express high-level structural settings for algorithms, which are decided before fitting the model because they can not be learned from the data. Different model training algorithms require different hyperparameters, but there is no need for requirements for simple model algorithms. Having these hyperparameters, the training algorithm learns the model parameters from the data. The model hyperparameters used in our model are described below.

Hidden layers:
In a deep neural network, hidden layer is a layer, which is in between input layers and output layers. The hidden layers are set up with a default weighted random input. For the base classifiers, we used more than two hidden layers whereas for the stacked classifier it has one hidden layer.

Hidden units:
Hidden units are units, which are found in the hidden layers and receives information, process it according to a given activation function, and finally, send it to the next units. We have used different numbers of hidden units between 100 and 500 units at different hidden layers.

Activation function: In a neural network, activation is a mathematical formalism that is used to decide the approximated output of the neuron. There are many activation functions available that can be used according to the application output preference such as step function, sign function, linear function, sigmoid function, tanh function, Relu function and softmax function. Our models use ReLu function for the non-output layers and softmax function for the output layer. These two functions are described below.

ReLu function:
It is a function which outputs X if X is positive and zeroes otherwise. Mathematically

$$\mathrm{F(x) = max(0,x)}$$

Softmax function:
Softmax functions is a function that takes as input a vector of K real numbers and normalizes it into a probability distribution consisting of K probabilities. Mathematically,

$$F(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \quad for \quad i = 1, ..., K \quad and \quad z = z_1, ..., z_K$$

5.2.2 Loss function

Machine learning models learn by means of the loss function. It is a method of evaluating how much a given specific machine learning algorithm models the given actual data. Loss function would lead to a very large number of predictions deviates too much from the actual ground truth. Gradually, with the help of some optimization function, loss function learns to reduce the error in prediction.

There is no universal ground rule that states one-size-fits all loss functions to algorithms in machine learning. there are different factors involved in choosing the right loss function for a specific problem such as type of machining learn-

ing algorithm chosen, computation and distribution of the dataset. These loss functions can broadly be grouped into regression loss function and classification loss functions based on the type of task the algorithm is dealing with. Since our model task is to classify a given emotion into one of the common seven emotion categories, our loss function is chosen from the available loss function that is used for multi-class classification purpose. We used categorical cross-entropy as our loss function. The description of how categorical cross-entropy works is explained below.

Categorical-cross entropy: This type of loss function is also called a negative log-likelihood. It is one of the most common loss functions for categorization problems. It measures the similarity between two probability distributions, typically the true labels and the predicted labels. It is given by L = -sum(y * log(y_prediction)) where y is the probability distribution of true labels (typically a one-hot vector) and y_prediction is the probability distribution of the predicted labels, often coming from a softmax. The network trains to output a probability over the number of classes(C) for each input. To do this it incorporates the softmax activation and cross-entropy loss. This step is shown in Figure 30.

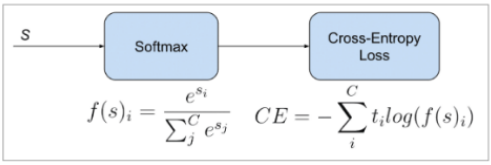


Figure 30: categorical cross-entropy loss.

5.2.3 Optimizer

Optimization algorithms are capable of minimizing or maximizing an objective function which is simply a mathematical function dependent on the model's internal learnable parametre. The learnable parameters are used in computing the target values(Y) from a set of predictors(X) used in the models. The internal parameters of a Model play a very important role in efficiently and effectively training a Model and produce accurate results. we can use various Optimization strategies and algorithms to update and calculate appropriate and optimum values of such model's parameters which influence our model's learning process and the output of a model. There are already many deep neural models optimizers available for use. From these optimizers, gradient descent is the most popular

optimizer in deep neural models. We used in our model an optimized variation of gradient descent optimizer called Adam. A brief description of Adamoptimizer is described below.

Adam:
Adam is one of the most used types of optimizers in the field of deep learning models, which stands for Adaptive Moment Estimation. It is a method that computes adaptive learning rates for each parameter. In addition to storing an exponentially decaying average of past squared gradients like AdaDelta, Adam also keeps an exponentially decaying average of past gradients. There are two functions called M(t) and V(t) which are used for the final formula of Adam optimizer. M(t) and V(t) are the values of the first moment which is the Mean and the second moment which is the uncentered variance of the gradients respectively. The formulas of these and final formula for the Adam optimizer are described below.

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t}$$

$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t}$$

The formulas for the first moment(mean) and the second moment(variance) of the Gradients. Then the final formula for the parameter update is :

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon}$$

Adam works well in practice and compares favourably to other adaptive learning-method algorithms as it converges very fast and the learning speed of the model is quite fast and efficient. It also rectifies every problem that is faced in other optimization techniques such as vanishing learning rate, slow convergence or high variance in the parameter updates which leads to fluctuating loss function.The sample code in keras to use this optimizer with default parameters is:

from keras.optimizers import Adamax adam = Adamax()

5.3 Testing modules

Testing a learned model in machine learning, particularly in a deep neural network is all about measuring the performance of the model by providing a similar unseen data using different matrices so as to know how much the model is learnt. We have used 25% of our dataset to test both our base classifiers and stacked (final) classifier. We used a classification matrix, which isdescribe one by one below. From these, we particularly consider the accuracy of the model as our performance measure.

Confusion matrix: In classification, a confusion matrix is a specific table layout that allows visualization of the performance of an algorithm, particularly in supervised learning. The size of the confusion is N x N where N is the number of category labels. The content of the matrix is filled with the four types of outcome called true positive, true negative, false positive and false negative for each class.

True positive:
Predicting an observation belongs to a class and it actually does belong to that class

True Negative:
Predicting an observation does not belong to a class and it actually does not belong to that class.

False Positive:
Predicting an observation belongs to a class when in reality it does not.

False Negative:
Predicting an observation does not belong to a class when in fact it does.

```
audiomodel.summary()
```

Layer (type)	Output Shape	Param #
dense_4 (Dense)	(None, 512)	262656
dropout_3 (Dropout)	(None, 512)	0
dense_5 (Dense)	(None, 256)	131328
dropout_4 (Dropout)	(None, 256)	0
dense_6 (Dense)	(None, 7)	1799

Total params: 395,783
Trainable params: 395,783
Non-trainable params: 0

Table 1: audio classifier model summary

Accuracy:
It is the percentage of correct predictions for the test data. It can be calculated easily by dividing the number of correct predictions by the number of total predictions.

$$\text{Accuray} = \text{correct predictions} / \text{Total predictions}$$

The sample code used for displaying the confusion matrix on a test dataset is:

#confusion matrix to see which class is detected well and the other around

$$\begin{aligned}
&\mathrm{Y}_{p}red = newmodel.predict(X_{t}estl)\\
&y_{p}red = np.argmax(Y_{p}red, axis = 1)\\
&y_{t}est = [np.where(r == 1)[0][0] for r in y_{t}estl]\\
&print('ConfusionMatrix')\\
&print("")\\
&print(confusion_{m}atrix(y_{t}est, y_{p}red)))\\
&print('ClassificationReport')\\
&target_{n}ames = ['angery','Disguste','Fear','Happy','Neutral','Sad','Surprise']\\
&print(classification_{r}eport(y_{t}est, y_{p}red, target_{n}ames = target_{n}ames))
\end{aligned}$$

and table 1 shows the audio classifier network summary.

6 Results and Discussion

In this section, the performance of the whole model proposed, the performance of the audio classifier, the performance of the visual classifier, performance of the stacked(final) classifier are discussed on both training data as well as testing data. We first present the results considering the independent individual classifiers models followed by the merged stacked classifier. For each experiment, we also comment on the experiment setup and results obtained in comparison to different configurations of the models. The summary of the results and related performance indexes are shown in the form of confusion matrix and graph. All results are based on the RAVDESS database. The database contains audio-visual clips with 24 professional actors saying a given sentence in English. In the database, there are eight emotion expressions but in this paper, seven basic emotions are considered: anger, disgust, fear, happiness, neutral, sadness and surprise. So let's dive to the individual models.

6.1 Audio Model results

The audio model is trained and tested on the audio dataset from RAVDESS database. Fifty percent (50%) and twenty five percent (25%) of the dataset are used for training and testing the model respectively. the rest twenty five percent (25%) were used for training the combined audio and video final classifier. The clips expressing emotions have different variations. The professional actors express the acted emotion with and without intensities for each emotion class except for the neutral class. Therefore, each class has the same number of clips and is balanced apart from the neural class which is halved due to the absence of intensity expression.

The experiment for the audio model is carried out with a system with the Intel Core i7, processor 2.5 GigaHertz, with 8 Gigabytes RAM running Linux 16.04. The model is implemented in jupyter notebook on python 3.6 programming language. Python is chosen for implementation because of its rich wide range of packages in audio and image processing. In the subsection below, the performance in terms of accuracy is discussed in detail for both training and testing dataset.

6.1.1 Accuracy on training

A neural network classifier is trained with an audio dataset based on the setting of the network. The network is trained for a given epoch included with callbacks. the callbacks are EarlyStoping and ModelCheckpoint. EarlyStopping is a Keras library, which is useful to monitor the performance of the model while training. This method prevents the model not to overfit by stopping the training. 'val_loss' parameter of the EarlyStoping function is used as a monitoring mechanism for stopping the model from further training. Moreover, ModelCheckpoint is also a Keras library which checks the performance of the model at each epoch. It reads the performance of the current epoch of the model and compares it with the previous best performance value of the model. If the performance value of the current epoch value is greater than the previous best epoch performance value, it updates the performance with the new value otherwise it keeps the previous state. The performance of the neural network model on the training dataset for each of the classes, after it has been trained using this type of configuration, is shown in a bar chart in figure 31.

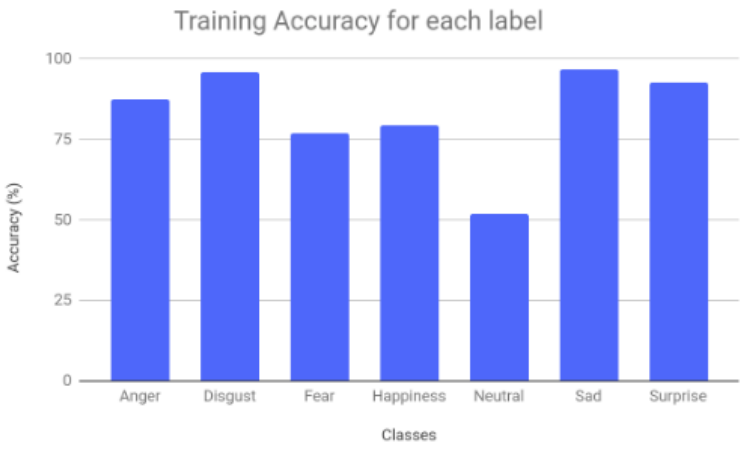


Figure 31: Audio classifier on the training dataset

As it is depicted in figure 31, the audio classifier model is more capable of distinguishing the emotion expressions like anger, disgust, sadness and surprise. The model also learns well for fear and happiness emotion expressions with accuracy above seventy five percent (75%). On the contrary, the model is having difficulty in recognizing for the neutral emotion expressions. This may be because the expression used to express neutral by the acting actors has not the intensity and also it has a smaller number of samples than the other labels. Generally, the overall training accuracy of the audio classifier model, with so many hyperparameter tuning, is 85%.

6.1.2 Accuracy on testing

As mentioned earlier, to evaluate the accuracy of the audio classifier model on the test dataset of RAVDESS audio database, the data is split into three sections: training, testing and a holdout data for the third stacked classifier with 50%, 25% and 25% respectively. The result of the audio classifier is 58%. This result is the best-obtained accuracy for the model in comparison to other parameter and architecture design we have applied. The performance of the model on individual emotions is shown in figure 32.

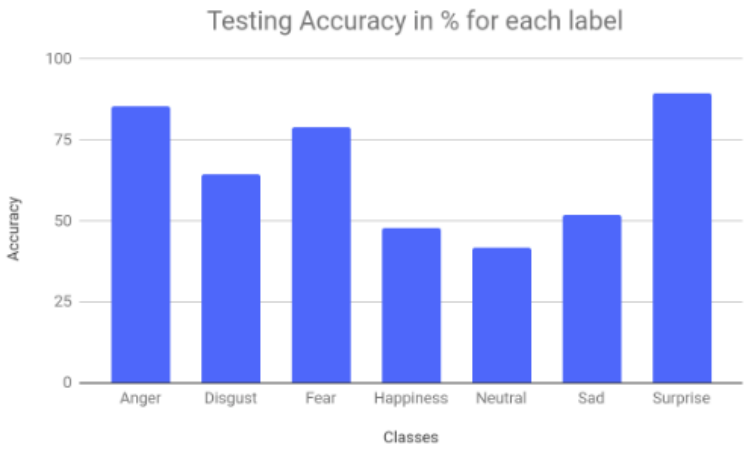


Figure 32: Audio Classifier on Testing data

As shown in Figure 32, the audio performance on the testing data performance well for emotions anger, disgust, fear and surprise. But its performance against the emotions happiness, neutral and sad is quite weak.

6.2 Visual model results

In order to show the visual emotion recognition results, similar to the audio data schema the visual is also separated into the three sections.The visual model is trained and tested on the visual dataset from RAVDESS database. Fifty percent (50%) and twenty five percent (25%) of the dataset are used for training and testing the model respectively. The rest twenty five percent (25%) were used for training the combined audio and video final classifier. Likewise audio clips, visual clips have also different number of clips for each visual emotion. The emotions anger, disgust, fear, happiness, sadness, and surprise have the same number of samples. Whereas the number of examples for the neutral emotion is half of the examples of the other emotions due to the absence of clips that express emotion with intensity.

The experiment for the visual model is carried out with a system that has a GPU processor. GPU is used for fast processing and reducing memory latency. Similar to the audio classifier, the model is implemented in jupyter notebook on python 3.6 programming language. The performance during training and testing with brief descriptions are described in the subsection below.

6.2.1 Accuracy on training

LSTM, which is a special recurrent neural network, is used as a base visual classifier. The LSTM model is trained for a different number of units(neurons), epochs, regularization, learning rate and other model hyperparameter and tuning options. During the training of our LSTM, we experimented with two types of methodology: training with fixed sequence and dynamic sequence size. LSTM trained with a fixed sequence size performs well such that the visual data are arranged in such a way that every clip is segmented with a bag of 45 frame sequence and later reshuffle the whole visual training data. Whereas LSTM trained with a dynamic size performs a little slower performance compared with the fixed one. the former model is trained with variable size for each training sample also known as online learning. This model is, even though it has slightly low performance, chosen because we have to use the same timestamp and length with the audio classifier model for further synchronized data integration. Figure 33 demonstrates the accuracy of the model for each emotion class.

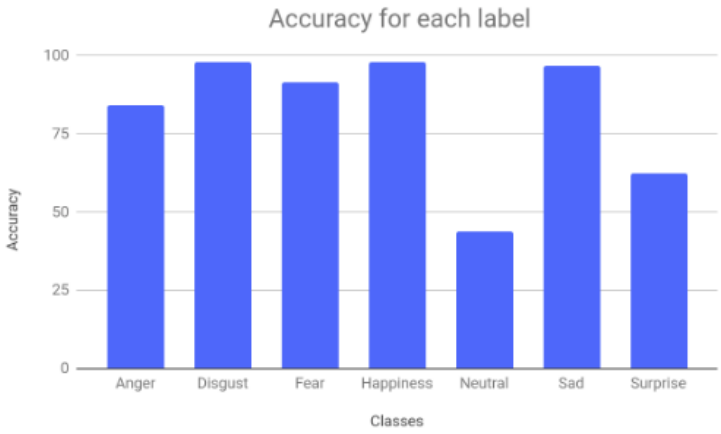


Figure 33: Visual Classifier on Training data

As is shown in figure 33 the visual model learned well for emotions disgust, fear, anger, happiness and sadness. Neutral and surprise emotions are the ones which are poorly learned by the model. The overall accuracy of the model on training data is 85%.

6.2.2 Accuracy on testing

The LSTM visual classifier model is tested using a twenty five percent(25%), out of the total visual data of the RAVDESS database, hold out data. The model performance on an overall accuracy is 61%. This performance is slightly better compared to the audio classifier, which is 58%. Figure 34 shows the performance of the model for each class label with their corresponding accuracies.

From the bar graph shown in figure 34 the performance of the model on the test data for the emotions types disgust, happiness, fear and sadness is still as good as it was in training performance. But for the anger and surprise emotions, they decreased their accuracy compared to their training accuracy performances. To sum up, Overall performance of the model on the whole test set is 61%.

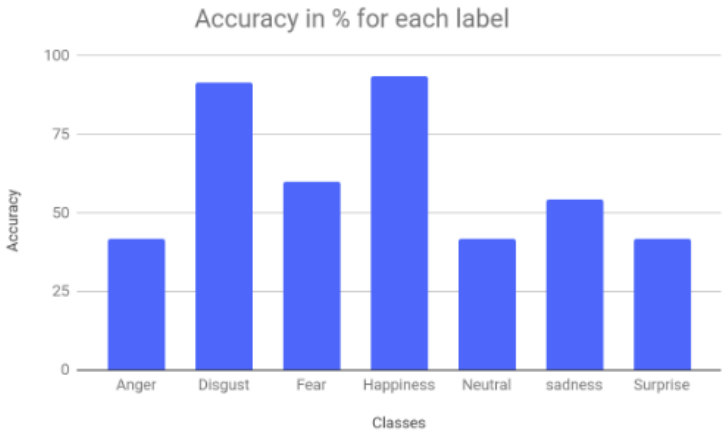


Figure 34: Visual Classifier on Testing data

6.3 Stacked model results

Finally, having the two trained models(audio and visual) their last second layer outputs were fused in order to train the stacked classifier and obtain the final emotion prediction. The fused vector for each training sample is composed of 350 features, 250 from audio classifier and 100 from the visual classifier. These 350 features per sample are used to train a new multiclass simple Dense fully connected network classifier in a stacked fashion with the same experimental setup as in the previous experiments used for the base classifiers.

The stacked classifier receives both feature inputs extracted using the audio and visual classifiers. Then it learns how to combine the inputs and find a pattern to increase the prediction force of the model. The two trained base classifiers (audio and visual classifiers) are used as feature extractors for the stacked classifier. The stacked classifier is a simple dense neural network that is trained and tested on the RAVDESS database. This model is trained with on 25% data of RAVDESS, which is explicitly reserved and is not used for training or testing during the base classifiers. Then these raw training data are given to their corresponding base classifiers for further feature extraction.

After features are extracted for each sample from the audio and visual part, both features, sample wise, are combined using a concatenation means. This step is repeated until all of the samples of the training dataset are combined. The model is trained with the new combined features according to the model hyperparameters. Similarly, the test data used for the model are combined and prepared following the procedure exactly the same as the training dataset. The

training process with its performance and testing performance are described in the subsections below.

6.3.1 Accuracy on training

As described previously, the simple DNN final classifier is trained on features that are from both the audio and visual. The training process is conducted by tuning the different options of the model hyperparameters and other important hyperparameters. The history training process for this final classifier is shown in figure 35 and figure 36 that represent the accuracy and loss between the training and validation dataset respectively.

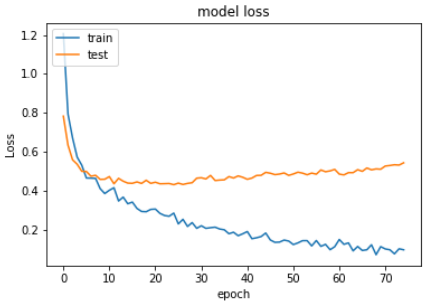


Figure 35: Stacked model's Accuracy for training and validation dataset

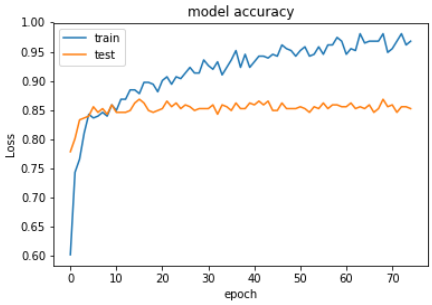


Figure 36: Stacked model's loss for training and validation dataset

Likewise, the performance of the model on each of the emotions on the training data is shown as a heatmap and as a bar graph in table 2 and figure 37 respectively.

	Anger	Disgust	Fear	Happiness	Neutral	Sadness	Surprise
Anger	48	0	0	0	0	0	0
Disgust	0	42	2	2	0	2	0
Fear	0	0	48	0	0	0	0
Happiness	0	0	0	46	0	1	1
Neutral	0	0	0	0	23	1	0
Sadness	0	6	0	2	0	40	0
surprise	0	0	0	0	0	1	47

Table 2: Stacked model's Heatmap on training data

As it is seen from table 2 as well as figure 37, the stacked model outperforms very well from the individual models as expected. This means the fused model is able to learn how to combine the two features and find important patterns among the features that lead it for better accuracy. The overall accuracy of the model on the training is 94%, which is much better than the individual classifiers.

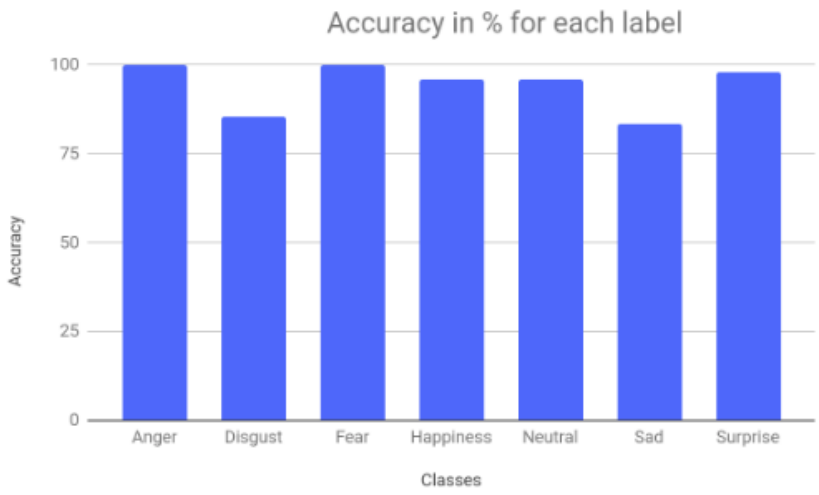


Figure 37: Stacked Classifier on training data

	Anger	Disgust	Fear	Happiness	Neutral	Sad	Surprise
Anger	47	0	0	0	0	1	0
Disgust	2	40	0	1	0	5	0
Fear	2	1	42	1	0	1	1
Happiness	0	1	1	44	0	0	2
Neutral	0	0	0	0	21	3	0
Sadness	0	3	4	1	1	39	0
Surprise	1	0	4	3	0	1	39

Table 3: Stacked model's Heatmap on testing data

6.3.2 Accuracy on testing

Testing this model means that testing the whole architecture model performance because it is the one that is responsible to categorize a given emotion expression to one of the seven corresponding labels. After the model is well trained, the model is tested on the new combined test dataset, which is 25% out of the total RAVDESS database. The obtained results for each of the emotions labels are shown in table 3 and figure 38 respectively.

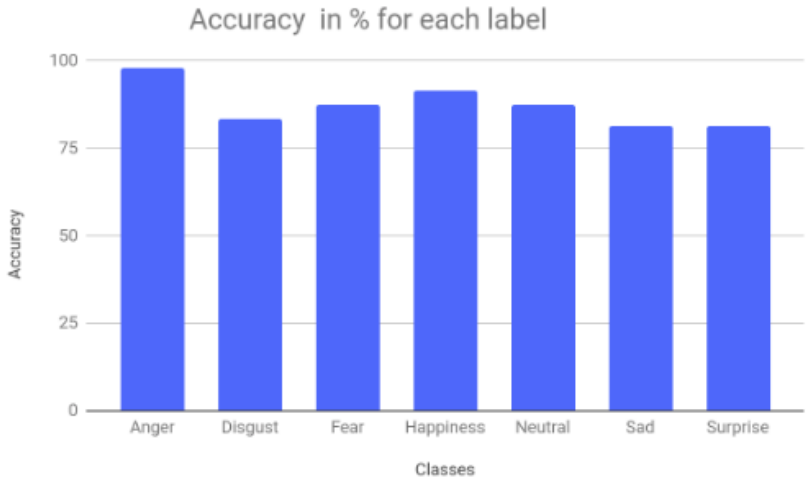


Figure 38: Stacked Classifier performance on testing data

As shown from the bar graph of figure 38, the model performs well on the test

dataset for all the different emotion types. The overall accuracy of the model on the test dataset is 87%. This indicates that the performance has improved by more than 25% compared to performances of the individual accuracies on the test dataset.

From the above three classifier results, it is apparent that the stacked classifier has improved a great predicting force than the individual classifier. There are two main reasons why this happens: combining data features and using the stacking method of the machine learning algorithm. Having data features that are combined from different data channels helps to obtain more information for easy extraction of patterns and this supports the stacked model to identify advantages and limitations of the previous models. The second reason is due to the introduction of the stacked generalization model. This type of model is experimentally proved that combining models that are trained separately performs better than the single models. This is because the model tries to learn the limitations and good patterns of the model and learns a new pattern to increase the accuracy of the model.

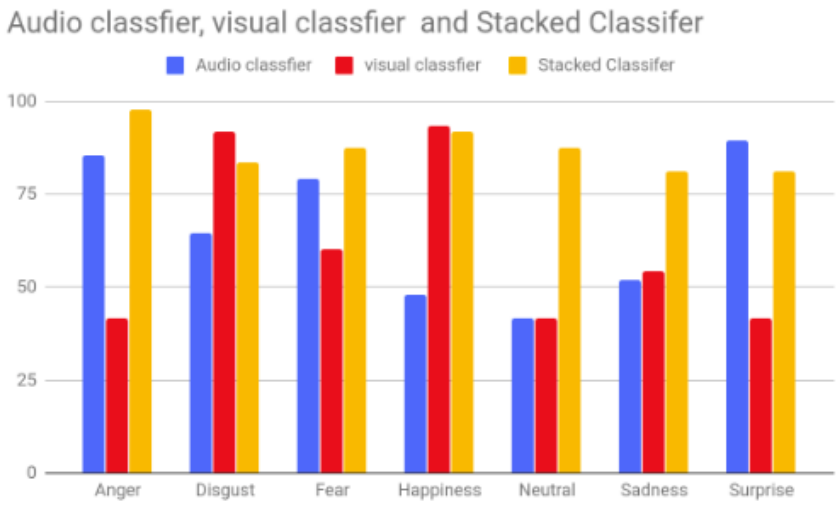


Figure 39: Three models Comparison on the test dataset

The performance of the three models: audio model, visual model and stacked classifier, on the test dataset of the database, for each of the emotion types are provided in figure 39 for visual comparison reasons. Overall performance for

audio classifier , visual classifier, and stacked classifier are 58%, 61%, and 87% respectively.

From the comparison graph shown in figure 39 the last classifier (stacked classifier) performs above 80% for all of the emotion labels. This performs comes from the idea that in machine learning and deep learning algorithms especially for classification problems stacking models have a more predictive force than any individual models. Our model exactly exploited this kind of improvement techniques, which helps us to have a moderate emotion recognizer model.

6.4 Final comments

For having the best emotion recognizer model, we have applied different feature extraction as well as model selection methodology options for both audio and visual classifiers. For audio classifier, we have tried to train the model using features that are extracted by a means of feature engineering. Each audio clip is given to MFCC extractor with a window length of 25ms and window step of 10ms and 13 feature vector is generated for each window size. For everysingle audio file 2d feature are obtained and then flattened to make a single vector feature of specific audio attached with the ground truth of this full single clip audio. The model trained using theses features performs worst compare to a model trained with features extracted from a pre-trained model.

Another experiment we have applied is increasing the training dataset for both audio and visual classifier to check if increasing training dataset has an effect on the performance of the model keeping other parameters and model hyperparameters intact. For visual model when it is trained with 50% of the dataset and performs 61% whereas when it is trained with 75%, adding 25% of the visual dataset normally used for training the stacked classifier, of the database and performs with an accuracy of 72%. Similarly, when an audio classifier is trained with 50% of the dataset and performs 58% whereas when it is trained with 75% dataset of the database, adding 25% of the audio dataset normally used for training the stacked classifier, performs an accuracy of 68%. Both classifiers improve their performance accuracy by 10% compared to the performance of the models trained with their datasets that we have used(50%). Therefore, for high predicting performance, it is better to train the models with more similar datasets than we have used.

On the contrary, we have tried to train the models with datasets that are col-

lections of two databases(SAVEE and RAVDESS). We have exactly used the same architectures and other settings of the models to see their performance. the accuracy of both models becomes exactly the same as the models that are trained with 50% of the RAVDESS database only. This shows that adding more training from other databases that are much different from the dataset we have used does not have an effect whereas having more training dataset similar to the one which is used, improves the performance force of the models.

7 Conclusion and future work

7.1 Conclusions

In this paper, we addressed the idea of emotion recognition from video using deep learning techniques and one of the ensembles called Stacking. One of the main contributions of our is to train an emotion recognition model using a sequence of frames from the video clip for recognizing the expressed emotion rather usually used a single frame. Moreover, we used combined features from both the audio channel as well as the visual channel to train the final classifier of the application using a newly introduced machine learning model ensembles called stacking generalizations.

The proposed video (audio-visual) emotion recognition system which performs averagely really good performance among the different labels of emotions is implemented. Audio features were transformed into a spectrogram and the spectrogram is given to a VGGsih, a CNN pre-trained model, as a base model for feature extraction.

Similarly, visual features were computed from a sequence of frames representing each video clip. Visual features are extracted using CNN based pre-trained model called VGG16. After we have trained the base classifier for each of the channels separately, they are used as feature extractors for the final classifier. the second last layer output values of the classifiers are fused to create a new feature that is used by the stacked classifier for learning as a final classification predictor. The experimented stacked audio-visual classifier results in 87% accuracy on the testing data. It significantly improves upon the two independent classifiers on the RAVDESS database by more than 26% accuracy. This result indicates that adding a stacking classifier and using combined features from multi-channel inputs improves and has more predictive performance compared to other fusion methodologies.

7.2 Feature Work

Recognizing emotion from multi-channel inputs and combining features opens some important finding for future research topics. In particular, using a certain sequence of frames and audio segments within a clip helps to identify and recognize whether the expressed emotion was real or fake. To do so there are two steps

or phases: recognizing the type of emotion and checking the recognized emotion if it is real or fake.

In this paper, recognizing part of the emotion is done, which is the first phase. By further investigation and exploiting the first phase the second phase can be introduced. Therefore, we highly recommend anyone interested to take this paper one step forward by implementing an application that identifies whether a given emotion is a real emotion expression or fake(sarcastic) expression. For better performance accuracy of our model, we also recommend training the model with more training dataset than that has been used in this paperwork.

Appendix

Sample Code:
For the implementation of the whole design varieties of scripts are developed at different stages like libraries used, preprocessing, feature extraction and construction of the classifier models. Below are some samples that are used in the main parts of the development.

======== **libraries importing** ==========

```
from __future__ import division

import numpy as np
import mel_features
import vggish_params
import resampy
import soundfile as sf
import pickle
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import os
import cv2
import glob
from PIL import Image
import ntpath
from skimage.io import imread
import random
from skimage.color import rgb2gray
import copy
import tensorflow as tf
from keras.models import Sequential
from keras import backend as K
from keras.applications.vgg16 import VGG16
from keras.layers import Dense,InputLayer,Dropout
from keras.layers import LSTM
```

```
from keras.optimizers import Adamax
from keras import losses
from keras.applications.vgg16 import preprocess_input
from keras.utils import np_utils
from keras.models import load_model
from keras.models import Model
from keras.callbacks import EarlyStopping
from keras.callbacks import ModelCheckpoint
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from keras import regularizers
from sklearn import svm, metrics, datasets
from sklearn.utils import Bunch
from sklearn.model_selection import GridSearchCV
from sklearn.model_selection import train_test_split
%matplotlib inline
import ntpath
from scipy.io import wavfile
import pylab
from scipy import signal
import wave
from sklearn.model_selection import train_test_split
from keras.layers import Conv2D, MaxPooling2D,Activation
from keras.layers import Flatten,GlobalMaxPooling2D
```

=== Audio preprocessing for a given audio file ===

```
# preprocesing function that returns spectrum samples

def preprocess_sound(data, sample_rate):
    if len(data.shape) > 1:
        data = np.mean(data, axis=1)
    # Resample to the rate assumed by VGGish.
    if sample_rate != vggish_params.SAMPLE_RATE:
        data = resampy.resample(data, sample_rate,
        vggish_params.SAMPLE_RATE)
    # Compute log mel spectrogram features.
    log_mel = mel_features.log_mel_spectrogram(
        data,
        audio_sample_rate=vggish_params.SAMPLE_RATE,
        log_offset=vggish_params.LOG_OFFSET,
        window_length_secs=vggish_params.STFT_WINDOW_LENGTH_SECONDS,
        hop_length_secs=vggish_params.STFT_HOP_LENGTH_SECONDS,
        num_mel_bins=vggish_params.NUM_MEL_BINS,
        lower_edge_hertz=vggish_params.MEL_MIN_HZ,
        upper_edge_hertz=vggish_params.MEL_MAX_HZ)

    # Frame features into examples.
    features_sample_rate = 1.0/vggish_params.STFT_HOP_LENGTH_SECONDS
    example_window_length = int(round(
      vggish_params.EXAMPLE_WINDOW_SECONDS * features_sample_rate))
    example_hop_length = int(round(
      vggish_params.EXAMPLE_HOP_SECONDS * features_sample_rate))
    log_mel_examples = mel_features.frame(
        log_mel,
        window_length=example_window_length,
        hop_length=example_hop_length)
    return log_mel_examples
```

```
# a function that reads the audio and send to the preprocessing
# function by normalizing it

def wavfile_to_examples(wav_file):
    wav_data, sr = sf.read(wav_file, dtype='int16')
    assert wav_data.dtype == np.int16, 'Bad sample type:
    %r' % wav_data.dtype
    samples = wav_data / 32768.0  # Convert to [-1.0, +1.0]
    audio_length=sr*5
    if len(samples) < audio_length:
        tempo=np.zeros(audio_length)
        tempo[:samples.shape[0]]=samples
        samples=tempo
    elif len(samples) > audio_length:
        samples=samples[:audio_length]

    return  preprocess_sound(samples, sr)
```

====== **VGGish model architecture**==========

```
# builting a similar architecture of  vggish

def build_model_vggish():

    model_vggish = Sequential()

    model_vggish.add(Conv2D(64, (3, 3), strides=(1, 1),
    activation='relu', padding='same',
    name='conv1',input_shape=(496,64,1)))
    model_vggish.add(MaxPooling2D((2, 2), strides=(2, 2),
    padding='same', name='pool1'))

            # Block 2
    model_vggish.add(Conv2D(128, (3, 3), strides=(1, 1),
    activation='relu', padding='same', name='conv2'))
    model_vggish.add(MaxPooling2D((2, 2), strides=(2, 2),
```

```
padding='same', name='pool2'))

        # Block 3
model_vggish.add(Conv2D(256, (3, 3), strides=(1, 1),
activation='relu', padding='same', name='conv3/conv3_1'))
model_vggish.add(Conv2D(256, (3, 3), strides=(1, 1),
activation='relu', padding='same', name='conv3/conv3_2'))
model_vggish.add(MaxPooling2D((2, 2), strides=(2, 2),
padding='same', name='pool3'))

        # Block 4
model_vggish.add(Conv2D(512, (3, 3), strides=(1, 1),
activation='relu', padding='same', name='conv4/conv4_1'))
model_vggish.add(Conv2D(512, (3, 3), strides=(1, 1),
activation='relu', padding='same', name='conv4/conv4_2'))
model_vggish.add(MaxPooling2D((2, 2), strides=(2, 2),
padding='same', name='pool4'))
model_vggish.add(GlobalMaxPooling2D())
return model_vggish
```

====== parameter setting for Audio file processing======

```
# Architectural constants.
NUM_FRAMES = 496  # Frames in input mel-spectrogram patch.
NUM_BANDS = 64  # Frequency bands in input mel-spectrogram patch.
EMBEDDING_SIZE = 128  # Size of embedding layer.

# Hyperparameters used in feature and example generation.
SAMPLE_RATE = 16000
STFT_WINDOW_LENGTH_SECONDS = 0.025
STFT_HOP_LENGTH_SECONDS = 0.010
NUM_MEL_BINS = NUM_BANDS
MEL_MIN_HZ = 125
MEL_MAX_HZ = 7500
# Offset used for stabilized log of input mel-spectrogram.
LOG_OFFSET = 0.01
# Each example contains 96 10ms frames
EXAMPLE_WINDOW_SECONDS = 4.96
EXAMPLE_HOP_SECONDS = 4.96     # with zero overlap.

# Parameters used for embedding postprocessing.
PCA_EIGEN_VECTORS_NAME = 'pca_eigen_vectors'
PCA_MEANS_NAME = 'pca_means'
QUANTIZE_MIN_VAL = -2.0
QUANTIZE_MAX_VAL = +2.0

# Hyperparameters used in training.
INIT_STDDEV = 0.01#Standard deviation used to initialize weights.
LEARNING_RATE = 1e-4  # Learning rate for the Adam optimizer.
ADAM_EPSILON = 1e-8  # Epsilon for the Adam optimizer.

# Names of ops, tensors, and features.
INPUT_OP_NAME = 'vggish/input_features'
INPUT_TENSOR_NAME = INPUT_OP_NAME + ':0'
OUTPUT_OP_NAME = 'vggish/embedding'
OUTPUT_TENSOR_NAME = OUTPUT_OP_NAME + ':0'
AUDIO_EMBEDDING_FEATURE_NAME = 'audio_embedding'
```

= face detection and re-sizing visual dataset =

```
# Function for face detection and cropping

def face_crop():
    for filename in glob.glob(path + '/' + folder +'/'+ subfolder
    +'/*.jpg'):
        image = cv2.imread(filename)
        img = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
        face_view = face_detector.detectMultiScale(img,1.3,5)
        for (x,y,w,h) in face_view:
            faceimg = img[y:y+h, x:x+w]
            cv2.imwrite(filename,faceimg)
            break


# Function for resizing the frames

def image_resize():
    for folder  in sorted(os.listdir(path)):
        for subfolder in sorted(os.listdir(path + '/' +folder)):
            for filename in glob.glob(path + '/' + folder +'/'+
            subfolder +'/*.jpg'):
                image = cv2.imread(filename)
                cv2.imwrite(filename,cv2.resize(image,(48,48)))
```

== visual data reshaping , feature extraction functions ===

```
#creating the pre-trained base-model
if not sys.warnoptions:
    warnings.simplefilter("ignore")

base_model_vgg16 = VGG16(include_top = False,input_shape
=(48,48,3),pooling = 'avg', weights = 'imagenet')

#reshaping the input data


def change_input_into_3D(x_input, size):
    vgg_input = np.empty([size, 48, 48, 3])
    for index, item in enumerate(vgg_input):
        item[:, :, 0] = x_input[index]
        item[:, :, 1] = x_input[index]
        item[:, :, 2] = x_input[index]
    return vgg_input


# iterater for all the bags of the frames

def into_3d(input_data):
    xp=[]
    for i,item in enumerate(input_data):
        x3d=change_input_into_3D(item, int(len(item)))
        xp.append(x3d)
    return xp


#function that returns the extracted values
def Extract_feature(image_input):
    z=[]
    for i,item in enumerate(image_input):
        x=base_model_vgg16.predict(item)
        z.append(x)
    return z
```

References

[1] URL: https://www.analyticsindiamag.com/6-types-of-artificial-neural-networks-currently-being-used-in-todays-technology/.

[2] URL: https://thoughtcatalog.com/january-nelson/2018/06/list-of-emotions/.

[3] URL: https://www.gartner.com/smarterwithgartner/13-surprising-uses-for-emotion-ai-technology/.

[4] URL: https://www.datascience.com/blog/convolutional-neural-network.

[5] TS Ashwin, Sai Saran, and G Ram Mohana Reddy. "Video affective content analysis based on multimodal features using a novel hybrid svm-rbm classifier". In: *2016 IEEE Uttar Pradesh Section International Conference on Electrical, Computer and Electronics Engineering (UPCON)*. IEEE. 2016, pp. 416–421.

[6] Anja Austermann et al. "Fuzzy emotion recognition in natural speech dialogue". In: *ROMAN 2005. IEEE International Workshop on Robot and Human Interactive Communication, 2005.* IEEE. 2005, pp. 317–322.

[7] Kiavash Bahreini, Wim van der Vegt, and Wim Westera. "A fuzzy logic approach to reliable real-time recognition of facial emotions". In: *Multimedia Tools and Applications* (2019), pp. 1–24.

[8] Dario Bertero et al. "Real-time speech emotion and sentiment recognition for interactive dialogue systems". In: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. 2016, pp. 1042–1047.

[9] L Daniel. *Psychology Second Edition. 41 Madison Avenue, New York, NY 10010*. 2011.

[10] Dan Duncan, Gautam Shine, and Chris English. "Facial emotion recognition in real time". In: *Stanford University* (2016).

[11] Jort F Gemmeke et al. "Audio set: An ontology and human-labeled dataset for audio events". In: *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2017, pp. 776–780.

[12] Jelena Gorbova et al. "Automated screening of job candidate based on multimodal video processing". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 2017, pp. 29–35.

[13] Shawn Hershey et al. "CNN architectures for large-scale audio classification". In: *2017 ieee international conference on acoustics, speech and signal processing (icassp)*. IEEE. 2017, pp. 131–135.

[14] Heysem Kaya, Furkan Gürpınar, and Albert Ali Salah. "Video-based emotion recognition in the wild using deep transfer learning and score fusion". In: *Image and Vision Computing* 65 (2017), pp. 66–75.

[15] Byoung Ko. "A brief review of facial emotion recognition based on visual information". In: *sensors* 18.2 (2018), p. 401.

[16] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "Imagenet classification with deep convolutional neural networks". In: *Advances in neural information processing systems*. 2012, pp. 1097–1105.

[17] Azadeh Kushki et al. "Comparison of blood volume pulse and skin conductance responses to mental and affective stimuli at different anatomical sites". In: *Physiological measurement* 32.10 (2011), p. 1529.

[18] Hosub Lee et al. "Towards unobtrusive emotion recognition for affective social communication". In: *2012 IEEE Consumer Communications and Networking Conference (CCNC)*. IEEE. 2012, pp. 260–264.

[19] Erik Marchi et al. "Voice-enabled assistive robots for handling autism spectrum conditions: an examination of the role of prosody". In: *Speech and Automata in the Health Care, A. Neustein, Ed. Walter de Gruyter GmbH & Co KG* (2014), pp. 207–236.

[20] Shasha Mo et al. "A novel feature set for video emotion recognition". In: *Neurocomputing* 291 (2018), pp. 11–20.

[21] Fatemeh Noroozi et al. "Audio-visual emotion recognition in video clips". In: *IEEE Transactions on Affective Computing* 10.1 (2017), pp. 60–75.

[22] Kyo-Joong Oh et al. "A chatbot for psychiatric counseling in mental healthcare service based on emotional dialogue analysis and sentence generation". In: *2017 18th IEEE International Conference on Mobile Data Management (MDM)*. IEEE. 2017, pp. 371–375.

[23] Shyamal Patel et al. "A review of wearable sensors and systems with application in rehabilitation". In: *Journal of neuroengineering and rehabilitation* 9.1 (2012), p. 21.

[24] Lucy Rathbone. "The role of emotions in obsessive-compulsive experiences". PhD thesis. Lancaster University, 2017.

[25] Frank A Russo and Steven Livingstone. "The ryerson audio-visual database of emotional speech and song". In: *Acoust. Week Canada* (2015).

[26] Björn Schuller, Lucas Paletta, and Nicolas Sabouret. "Intelligent Digital Games for Empowerment and Inclusion–An Introduction". In: *Proceedings 1st International Workshop on Intelligent Digital Games for Empowerment and Inclusion (IDGEI 2013) held in conjunction with the 8th Foundations of Digital Games*. 2013.

[27] Nicu Sebe. "Multimodal interfaces: Challenges and perspectives". In: *Journal of Ambient Intelligence and smart environments* 1.1 (2009), pp. 23–30.

[28] Jamie Shotton et al. "Real-time human pose recognition in parts from a single depth image". In: (2011).

[29] Peggy A Thoits. "The sociology of emotions". In: *Annual review of sociology* 15.1 (1989), pp. 317–342.

[30] Eero Väyrynen. "Emotion recognition from speech using prosodic features". In: *University of Oulu, Oulu* (2014).

[31] Timothy D Wilson and Elizabeth W Dunn. "Self-knowledge: Its limits, value, and potential for improvement". In: *Annu. Rev. Psychol.* 55 (2004), pp. 493–518.

[32] Kaili Zhao, Wen-Sheng Chu, and Honggang Zhang. "Deep region and multi-label learning for facial action unit detection". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 3391–3399.