PSYCHOMETRIC ANALYSIS TOOL

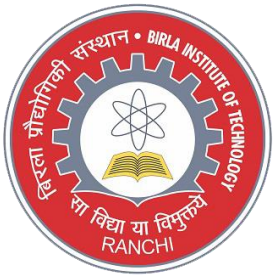
A project report submitted in partial fulfillment of the requirements for the award of the Degree of

BACHELOR OF ENGINEERING
IN
COMPUTER SCIENCE & ENGINEERING

BY

APURVA BHARGAVA (BE/25022/15)

PALLAVI JAIN (BE/25001/15)



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

BIRLA INSTITUTE OF TECHNOLOGY, MESRA
JAIPUR CAMPUS, JAIPUR

SP-2019

DECLARATION CERTIFICATE

This is to certify that the work presented in the project entitled "PSYCHOMETRIC ANALYSIS TOOL**"** in partial fulfillment of the requirements for the award of Degree of Bachelor of Engineering in Computer Science and Engineering of Birla Institute of Technology, Mesra, Ranchi, Extension Center Jaipur is an authentic work carried out under my supervision and guidance.

To the best of my knowledge, the content of this project does not form a basis for the award of any previous degree to anyone else.

Date: May 1 2019

Dr. Shripal Vijayvargia
Associate Professor
Birla Institute of Technology, Mesra ,Ranchi
Extension Centre,Jaipur

Contents

1. Introduction

Psychometrics, is a field of study concerned with the theory and technique involved behind psychological measurement. This field is primarily concerned with testing, measurement, and assessment, that is, objective measurement of skills and knowledge, abilities, attitudes, and personality traits. The two areas of research focus are- (i) the construction and validation of assessment instruments such as questionnaires, tests, raters' judgments, and personality tests, and (ii) measurement theory (e.g., item response theory; intra-class correlation).[1] The focus of the project is to employ machine learning models to automate certain aspects of psychometric analysis that usually require human intervention.

1.1 Problem Statement

To design a tool (Psychometric Analysis Tool) for interviewers and/or psychologists that powers judgement by determining the interviewee's psychological characteristics based on the detection and analysis of subjective factors such as facial expressions, speech emotions, written opinion and question answering.

1.2 Psychometrics and Machine Learning

Machine learning (ML) is the study of algorithms and statistical models that computer systems use to progressively improve their performance on a specific task. Machine learning algorithms build a mathematical model of sample data, known as "training data", in order to make predictions or decisions without being explicitly programmed to perform the task.[2], [3] Machine learning will facilitate the evaluation of psychometric information outside of a testing context, both from past information (e.g. internet corpus) and from current information (e.g. live cameras, smartphones). This may turn out to be a boon or a dystopian nightmare depending on who uses it and how it is used. (Which side you see is itself an interesting psychometric datum). Take for instance a non-controversial topic like IQ. Machine learning will evaluate how trainable a person is in general. What is the best way to train that person? Would this person do well in this new task? How long will the training last? All this without a test. Of course, psychometrics and psychometricians will still be needed to validate those algorithms.

This age-old domain is largely based upon the work of Charles Spearman, but the technological changes across the entire landscape of instruments, data, applications, and domains associated have brought machine learning techniques into psychometrics. Machine Learning is expected to revolutionize Psychometrics. IRT (Item Response Theory) psychometrics are usually based upon logistic regression techniques. However, the technique fails to promise best class of models for classification anymore. Machine Learning can be utilized to reveal candidate's strengths in the in the social components of collaborative problem solving, such as perspective taking, participation, and social regulation.

This can be achieved by simply analyzing a participant's video/audio recording, or social media data, leveraging Machine Learning techniques. It's already making some waves. Machine learning can be used to match employees with training programs that fit their profiles and career goals, predict responses to certain drugs (particularly neuropsychological disorders), and validation of a wider range of survey tools. Machine Learning techniques can extend to incorporate Virtual Reality technology. This will help towards expanding the scope of problem solving tasks, while enriching the resulting data stream. These techniques can also be applied to study difficult real life scenarios.

1.3 Literature Review

Facial or speech emotion recognition, as well as sentiment classification problem can be reformulated in mathematical terms as a classification task, which can very well be done using statistical machine learning models. Artificial neural networks or connectionist systems are computing systems vaguely inspired by the biological neural networks that constitute animal brains. Computational models of neural networks have been around for a long time, first model proposed was by McCulloch and Pitts.

Neural Network with Backpropagation: Neural networks are made up of a number of layers with each layer connected to the other layers forming the network. A feed-forward neural network or FFNN can be thought of in terms of neural activation and the strength of the connections between each pair of neurons. In FFNN, the neurons are connected in a directed way having clear start and stop place i.e., the input layer and the output layer. The layer between these two layers, are called as the hidden layers. Learning occurs through adjustment of weights and the aim is to try and minimize error between the output obtained from the output layer and the input that goes into the input layer. The weights are adjusted by process of back propagation (in which the partial derivative of the error with respect to last layer of weights is calculated). The process of weight adjustment is repeated in a recursive manner until weight layer connected to input layer is updated.

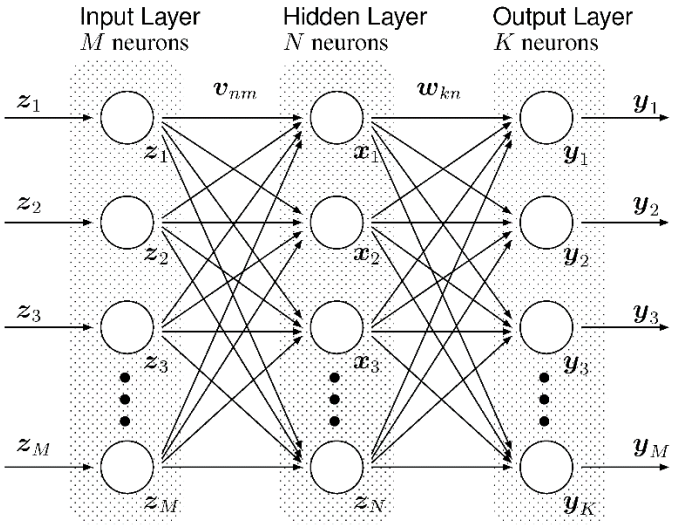


Figure 1

Convolutional Neural Networks (CNN): It is a variant of Multi- Layer Perceptron (MLP) which is inspired from vision, as observed in animals. Convolutional neural networks are designed to process two-dimensional (2-D) image.[4] A simple CNN architecture may consist of three types of layers namely convolution layer, sub sampling layer and the output layer. CNNs exploit spatially local correlation by enforcing a local connectivity pattern between neurons of adjacent layers. In the CNN algorithm, each sparse filter is replicated across the entire visual field. These units then form a feature maps, these share weight vector and bias. The gradient of shared weights is the sum of the gradients of the parameters being shared. Such replication in a way allows features to be detected regardless of their position in visual field. In addition to this, weight sharing also allows to reduce the number of free learning parameters. Due to this control, CNN tends to achieve better generalization on vision problems. CNN also make use of the concept of max-pooling, which is a form of non-linear down-sampling. In this method, the input image is partitioned into non-overlapping rectangles. The output for each sub-region is the maximum value. Recent deep FER systems generally focus on two important issues: overfitting caused by a lack of sufficient training data and expression-unrelated variations, such as illumination, head pose and identity bias.[5], [6]

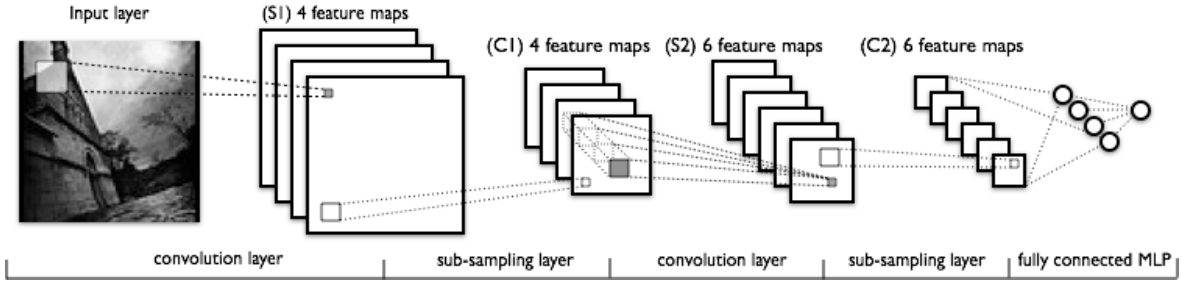


Figure 2

Recurrent Neural Network (RNN): It is a class of neural networks whose connections between neurons form a directed cycle. Unlike feedforward neural networks, RNN can use its internal "memory" to process a sequence of inputs, which makes it popular for processing sequential information. The "memory" means that RNN performs the same task for every element of a sequence with each output being dependent on all previous computations, which is like "remembering" information about what has been processed so far.

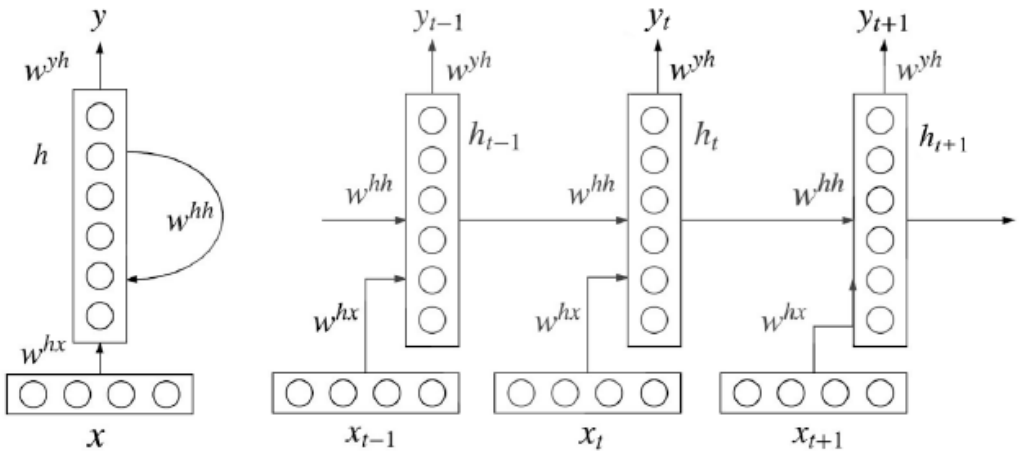


Figure 3

The left graph is an unfolded network with cycles, while the right graph is a folded sequence network with three time steps. The length of time steps is determined by the length of input. x_t is the input vector at time step t. h_t is the hidden state at time step t, which is calculated based on the previous hidden state and the input at the current time step.

$$h_t = f(w^{hh}h_{t-1} + w^{hx}x_t)$$

The activation function f is usually the tanh function or the ReLU function. w^{hx} is the weight matrix used to condition the input x_t . w^{hh} is the weight matrix used to condition the previous hidden state h_{t-1}. y_t is the output probability distribution over at step t.

$$y_t = softmax(w^{yh}h_t)$$

The hidden state h_t is regarded as the memory of the network. It captures information about what happened in all previous time steps. y is calculated solely based on the memory h_t at time t and the corresponding weight matrix w^{yh}.[7], [8]

Long Short Term Memory network (LSTM): It is a special type of RNN, which is capable of learning long-term dependencies. All RNNs have the form of a chain of repeating modules. In standard RNNs, this repeating module normally has a simple structure. However, the

repeating module for LSTM is more complicated. Instead of having a single neural network layer, there are four layers interacting in a special way. Besides, it has two states: hidden state and cell state.[9]

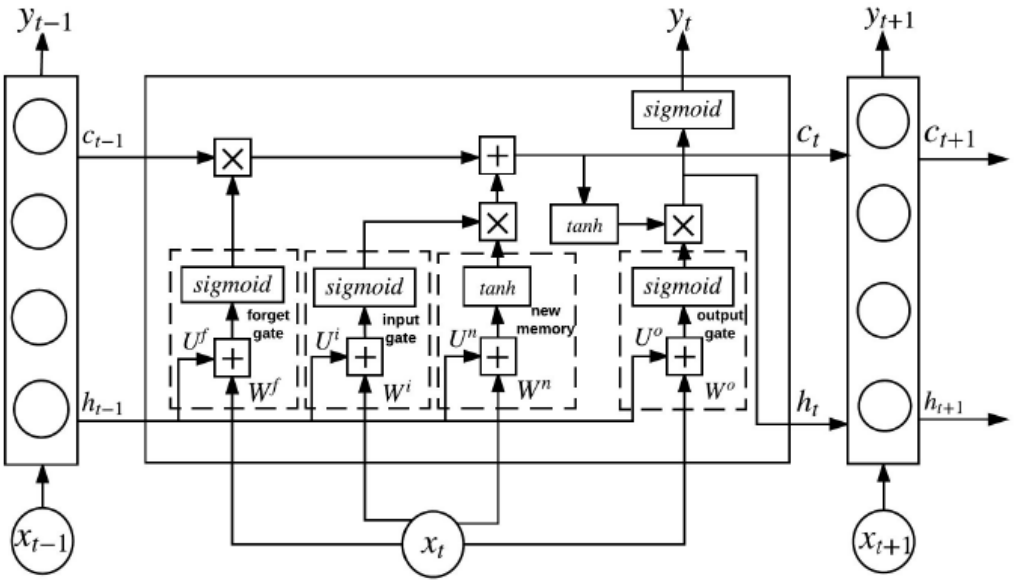


Figure 4

At time step t, LSTM first decides what information to dump from the cell state. This decision is made by a sigmoid function/layer σ, called the "forget gate". The function takes h_{t-1} (output from the previous hidden layer) and x_t (current input), and outputs a number in [0, 1], where 1 means "completely keep" and 0 means "completely dump".

$$f_t = \sigma(W^f x_t + U^f h_{t-1})$$

Then LSTM decides what new information to store in the cell state. This has two steps. First, a sigmoid function/layer, called the "input gate", decides which values LSTM will update. Next, a tanh function/layer creates a vector of new candidate values $\tilde{C}_t$, which will be added to the cell state. LSTM combines these two to create an update to the state.

$$i_t = \sigma(W^i x_t + U^i h_{t-1})$$

$$\tilde{C}_t = \tanh(W^n x_t + U^n h_{t-1})$$

It is now time to update the old cell state C_{t-1} into new cell state C_t. Note that forget gate f_t can control the gradient passes through it and allow for explicit "memory" deletes and

updates, which helps alleviate vanishing gradient or exploding gradient problem in standard RNN.

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Finally, LSTM decides the output, which is based on the cell state. LSTM first runs a sigmoid layer, which decides which parts of the cell state to output, called "output gate". Then, LSTM puts the cell state through the tanh function and multiplies it by the output of the sigmoid gate, so that LSTM only outputs the parts it decides to.[9], [31]

$$o_t = \sigma(W^o x_t + U^o h_{t-1})$$

$$h_t = o_t * \tanh(C_t)$$

Cosine Similarity: Cosine similarity is a measure of similarity between two non-zero vectors of an inner product space that measures the cosine of the angle between them [15]. The cosine of 0° is 1, and it is less than 1 for any angle in the interval (0,π] radians. It is thus a judgment of orientation and not magnitude: two vectors with the same orientation have a cosine similarity of 1, two vectors oriented at 90° relative to each other have a similarity of 0, and two vectors diametrically opposed have a similarity of -1, independent of their magnitude. The cosine similarity is particularly used in positive space, where the outcome is neatly bounded in [0,1].

The cosine of two non-zero vectors can be derived by using the Euclidean dot product formula:

$$\mathbf{A}.\mathbf{B} = \big||\mathbf{A}|\big|\,\big||\mathbf{B}|\big|\,cos\theta$$

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A}\cdot\mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}},$$

Jaccard Similarity: The Jaccard index, also known as Intersection over Union and the Jaccard similarity coefficient, is a statistic used for comparing the similarity and diversity of sample sets [16], [17]. The Jaccard coefficient measures similarity between finite sample sets, and is defined as the size of the intersection divided by the size of the union of the sample sets:

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}.$$

Levenshtein Distance: The Levenshtein distance is a string metric for measuring the difference between two sequences. Informally, the Levenshtein distance between two words is the minimum number of single-character edits (insertions, deletions or substitutions) required to change one word into the other. Mathematically, the Levenshtein distance between two strings a, b (of length |a| and |b| respectively) is given by $lev_{a,b}(|a|, |b|)$ where

$$\operatorname{lev}_{a,b}(i,j) = \begin{cases} \max(i,j) & \text{if } \min(i,j) = 0, \\ \min \begin{cases} \operatorname{lev}_{a,b}(i-1,j) + 1 \\ \operatorname{lev}_{a,b}(i,j-1) + 1 \\ \operatorname{lev}_{a,b}(i-1,j-1) + 1_{(a_i \neq b_j)} \end{cases} & \text{otherwise.} \end{cases}$$

Where $1_{(ai \neq bj)}$ is the indicator function equal to 0 when $a_i = b_j$ and equal to 1 otherwise and $lev_{a,b}(i, j)$ is the distance between the first i characters of a and the first j characters of b [18].

2. Objectives

Psychometric Analysis Tool acts as an interface between the two interacting parties, one of whom evaluates the other's psychological state and competence using the various modules provided by the software as mentioned further. It automates, to a large extent, the task of the interviewer or psychologist. It can be widely used by the companies for recruitment purposes. Also, a virtual psychologist may use it for evaluation of the patient's emotional state. This is achieved by various modules that analyze the responses of the interviewee or patient, which are in the form of text, speech, facial emotion, etc.

Objectives realized in the project are stated as follows:

2.1 Expression Recognition from Face

While the interviewee speaks or gives a verbal response to the prompt or question, images are captured periodically from the webcam or video camera feed (where faces are detected in real time using Haar-cascade based face detection). The facial expression recognition model classifies each image, calculating the class probability for each possible target class (angry, disgust, fear, happy, sad, neutral, or surprise). The two classes with highest probabilities are used for analysis.

2.2 Emotion Recognition from Speech

The interviewee's verbal response to the question or prompt is recorded. The speech emotion recognition model is used on the sequences of pressure levels (of successive frames) extracted from the audio recording to assign a probability for each possible target class (angry, disgust, fear, happy, sad, neutral, or surprise). The two classes with highest probabilities are used for analysis.

2.3 Confidence and Certainty Scores of Speech Converted to Text

The interviewee's response to the prompt or question is converted to text. The text is processed and a list of n-grams is generated. A lexicon of words relevant to confidence and certainty levels is used to calculate confidence score and certainty score (between -1 and +1).

2.4 Sentiment Analysis of Image-based Description

The interviewee writes an honest descriptions of the shown image. The text in the description is processed and converted to a feature vector. The sentiment analysis model takes the feature vector as input and calculates the probability for each target class (positive, neutral, or negative). This is used to determine the polarity of the interviewee's opinions, and thus, his/her attitude.

2.5 Adaptive Interviewing Chatbot

The interviewee interacts with the bot. The bot asks interview questions to which the interviewee writes a response. The database contains the expected answer to the question. The given and expected responses are compared using similarity measures like cosine similarity and jaccard similarity, and a score is assigned to the interviewee's response. Depending on the interviewee's performance and difficulty level of the question, the bot selects the next question. After the interview, the user can have a conversation with or ask domain-specific questions from the bot.

2.6 Result as the Analysis of the Various Inputs to Each Individual Functionality

For each functionality (audio or video analysis, speech to text, image description, or interview question/answer), raw results are displayed as plain values, scores or bar charts as well as a simple and brief analysis of the raw results is shown under the 'View Report' section.

2.7 Browser User Interface, Frontend and Backend

The tool runs on a web browser. Bootstrap4 and ReactJS are used for UI and frontend and NodeJS for backend. There are three pages accessible from the navigation bar– Create Test, Take Test and View Report. The test has three phases- audio/video emotion recognition with

speech to text for confidence and certainty determination, sentiment analysis of image-based description, and an interview with a chatbot with a query answering follow-up session.

3. SRS (Software Requirements Specification)

3.1 Introduction

3.1.1 Purpose

This document describes the software requirements specification (SRS) for the application software that enables the psychometric analysis of an individual. Psychometric Analysis Tool looks at measuring the subjective characteristics of an individual: a tool that powers assessment based on mental and emotional state of an individual. It can be used by companies for psychological evaluation for recruitment purposes or by those preparing for interviews for self-evaluation. Also, a virtual psychologist can use it for measuring and quantifying a range of different metrics and personal characteristics through the use of psychometric testing. This is achieved by various modules that analyze the responses of the interviewee in the form of speech, facial expression, written opinion and question answering.

3.1.2 Document Conventions

There is no special highlighting. Font sizes 16 for headings and 14 for sub-headings has been used throughout the document for maintaining uniformity. The font size for text is 12. Square bullet list is used in case of listing the features and similar.

3.1.3 Intended Audience and Reading Suggestions

- The document is intended for researchers, software developers, advanced practitioners, documentation writers, users, testers and evaluators. The SRS contains the requirements and perspective of the software in an elaborated and organized manner which should be read in the same sequence as it is written.
- In the next section, system features with their functional requirements are presented to highlight the major services provided by the intended product. Then the external interface requirements highlighting the logical characteristics of each interface between the software product and the users are discussed. Finally, this specification is concluded with the reference documents on which this document is based on.

3.1.4 Product Scope

We describe what features are in the scope of the software and what are not in the scope of the software to be developed:

In the scope:

- Test creator can create, modify and delete tests.
- Test creator can add, remove or modify images and questions for tests.
- Test taker can select test, give test and see responses and results.
- Face detection for facial emotion recognition

- Speech recording for speech emotion recognition
- Speech to Text Conversion for confidence and certainty testing
- Response as input by test giver in the form of speech, mouse click and text through keyboard

Out of the scope:

- Remembering old response in forms.
- Communication interface for test creators and test takers.

3.2. Overall Description

3.2.1 Product Perspective

The word 'psychometric' conflates the word 'psyche', which is defined as 'mind', with the word 'meter', which means 'measure'. So, psychometric testing and psychometric assessments are effective tools that quantify a person's psychological characteristics in a way that can be measured and analyzed. For example, psychometric tests and psychometric assessments can be used to measure with a great degree of accuracy the characteristics of a person like their personality, cognitive abilities, behavior patterns as well as a wide range of other factors. Psychometrics is the science of assessment. Psychometric Analysis Tool is a tool that powers judgment based on the mental assessment and emotional state of an individual. It is of similar utility as existing interviewers and/or psychologists, who tend to deduce the state of mind of a person by various observations and analysis. Our tool aims at provisioning solutions to administer the same in the absence of a human interviewer and/or psychologist. It acts as the interface between the two interacting parties, one of which evaluates the other using the various modules provided by the software as mentioned further.

3.2.2 Product Functions

- Provision for setting up test and interview
- Provision for selecting tests and viewing responses and results
- Emotion recognition from facial expressions
- Emotion recognition from speech
- Speech to text conversion for confidence and certainty measurement
- Sentiment Analysis of the image based description to test the attitude of a person
- Adaptive chatbot for holding conversations, interviewing user and answering queries
- Result as the analysis of the various inputs to each individual functionality

3.2.3 User Classes and Characteristics

There are two user classes: test creators and test takers. Test creator can create, modify and delete tests, as well as add, remove or modify images and questions for tests. Test taker can select test based on test name, give test and see responses and results. The most common among the test creators may be the recruiters of various small companies, who avail this service centrally. The other prospective test creators are the psychologists who could take advantage of our image description based sentiment analysis and the emotion recognition

features or people looking for a way to self-evaluate prior important interviews. Test takers may be interviewees or people seeking psychological evaluation.

3.2.4 Operating Environment

This is a tool that runs on a browser interface and hence will require any compatible browser on any operating system. A modular implementation approach would be useful to perform testing on modules at different stages to ensure correct implementation. It is anticipated that the tool will work seamlessly in any functional environment provided the complete and standard configuration and installation of the hardware (webcamera and microphone).

3.2.5 Design and Implementation Constraints

This system is provisioned to be built using natural language processing (NLP), machine learning and deep learning models implemented in Python. The tool is used through a browser interface. Bootstrap was used for UI, ReactJS for frontend and NodeJS and Django (Python) for backend. Decision regarding the database is taken considering the fact that data being stored is hierarchical, and hence, file system is used for images and questions. For the adaptive Chatbot, SQLite database and JSON files have been used for storing structured and semi-structured data.[21]

3.2.6 User Documentation

Along with the software product, a user manual would be written to help people understand the working methodology and usage of the developed prototype system. It would be written for nontechnical individuals and the level of content or terminology would differ considerably from, for example, a System Administration Guide, which is more detailed and complex. The user manual would follow common user documentation styles capturing purpose and scope of the product along with key system features and operations; step-by-step instructions for using the system including conventions, messaging structures, quick references, tips for errors and malfunctions; pointers to reference documents; and glossary of terms.

3.2.7 Assumptions and Dependencies

- For face and speech emotion classification, the seven classes are anger, disgust, fear, surprise, sadness, happiness and neutral.
- The language used by user is English.
- The scores for confidence and certainty lie between -1 and +1.
- For text, the sentiment classes are positive, negative and neutral.
- Webcamera and microphone (inbuilt or otherwise) are available.
- The combined observations from different modules are enough to reach to a conclusion.
- There are no dependencies that the project has on external factors.

3.3 External Interface Requirements

3.3.1 User Interfaces

This section describes the logical characteristics of each interface between the intended software product and the users. For user interface design, common UI standards will be followed along with the presence of keyboard shortcuts, error message display standards, pop-up notifications, alerts, etc., and standard buttons and functions will appear on every screen. The tool has a simplistic design and no prior experience or training is required to make use of it. The common page options are available on the top navigation bar. Exploring only requires using labeled tabs.

3.3.2 Hardware Interfaces

Reliable software device drivers usually handled by the OS distributors are required to run the webcamera, keyboard, mouse, and microphone for recording responses. All devices compatible with modern computing systems are supported.

3.3.3 Software Interfaces

The Psychometric Analyzer has different types of software interfaces (this term is used in a very broad meaning) to external packages, depending how the interaction is realized:
i. User Interface: The various functionalities of the application can be accessed by the means of simple buttons, tabs and forms.
ii. Message Interface: Since automation is the goal, methods have been implemented to run the various functionalities and communicate among different objects (instances of different classes) using APIs.
iii. Database Interface: The test creator can create and modify test and add, remove or modify images and questions-answers in the file system.

3.4 Functional Requirements

The major services and functional requirements for the product can be illustrated by system features. In the following, necessary description is provided for each module in the system. Each description provides information of the associated actors, triggering condition, preconditions, post-conditions, response sequences, exceptions and functional requirements (assumptions). Being a major important section of the SRS, this section is expected to go through iterative improvement to make the most logical sense for the intended product.

3.4.1 Face Expression and Speech Emotion Recognition

3.4.1.1 Introduction

This module provides the main emotion recognition function of the first phase of test where facial expressions and speech is captured and recorded in fixed intervals and sent to the emotion recognition models for classification. The user is given a question or prompt for the

recording. The web camera is handled by react-webcam library and microphone by react-mic library for capturing and saving images and audio for the classification models (written in Python using Keras) to recognize the emotional state. [38], [39]

3.4.1.2 Input

i. Audio feed from microphone

ii. Video feed from webcamera

3.4.1.3 Output

The result of the functionality is displayed on the results page at the end of the test. It includes a plot of the percentages of the strongest detected emotions among the seven, and a brief analysis. This exploits the CNN model for Facial Emotion Recognition and LSTM model for Speech Emotion Recognition using Keras.

3.4.2 Speech to Text Conversion for Confidence and Certainty Scoring

3.4.2.1 Introduction

This is a part of the first functionality. The verbal response of the user to the prompt or question is converted to text. This text is inserted as an argument to a function for calculation of confidence score and certainty score. The result is displayed on the results page at the end of the test.

3.4.2.2 Input

i. Audio feed from microphone

3.4.2.3 Output

The scores of the individual's response are on a scale of -1 to +1 for both confidence and certainty; +1 for most confident and most certain. This exploits NLP (Natural Language Processing) library, ntlk, in Python.

3.4.3 Sentiment Analysis of Image Based Description

3.4.3.1 Introduction

This analysis requires the responder to input a description for an image that is displayed to him/her within a pre-specified word limit. This text is then sent to the sentiment analysis model for estimation of the positive, negative and neutral perception percentages of an individual in response to the respective images. The result is displayed on the results page at the end of the test.

3.4.3.2 Input

i. Text input from keyboard

3.4.3.3 Output

The percentages of the individual's response being positive or negative is displayed, as well as a brief analysis. This exploits NLP (Natural Language Processing) and the ANN Keras model in Python for classification.

3.4.4 Adaptive Interview Chatbot

3.4.4.1 Introduction

Generalized Chatbot is trained using lists of conversations, and later consults sqlite database generated from the data. The question-answering system parses .json file to select questions and match responses. A confidence value or score is used to decide how the system responds. It chooses questions according to the question difficulty and user's performance on each question. The user can chat with the bot and ask questions post interview.

3.4.4.2 Input

i. Questions, answers and difficulty from file system (in build phase)

ii. User response (while working)

3.4.4.3 Output

Response to the user and calculated user score.

3.4.5 Browser User Interface, Frontend and Backend

3.4.5.1 Introduction

The frontend implementation uses ReactJS and NodeJS frameworks. Navigation on a page works through tabs for creating test, taking test and seeing report. All the functions including the clicks, navigation, inputs on the website are actions that are of importance in determining the behavior, responses and updates made to the database and this is how they move forward in their journey. The backend implementation uses NodeJS and Django (Python) frameworks. The file system interaction is handled by the NodeJS, for example, saving the images in folders, question-answers in .json files in separate folders for each user for each test. The processing of inputs and their responses in terms of score is generated by the python server and returned to the frontend.

3.4.5.2 Input

i. Questions and their expected answers as text entered by the test creator

ii. Images chosen by the test creator

iii. Responses of the user in different phases of the test

3.4.5.3 Output

Three-tier client server architecture is used because data is stored in database (File System) can be accessed and modified by the test creator and the test taker with the help of the GUI between them. Frontend output is the response to any event or action performed on the website that can be seen in the form of routing to different pages, the change of questions or images on next and previous buttons, starting, submitting and moving through the phases of the test, etc. All the inputs to the NodeJS server result in either publishing the data to files and folders or fetching it from there. The question-answers are saved as .json file, images are saved in the folder inside Creator→Test. The responses of the user in different phases are also saved in the .json files. These responses are sent to python server for evaluation and determination of score by using Machine Learning models or Chatbot system.

3.5 Other Nonfunctional Requirements

3.5.1 Performance Requirements

The tool will be interactive and the delays are dependent on the browser. In every action-response of the system, there are no immediate delays. In case of saving the settings or making submissions, there is delay much below 1 second. In case of changes made to database or file systems, sorting questions and evaluations there are no delays and the operation is expected to be performed in less than 1 second.

3.5.2 Security Requirements

Information transmission shall be securely transmitted to server/ database without any changes in information. Test creators and users or test takers have different use different functionalities and have restricted access to file systems and tool settings.

3.5.3 Software Quality Attributes

Availability: The tool runs on a browser and is independent of network connections pertaining to the node system on which it is installed. It can be run on any compatible browser on any operating system environment.
Usability: The tool is easy to handle and navigates in the most expected way with no delays. The built is responsive. The APIs and server handle requests swiftly.

4. SDS (Software Design Specification)

4.1 Introduction

This document is designed to be a reference for any person wishing to implement or any person interested in the architecture of the software. This document describes the application's architecture and sub architecture, the associated interfaces, the database and the motivations behind choosing the design. Both high level and low level designs are included in this document. This document should be read by an individual with a technical background and has experience reading data flow diagrams, control flow diagrams, interface designs and development experience in object-oriented programming as well as sequential programming.

The document will provide developers an insight in meeting client's needs efficiently and effectively. It would demonstrate how the design will accomplish the functional and nonfunctional requirements captured in the SRS.

4.1.1 Document Description

It is the "how will we do it" part after we wrote the "what will we do" part, which is the SRS.

The System Architecture section is the main focus of this document. It provides an overview of the system's major components and architecture, as well as specifications on the interaction between the system and the user.

The Detailed System Design description of components section will also be covered in this document. It will describe lower-level classes, components, and functions, as well as the interaction between these internal components. It contains specific information about the expected input, output, classes, and functions. The interactions between the classes to meet the desired requirements are outlined in detailed figures (class diagram) at the end of the document. It supplies a snapshot of the intended system from multiple points of view.

Here is the outline of the proposed software design specifications.

- Introduction
- System Overview
- Design Considerations
    - Assumptions and Dependencies
    - General Constraints
    - Goals and Guidelines
    - Development Methods
- System Architecture
    - Face Expression and Speech Emotion Recognition Models
    - Speech to Text for Confidence and Certainty Scoring

        - Sentiment Analysis Model
        - Adaptive Interview Chatbot
        - Browser User Interface
    - Detailed System Design
        - Real-time Emotional State Determination
        - Image Description-based Sentiment Analysis
        - Adaptive Interview Chatbot
        - Browser Interface Implementation
    - Use Case Diagrams
    - Dataflow Diagrams

4.1.2 System Overview

Psychometrics is the science of assessment of mental capacities and processes. Psychometric Analysis Tool powers judgment based on the assessment of emotional state of an individual by using webcam, microphone and keyboard input. It is a tool that can be used for assistance to existing interviewers and/or psychologists which aims at provisioning solutions to administer the same in the absence of a human interviewer and/or psychologist. It acts as the interface between the two interacting parties, one of which evaluates the other using the various modules provided by the software as mentioned further. It can be widely used by the companies for recruitment purposes. Also, a psychologist can use it for evaluation of emotional state or by a person as a self-evaluation tool for interview preparation. This is achieved by various modules that analyze the responses of the interviewee or patient in the form of text, speech, facial emotion, etc.

4.2 Design Considerations

This section describes many of the issues which need to be addressed or resolved before attempting to devise a complete design solution.

4.2.1 Assumptions and Dependencies

- For face and speech emotion classification, the seven classes are anger, disgust, fear, surprise, sadness, happiness and neutral.
- The language used by user is English.
- The scores for confidence and certainty lie between -1 and +1.
- For text, the sentiment classes are positive, negative and neutral.
- Webcamera and microphone (inbuilt or otherwise) are available.

- The combined observations from different modules are enough to reach to a conclusion.
- There are no dependencies that the project has on external factors.

4.2.2 General Constraints

- **Operating Environment:** The tool can run on any browser compatible with ReactJS v14.9 on any operating system environment.
- **Performance Requirements:** The system shall be interactive and the delays involved shall be set to a minimum. So, in every action-response of the system, there are no immediate delays. In case of submissions or saving sessions like saving tests, there is a delay of duration below 1 second.
- **Security Requirements:** Information transmission shall be securely transmitted to server/ database without any changes in information. Test creators and test takers use different functionalities, and have restricted access to file systems.
- **Availability:** The tool runs on a browser and is independent of network connections pertaining to the node system on which it is installed. It can be run on any compatible browser on any operating system environment.
- **Usability:** The tool is easy to handle and navigates in the most expected way with no delays. The built is responsive. The APIs and server handle requests swiftly.
- **User Interfaces:** The creator is expected to be familiar with the interface of the system. A simple GUI interface that has the same page for separate tasks, gives an enhanced test experience to both the user and the responder.
- **Hardware Interfaces:** Reliable software device drivers usually handled by the OS distributors are required to run the webcamera, keyboard, mouse, and microphone for recording responses. All devices compatible with modern computing systems are supported.
- **Software Interfaces:** The Psychometric Analyzer has different types of software interfaces (this term is used in a very broad meaning) to external packages, depending how the interaction is realized:
    i. User Interface: The various functionalities of the application can be accessed by the means of simple buttons, tabs and forms.
    ii. Message Interface: Since automation is the goal, methods have been implemented to run the various functionalities and communicate among different objects (instances of different classes) using Application Programming Interfaces (APIs).
    iii. Database Interface: The test creator can create and modify test and add, remove or modify images and questions-answers in the file system.

4.2.3 Goals and Guidelines

- Creating a tool or product that assists a human interviewer and/or psychologist and automates a large part of psychometric assessment.
- Classification into emotions or sentiment polarities and using some heuristic or guidelines to understand their significance.

- Combination of observations from different modules to reach to a conclusion.
- The efficiency of training of the model used for analysis depends on the CPU/GPU used and hence powerful ones are recommended.
- A properly working web camera and microphone are must for this to function appropriately.

4.2.4 Development Methods

Function-oriented design is considered for the design, wherein, each feature or utility of the tool is implemented as a function. The architectural model is client-server where in backend utilities are run by making calls using APIs. Responsive web design is used which an approach to web design that makes web pages render well on a variety of devices and window or screen sizes.

4.3 System Architecture

This section provides a high-level overview of how the functionality and responsibilities of the system are partitioned and then assigned to subsystems or components. The main purpose here is to gain a general understanding of how and why the system was decomposed, and how the individual parts work together to provide the desired functionality.
At the top-most level, the description of the major responsibilities that the software undertakes and how the higher-level components collaborate with each other in order to achieve the required results are given.
The overall architecture is based on the Client–server model- a distributed application structure that partitions tasks or workloads between the providers of a resource or service, called servers, and service requesters, called clients.

4.3.1 Facial Expression and Speech Emotion Recognition Models

The Facial Expression Recognition Model is a convolutional neural network model with a sequence of convolution layers, followed by a max-pooling layer, and a dense layer with nodes = 1024 and dropout rate = 0.5 succeeding all the convolution and pooling layers. Lastly, a dense layer for 7 outputs with softmax activation follows. This model is trained using the FER2013 dataset and saved for use. The face detection component is implemented using Haar-feature based cascade classifier [22] using cv2 (OpenCV). A rectangular frame is drawn around the detected part in real-time video feed from webcam. Detected sections of images with faces, captured periodically, are resized to 48x48 pixels, improved by increasing brightness and converted to grayscale. These grayscales are fed to the trained neural network model for classification of facial expressions into disgusted, angry, sad, happy, surprised,

fearful and neutral. The results of analysis are displayed on the user interface in results section.

This component records audio from microphone periodically and feeds it to the trained stacked LSTM model for classification of speech emotions into disgust, anger, sadness, happiness, surprise, fear and neutral.[20] The LSTM model is trained using the RAVDESS and SAVEE dataset.

4.3.2 Speech to Text for Confidence and Certainty Scoring

The speech recorded from microphone is converted to text. A function is created that cleans and analyses the text for words or phrases (n-grams) that reflect confidence level and certainty level and a score is calculated between -1 and +1 for each. Higher the confidence score, more the confidence. Higher the certainty score, higher the certainty in the interviewee's words.

4.3.3 Sentiment Analysis Model

This analysis requires the responder to input a description of a displayed image in about 100 words or some other specified limit. This text is then pre-processed by removing punctuations, extending contractions, POStagging for selecting sentiment descriptors, lemmatizing and stemming. From a similar pre-processing, followed by CountVectorizer and employing a self-designed variation of TF-IDF, a lexicon was generated. For the response as well, the lexicon is used for extracting numerical features. These features are fed to the trained sentiment analysis model (datasets: Stanford Movie Reviews, Amazon Reviews, Twitter Airline Sentiment) for estimation of the positive, negative and neutral perception percentages of an individual in response to the respective images. The results are displayed on the GUI window.

4.3.4 Adaptive Interview Chatbot

Chatbot is trained using .yml and .txt files containing conversations, and later consults sqlite database generated from the .yml and .txt files. The test creator's questions, answers, difficulties, and maximum scores from file system are used to generate a .json file. For each question, multiple similar answers are generated using the expected answer. Question Answering or Q/A system parses the .json file to select questions and match responses. A confidence value is used to decide how system responds. Questions are chosen based on question difficulty and user's performance.

4.3.5 Browser User Interface, Frontend and Backend

The frontend implementation uses ReactJS and NodeJS frameworks. There are different tabs for creating test, taking test and viewing results. All the functions including the clicks, navigation, inputs on the website are actions that are of importance in determining the behaviour, responses and updates made to the database and this is how they move forward in

their journey. Three-tier client server architecture is used because data is stored in database (File System) can be accessed and modified by the user with the help of the GUI between them. Frontend output is the response to any event or action performed on the website that can be seen in the form of routing to different pages, the change of questions or images on next and previous buttons, starting, submitting and moving through the phases of the test, etc. The backend implementation uses NodeJS and Django (Python) frameworks. The file system interaction is handled by the NodeJS, for example, saving the images in folders, question-answers in .json files in separate folders for each user for each test. The processing of inputs and their responses in terms of score is generated by the python server and returned to the frontend. All the inputs to the NodeJS server result in either publishing the data to files and folders or fetching it from there. The question-answers are saved as .json file, images are saved in the folder inside Creator→Test. The responses of the user in different phases are also saved in the .json files. These responses are sent to python server for evaluation and determination of score by using Machine Learning models or Chatbot system.

4.4 Detailed System Design

Most components described in the System Architecture section will require a more detailed discussion. Other lower-level components and subcomponents may need to be described as well. Each subsection of this section will refer to or contain a detailed description of a system software component.

4.4.1 Real-time Emotional State Determination

Model 1: FER model (CNN) trained using FER-2013 to 63% accuracy.
Emotion classes: angry, fearful, disgusted, happy, surprised, sad and neutral.
Model 2: SER model (LSTM) trained using RAVDESS to 83% accuracy.
Emotion classes: angry, fearful, disgusted, happy, surprised, sad and neutral.
Speech to text conversion is performed for calculation of the following scores:
Confidence Score: Score = (psum-nsum) / (psum+nsum) where psum and nsum are weighted sum of counts of high confidence and low confidence n-grams, respectively. Score lies between -1 and +1.
Certainty Score: Score = (psum-nsum) / (psum+nsum) where psum and nsum are weighted sum of counts of high certainty and low certainty (high uncertainty) n-grams, respectively. Score lies between -1 and +1.
Frontend: React modules: react-webcam and react-mic for video and audio feed.
Backend: The feed from frontend is sent to Django API which does the processing and returns the percentages of the 7 emotion classes.

Python libraries: speech_recogntion for speech to text conversion[40]. Libraries for FER model: keras, numpy, sklearn, cv2. Libraries used for SER: scipy for audio extraction and keras for training. Library used for processing text for scoring function: nltk.

4.4.2 Image Description-based Sentiment Analysis

Model: Sentiment classification model (ANN) trained using Amazon product reviews, Stanford movie reviews and Twitter Airline tweets to 93% accuracy. Sentiment classes: positive, neutral and negative.
Frontend: The images from the database (Node Server) are fetched from the file system and displayed. A text area where the description can be written.
Backend: The input from frontend is sent to the API (Django Server) for processing and it return the sentiment scores of three classes.
Libraries used: nltk, numpy, pandas, csv for pre-processing, and keras for model.

4.4.3 Adaptive Interview Chatbot

Model: Chatbot is trained using .txt and .yml files containing conversations under conversations as comma separated lists. A sqlite database is generated from these files. The chatbot matches input and searches responses from the sqlite database. The logic adapter uses a comparison function to match the user's input with the database statements and decides on the final response depending on a maximum similarity threshold. The creator's questions, answers, difficulties, and maximum scores from file system are used to generate a .json file. Question Answering or Q/A system parses the .json file to select questions and match responses. Both the chatbot and Q/A system run in the same scrollable chat and process user input. A confidence value is used to decide which system responds. Questions are chosen based on difficulty and user's performance on previous question.
Frontend: A conventional scrollable chat window
Backend: User's response is sent to server and bot's response is fetched and displayed.
Libraries used: chatterbot, nltk, thesaurus [32], [33].

4.4.4 Browser Interface Implementation

Landing Page: It is the first page which lands the visitor to discover the features and concept of the tool. There are three sections under three tabs for- (i) creating a test (naming the test, adding questions and images), (ii) taking a test (by selecting a test using its name and completing the three phases) and (iii) viewing report (results and analysis).
Test Phases:
Phase 1: For several questions/ prompts, the test taker's images will be periodically captured from webcamera and the audio recorded for emotion recognition using Facial Expression Recognition and Speech Emotion Recognition models. The recorded audio is also converted to text and confidence score and certainty score are calculated.

Phase 2: For all images, the user has to write a description in the given box. The sentiment classification model analyses the description and assign percentages to the sentiment classes.
Phase 3: Bot starts the conversation and asks questions. User responds in a message box below the conventional scrollable chat window. Response is submitted by pressing Enter key. After the interview, the user can talk with the bot and ask questions.

Report or Results/ Analysis: Each phase has its individual results and analyses section under the Results tab, as well as a combined result section. For each response, the maximum score for FER is 15, the maximum score for SER is also 15. Percentages of emotions are plotted as bar charts. The confidence and certainty scores range from -1 to +1. For sentiment analysis, it is simply the individual percentages for the three sentiments. Each question asked during chatbot interview is scored out of 10 multiplied by the difficulty level of the question. The question/ prompt/ image along with the response, score and brief analysis is displayed on the results section.

4.5 Use Case Diagrams

4.5.1 Functionality 1 (Create Test)

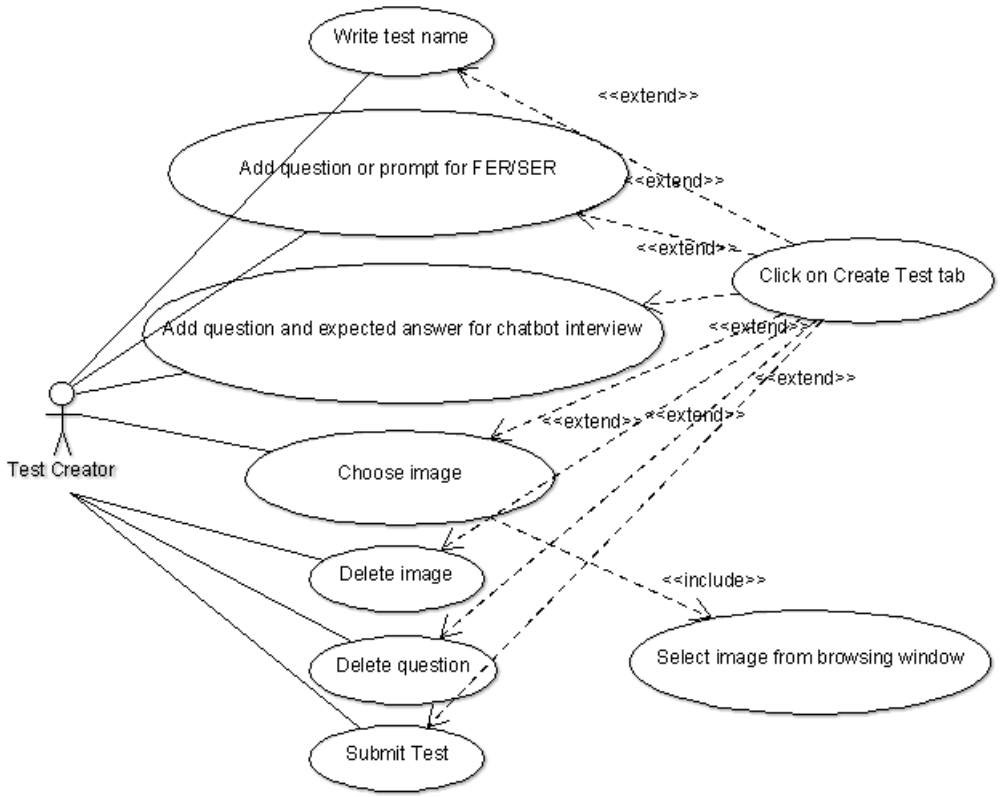


Figure 5

4.5.2 Functionality 2 (Test Phase 1 or FER/ SER Analysis and Confidence/ Certainty Score)

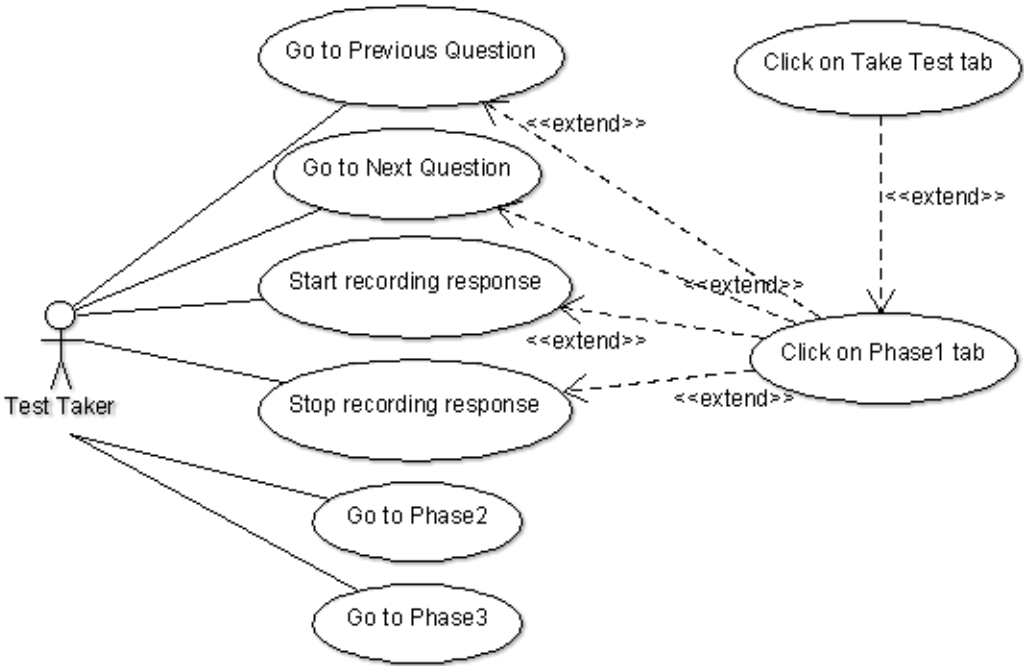


Figure 6

4.5.3 Functionality 3 (Test Phase 2 or Sentiment Analysis of Image-based Description)

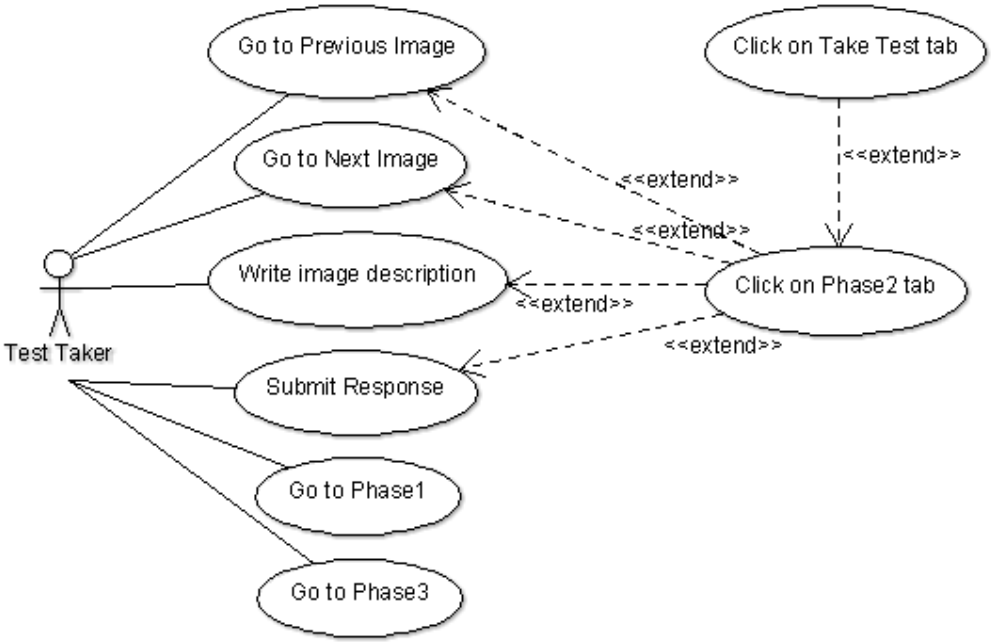


Figure 7

4.5.4 Functionality 4 (Test Phase 3 or Interview with Chatbot)

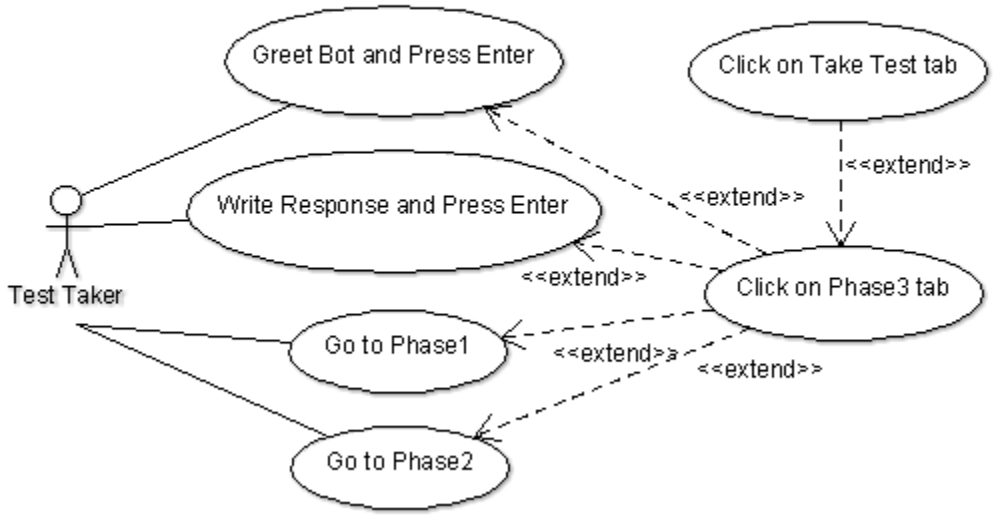


Figure 8

4.5.5 Functionality 5 (View Report or Results/ Analysis)

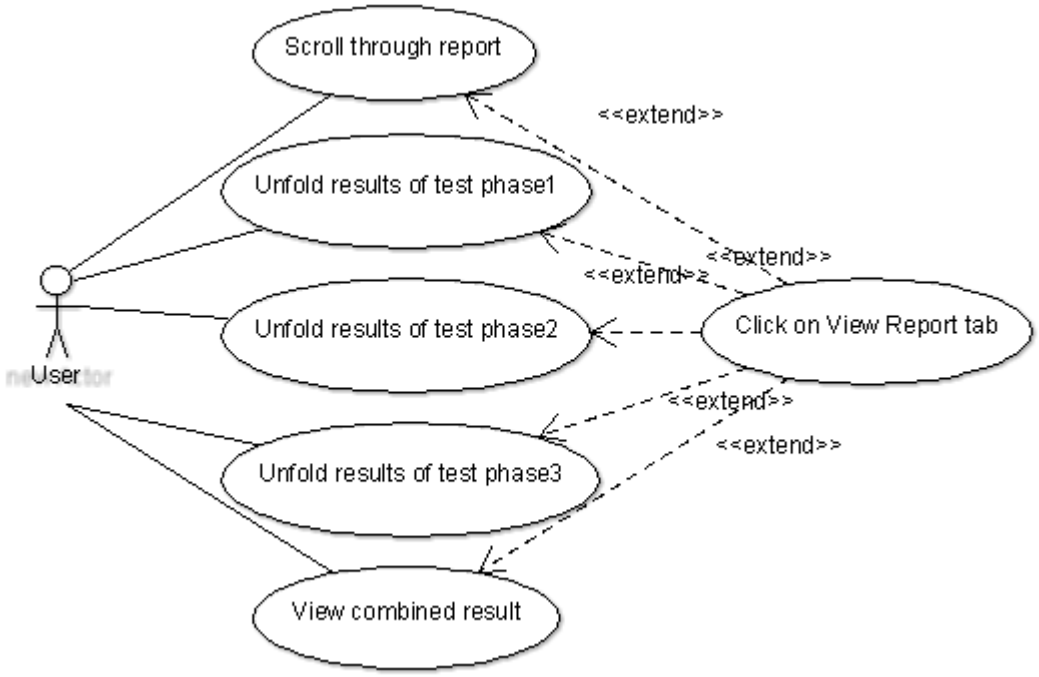


Figure 9

4.6 Dataflow Diagrams

4.6.1 Dataflow for Test Creator Interaction

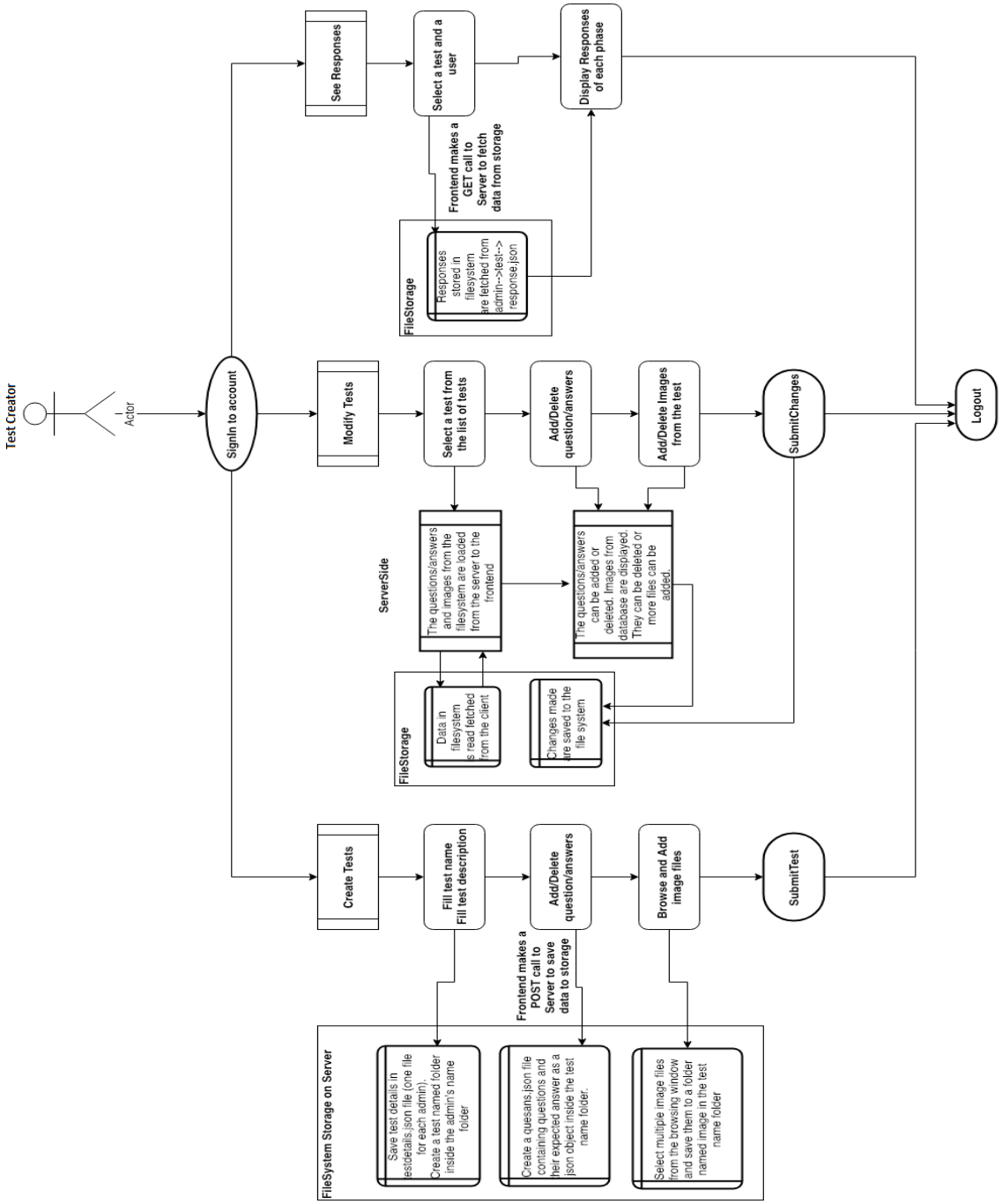


Figure 10

4.6.2 Dataflow for Test Taker Interaction

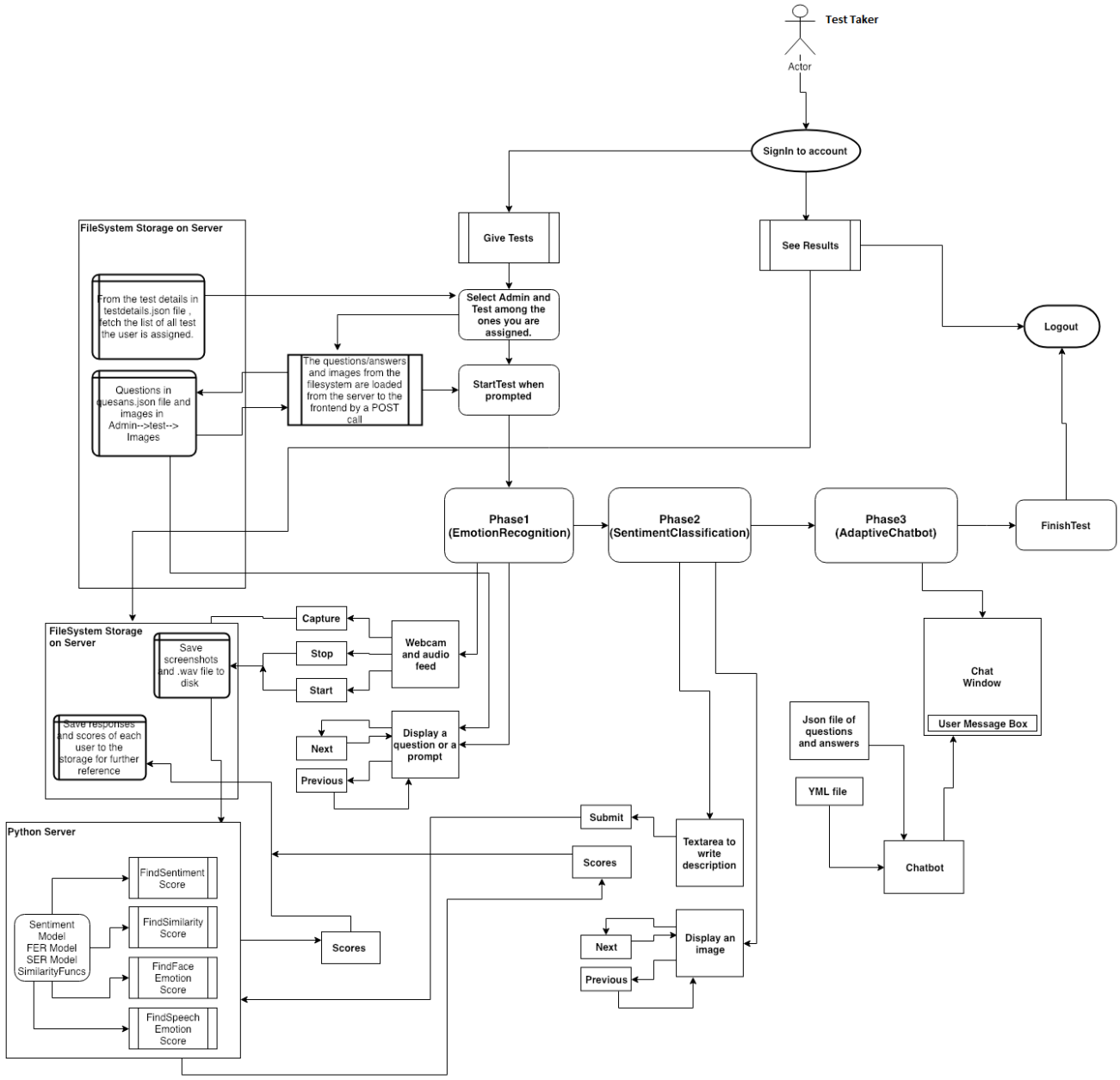


Figure 11

5. Implementation

5.1 Facial Expression Recognition Model

The images captured from webcam are classified using the Facial Expression Recognition or FER model. The FER model is trained using the publicly available FER-2013 dataset [24].

The dataset is challenging as the depicted faces vary significantly in terms of person age, face pose, and other factors, reflecting realistic conditions. The dataset is split into training, validation, and test sets with 28,709, 3,589, and 3,589 samples, respectively. Basic expression labels are provided for all samples. All images are grayscale and have a resolution of 48 by 48 pixels. The human accuracy on this dataset is around 65.5%.

The training has been performed on the images without any preprocessing in order to preserve the variations for realistic usage. The process for model design involved training the various models and different hyperparameters combinations on training set of 28709 samples, and testing on the validation set (3589 samples). The final performance data was gathered by training on the combined training and validation data and testing on the test dataset. The model with the highest accuracy (~63%) was saved for use. The model consists of a sequence of convolution and subsampling (max-pooling) layers, as shown in Figure 12, with subsequently increasing filters for better feature representation using more feature maps. The last of the feature maps are flattened to a vector which is passed to the dense layer (1024 nodes), followed by an output layer with softmax activation.[5], [6]

The model was built using keras library's Sequential model [19] and trained on Google Colaboratory's GPU (1xTesla K80, compute 3.7, having 2496 CUDA cores, 12GB GDDR5 VRAM).

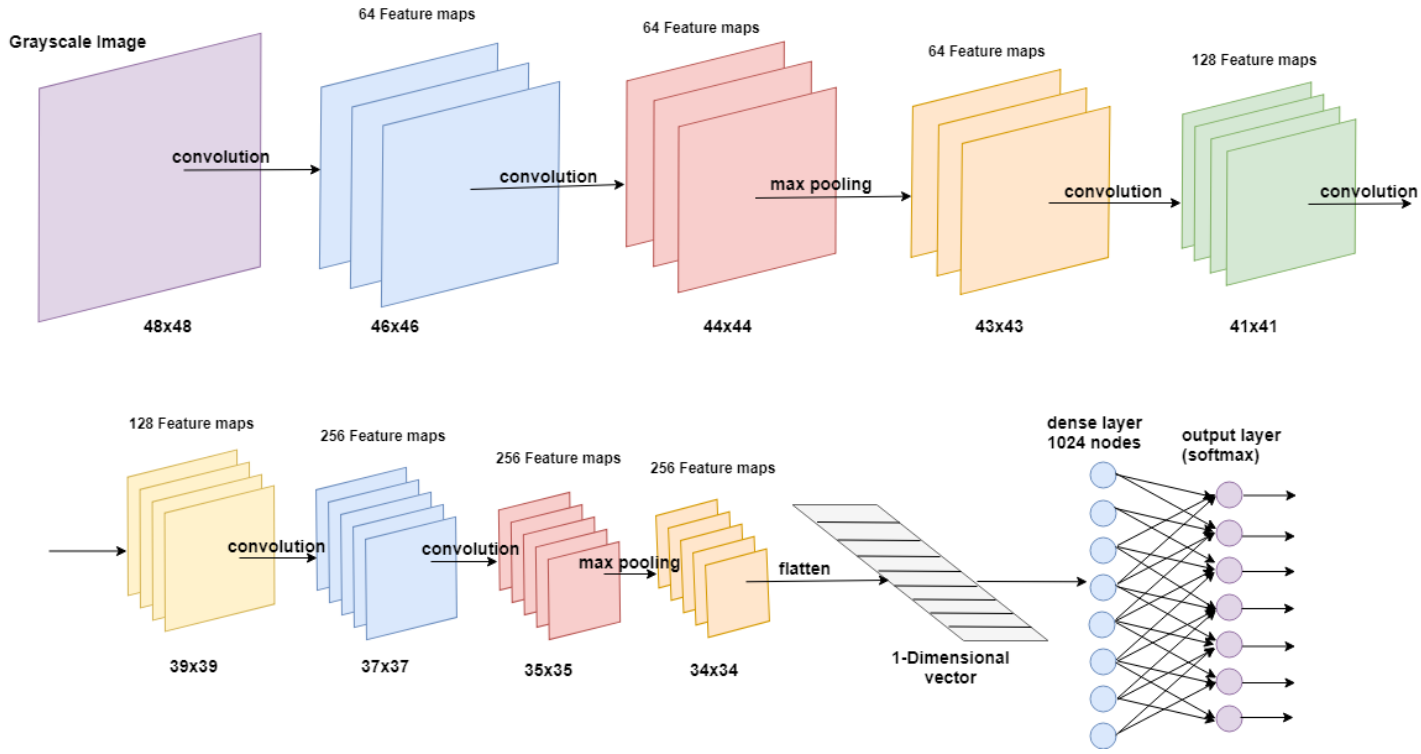


Figure 12

Training and testing accuracy vs epoch and loss vs epoch plots for batch_size = 256 are given in Figure 13 and 14 respectively. The confusion matrix for the same is shown in Figure 15.

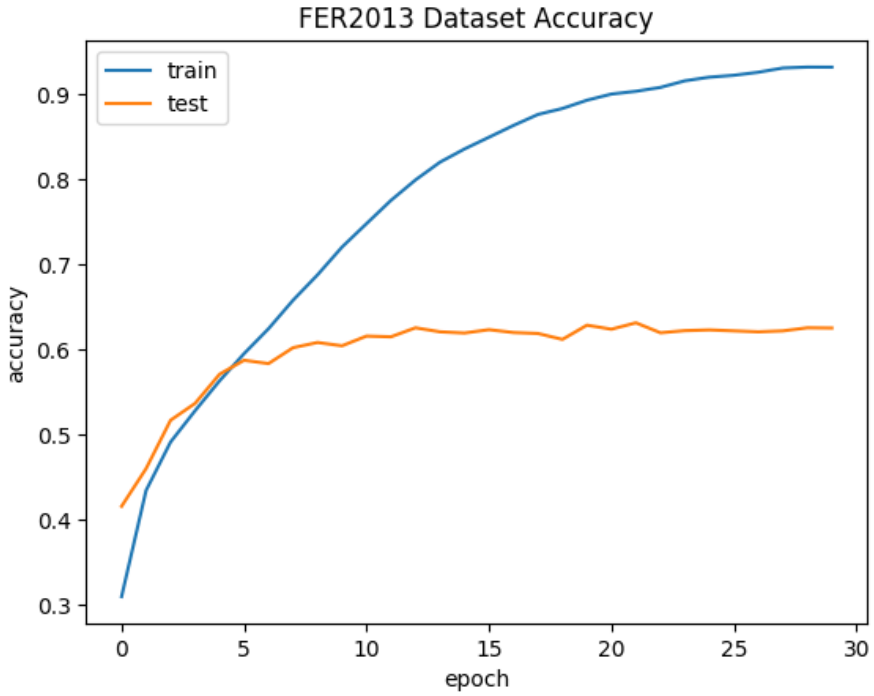


Figure 13

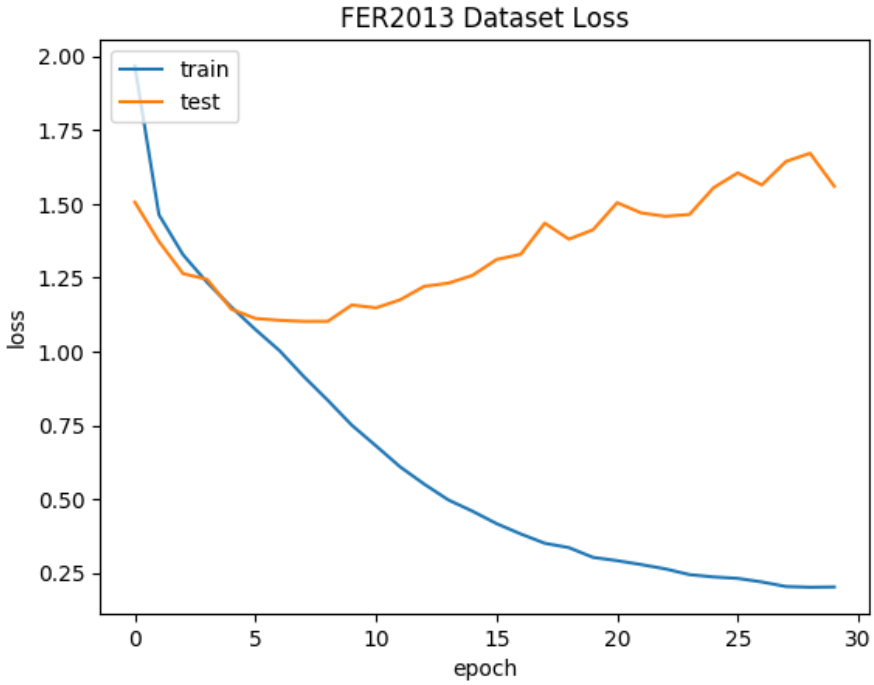


Figure 14

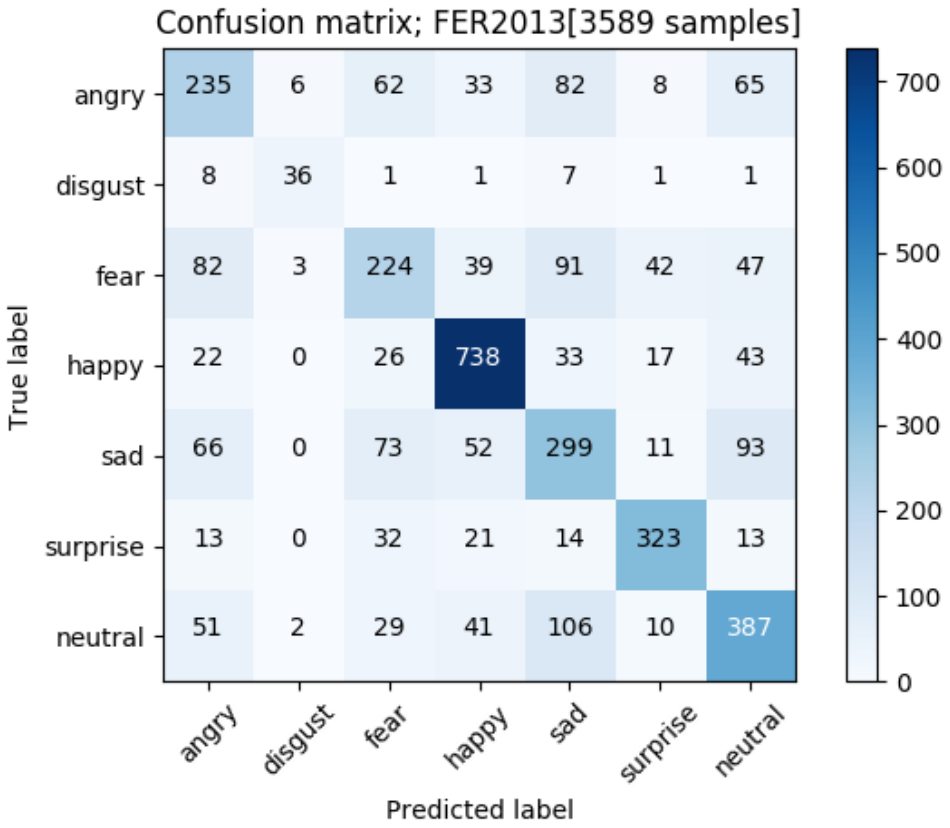


Figure 15

For using the model, preprocessing is required. Preprocessing entails operations that are applied once to each image. This includes face detection, smoothing and means for correcting for illumination variations. The libraries used are keras.preprocessing, cv2 (OpenCV), and PIL.

5.2 Speech Emotion Recognition Model

The Speech Emotion Recognition or SER model classifies the segments of a recording of a speaker (extracted using pyaudio [23]) into various categories of emotions. The SER model is trained on Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS). This dataset contains the complete set of 7356 RAVDESS files (total size: 24.8 GB). 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.

The other dataset used for training was Surrey Audio-Visual Expressed Emotion (SAVEE). The SAVEE database was recorded by four native English male speakers (identified as DC, JE, JK, KL), aged from 27 to 31 years. Emotion has been described in discrete categories: anger, disgust, fear, happiness, sadness, surprise and neutral. The text material consisted of 15 TIMIT sentences per emotion: 3 common, 2 emotion-specific and 10 generic sentences that were different for each emotion and phonetically-balanced. The 3

common and 2 × 6 = 12 emotion-specific sentences were recorded as neutral to give 30 neutral sentences. There is a total of 120 utterances per speaker- 480 utterances in all. [25], [26], [27].

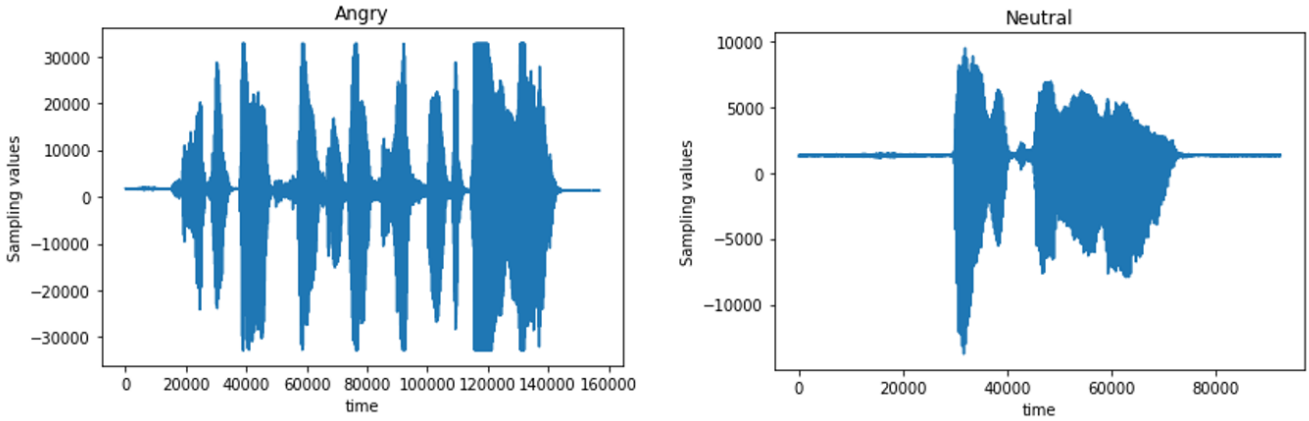


Figure 16 (contrasting the pressure values versus time plot for two samples labeled Angry and Neutral from SAVEE Dataset)

The SER model is implemented as a keras Sequential model [19] with a stack of three LSTM layers (256, 128 and 64 units respectively) followed by a dense layer and an output layer. The timesteps for an input sequence are set to 10.[10]

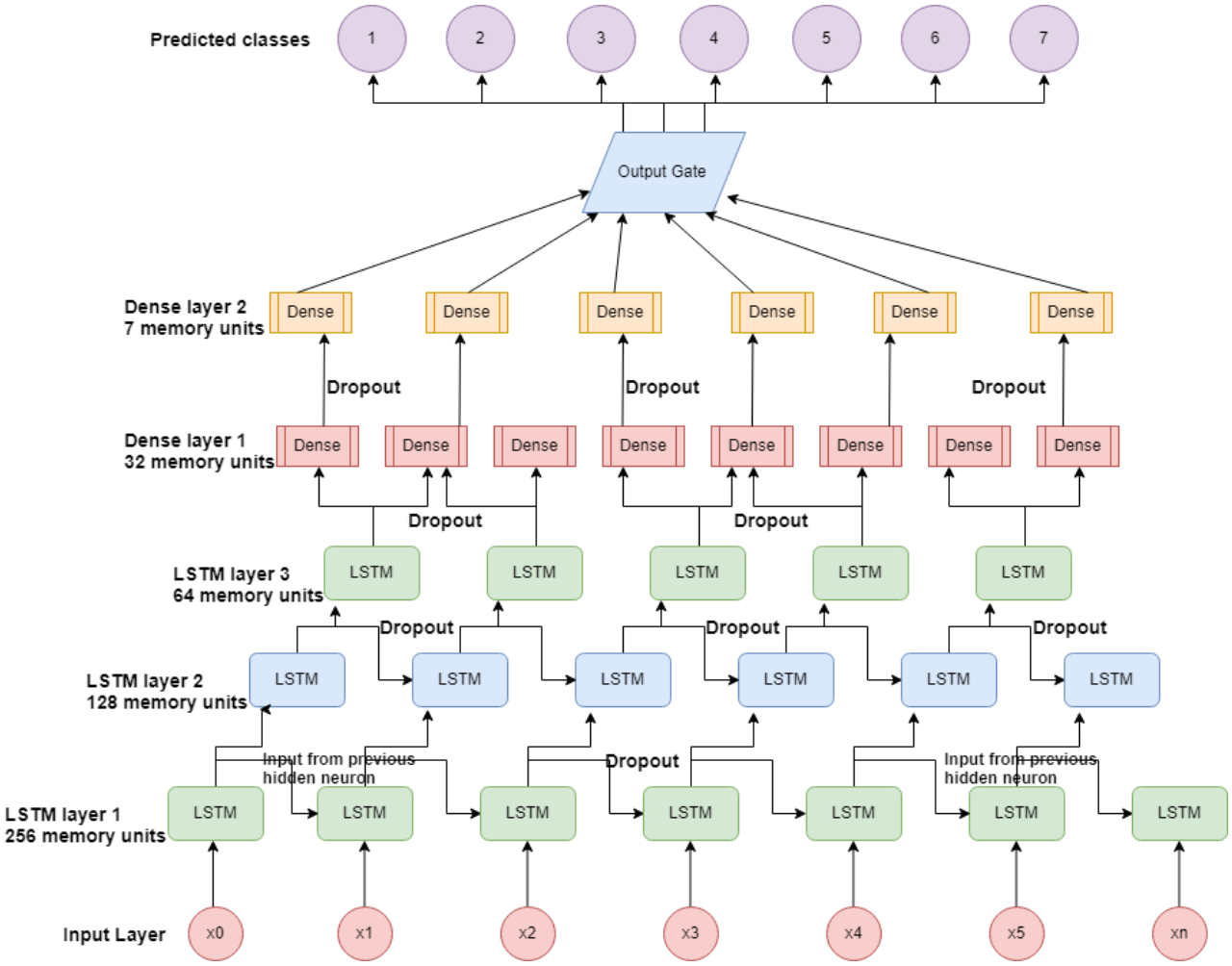


Figure 17

The SAVEE dataset has been augmented by doubling the samples. The training and testing accuracy vs epoch and loss vs epoch plots for RAVDESS Songs dataset are shown in Figure 18. The training and testing accuracy vs epoch and loss vs epoch plots for SAVEE dataset are shown in Figure 19. Figure 20 shows the confusion matrix for the entire dataset on RAVDESS Songs dataset, containing 1012 samples. Figure 21 shows the confusion matrix for augmented set on SAVEE dataset, with 960 samples.

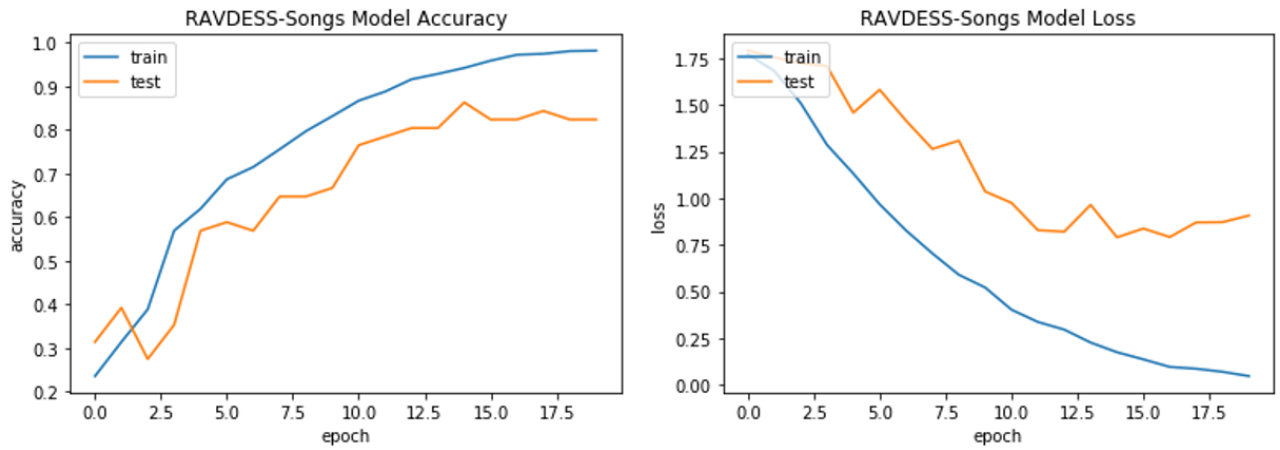


Figure 18

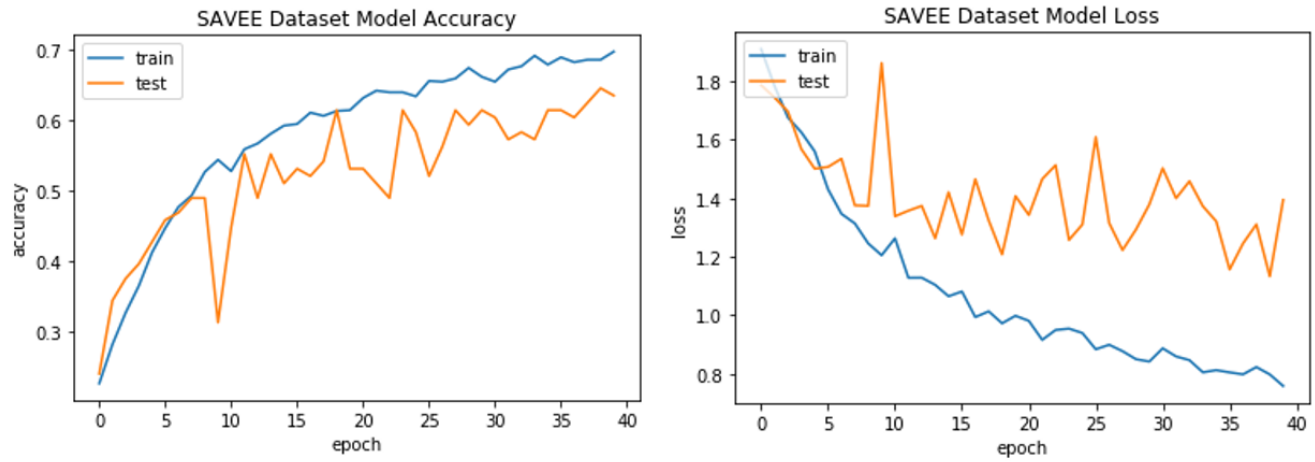


Figure 19

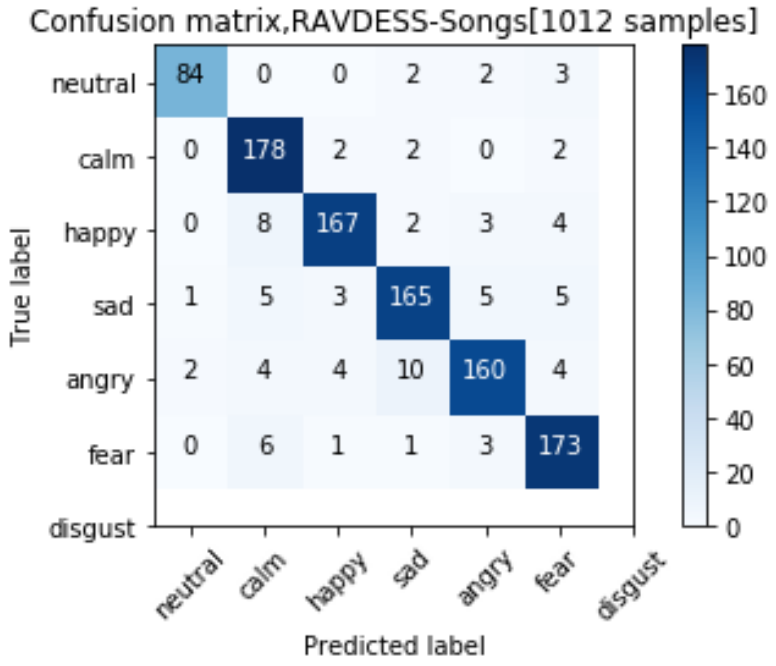


Figure 20

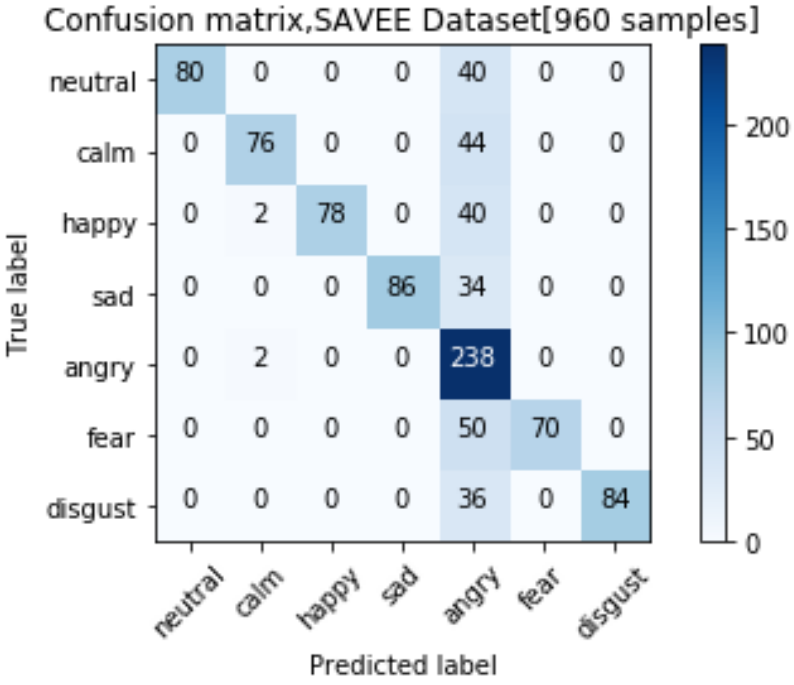


Figure 21

5.3 Confidence Score and Certainty Score Function

The text that is converted from speech, i.e., verbal response is preprocessed (conversion to lower case, expansion of contractions (e.g., I've → I have), removal of punctuations and other symbols). The function that calculates the score generates n-grams (n = 1, 2, and 3). There are 12 sets containing 1-grams, 2-grams and 3-grams reflecting high confidence (conf1, conf2, conf3), low confidence (unconf1, unconf2, unconf3), high certainty (cert1, cert2, cert3), and low certainty (uncert1, uncert2, uncert3). Words that depict confidence levels are usually adjectives, adverbs, nouns and verbs (e.g., assure, conclude, without a doubt, sneaking suspicion, etc.) whereas words that reflect certainty levels are usually modals, helping verbs and adverbs (e.g., might, maybe, definitely, surely, etc.). The pseudocode for the function is as follows:

```
function calculate_score(text):
    preprocess(text)
    grams1, grams2, grams3 = split_into_ngrams(text)

    p1 = length(grams1 intersection conf1)
    p2 = length(grams2 intersection conf2)
    p3 = length(grams3 intersection conf3)
    psum = p1 + 2*p2 + 3*p3
    n1 = length(grams1 intersection unconf1)
    n2 = length(grams2 intersection unconf2)
    n3 = length(grams3 intersection unconf3)
    nsum = n1 + 2*n2 + 3*n3
    if (psum+nsum)!=0:
        confidence_score = (psum-nsum)/(psum+nsum)
    else:
        confidence_score = 0

    p1 = length(grams1 intersection cert1)
    p2 = length(grams2 intersection cert2)
    p3 = length(grams3 intersection cert3)
    psum = p1 + 2*p2 + 3*p3
    n1 = length(grams1 intersection uncert1)
```

```
    n2 = length(grams2 intersection uncert2)
    n3 = length(grams3 intersection uncert3)
    nsum = n1 + 2*n2 + 3*n3
    if (psum+nsum)!=0:
        certainty_score = (psum-nsum)/(psum+nsum)
    else:
        certainty_score = 0

    return (confidence_score, certainty_score)
```

Both confidence and certainty scores lie between -1 and 1, and are weighted according to the value of n in n-gram. Higher the score, better the confidence or uncertainty.

5.4 Sentiment Classification Model

The Sentiment Classification model's lexicon is built using Amazon review dataset, Twitter Airline Sentiment Dataset and Stanford Movie Review dataset [28] [29] [30]. The model is trained on the combined text and label inputs from the three datasets. There are 36003 samples in training and 6354 samples in validation set. After preprocessing (according to the steps given in Figure 22) [11], [12], a file containing the occurrence count of words in each class is generated. Following that, a lexicon file is generated with each word having 3 representative values for each class (rv(w, X), where w is word and X is class), as calculated by:

```
for each word w:

  if (count of w in class X = 0):
       rv(w, X) = 0
  else if (count of w in class X = total count of w in all
  classes):
    rv(w, X) = 2.5 * count of w in class X
  else:
    rv(w, X) = 1/log10(total count of w in all classes/ count of w
  in class X)
```

Figure 22

For each sample, the words in the text are looked up in the lexicon files and their corresponding rv(w, X) values are summed to get a three-valued final feature vector, used as input to the model. The model consists of an input layer followed by a dense layer with 6 nodes and ReLU activation and a softmax output layer (Figure 23). The classes are negative, neutral and positive. The validation accuracy of the model is 76.24%. [13], [14]

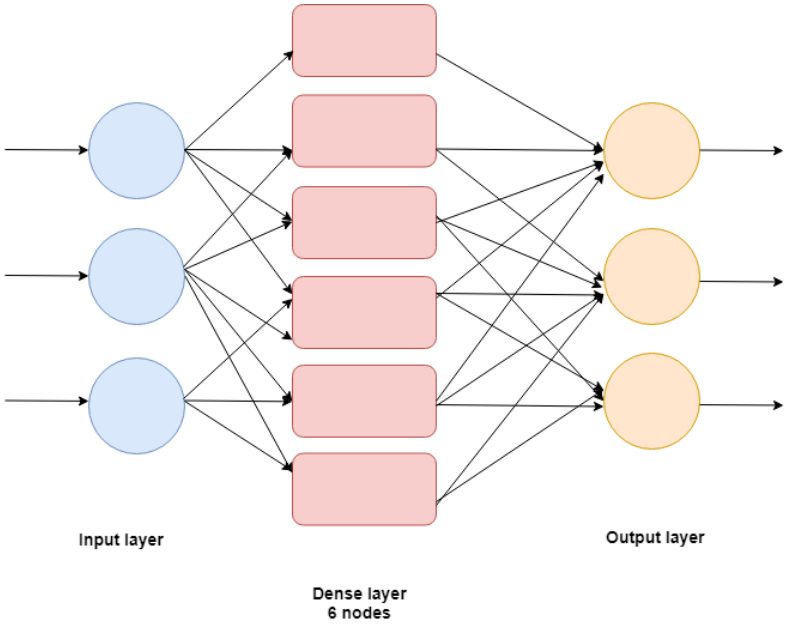


Figure 23

For batch size of 256 and 40 epochs, the training and validation accuracies and losses are plotted in Figure 24 and 25, respectively. The confusion matrix for the same is shown in Figure 26.

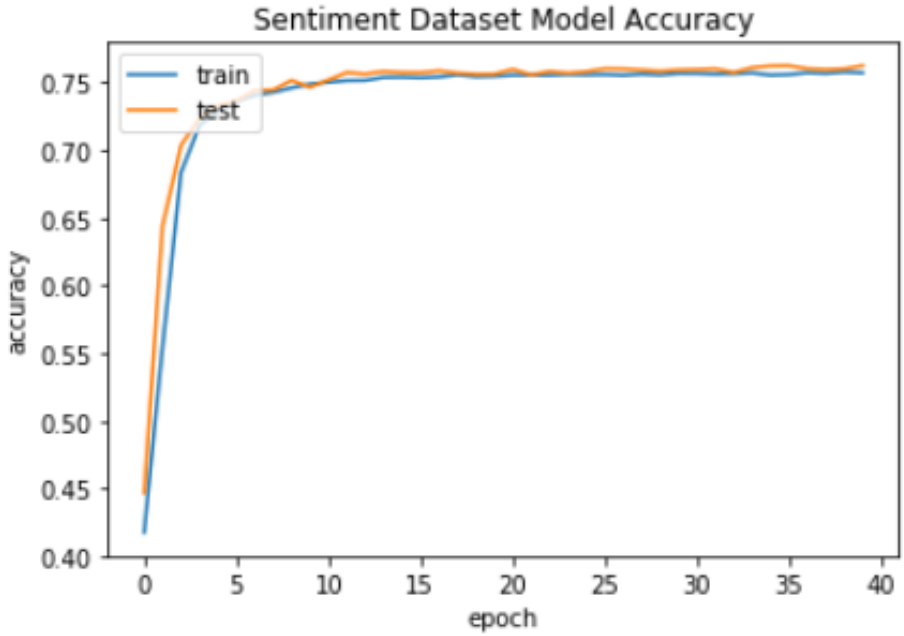


Figure 24

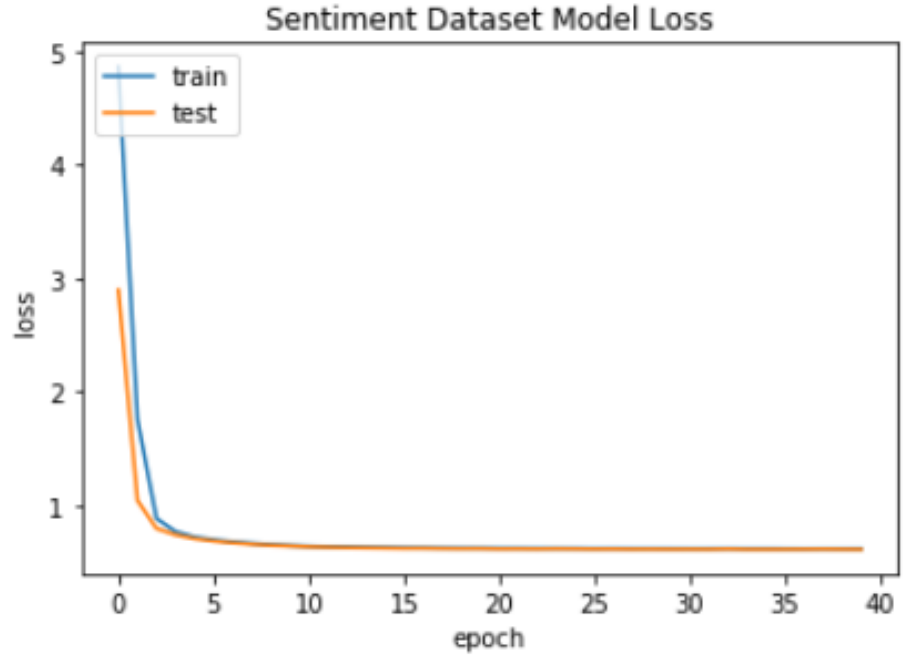


Figure 25

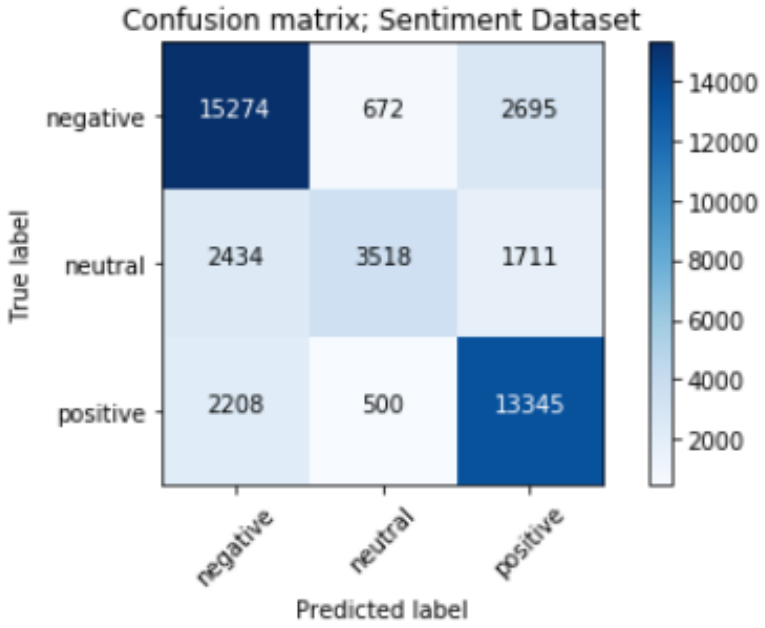


Figure 26

5.5 Adaptive Interview Chatbot

5.5.1 Chatbot Implementation

The chatbot has been implemented using the python library, chatterbot. An untrained instance of chatbot starts off with no knowledge of how to communicate. The chatbot is trained using multiple lists of general conversations and small talk from .yml or .txt files. The text in these files has been preprocessed by removing all characters aside from numeric, alphabet and whitespace, expanding word contractions and conversion to lowercase. After the training, a SQLite database is generated. The logic adapters in the program select the closest matching response by searching for the closest matching known statement that matches the input from the database using a similarity criteria (Levenshtein distance), and then choose a response from the selection of known responses to that statements based on a maximum similarity threshold.[32], [34]

The Question Answering or QA system or segment parses questions from the .json file generated when the test creator created the test. It is the ‘adaptive interview’ part as the interview adapts according to the responses of the test taker. If the response by the user scores high, a question with higher difficulty but better score is selected next, otherwise, a question with lower difficulty and low score is selected. The QA segment may be set up to run either parallel with general conversation chatbot instance or by interrupting the chatbot instance. After the interview, the chatbot can be used for conversation and asking questions on specific domains on which the chatbot instance has been trained.

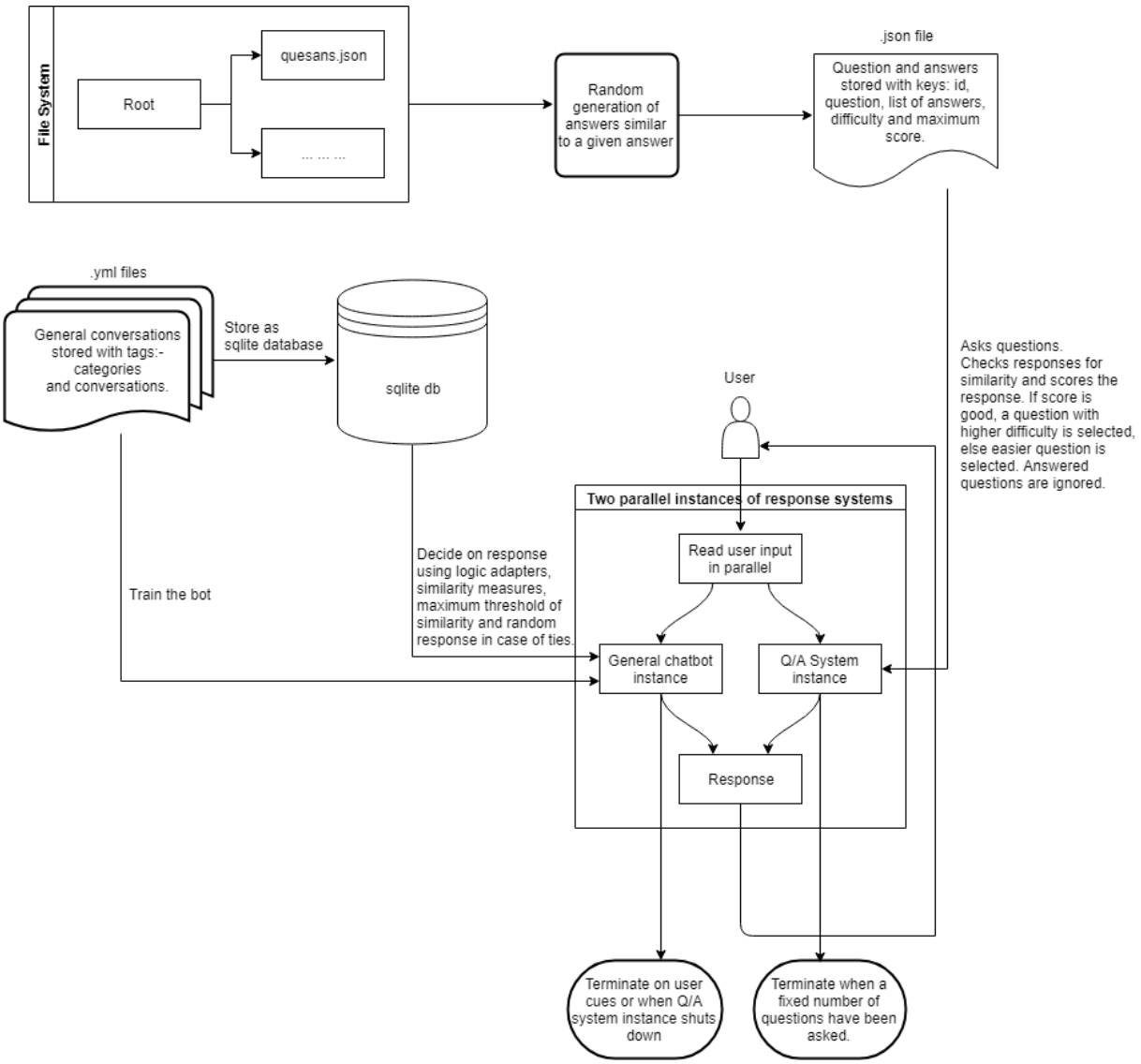


Figure 27

5.5.2 Adaptive Selection of Questions for QA Segment

The QA segment parses .json file from the backend file system for questions and the respective difficulty and expected responses. A python list, ques_list is created. It contains lists with three values- question identifier, its difficulty as a positive integer, and a boolean to indicate whether the question has been asked or not. This list is sorted in increasing order of difficulty. Two dictionaries (ques_dict and expected) are created with question identifier as the key and question and expected response as the values in the two dictionaries respectively. The function for choosing question according to the response score is given below. The argument highscore is a boolean which is set True if the response to the previous question scores high, and False

otherwise. The argument n is the number of questions remaining to be asked. The argument index refers to the index of the question in the ques_list.

```
function ask_question(highscore, n, index):
  if n == max_num_of_ques_to_be_asked:
     # select question of median difficulty
     index = ceil(len(ques_list)/2)
     ques_list[index][2] = False
     return ques_dict[ques_list[index][0]], index
  else if n > 0:
     flag = False
     if highscore:
        while index<=len(ques_list)-1 and ques_list[index][2]!=True:
             index += 1
             flag = True
             if index>len(ques_list)-1:
                 flag=False
                 index-=1
                 break
        if not flag:
             while index>=0 and ques_list[index][2]!=True:
                 index -= 1
      else:
         while index>=0 and ques_list[index][2]!=True:
             index -= 1
             flag = True
             if index<0:
                 flag=False
                 index+=1
                 break
         if not flag:
             while index<len(ques_list) and ques_list[index][2]!=True:
```

```
                    index += 1
        ques_list[index][2] = False
        return ques_dict[ques_list[index][0]], index
    else:
        return "END", -1
```

5.5.3 Scoring Function based on Similarity for QA Segment

In the question/answer functionality, the interviewee's response to a question is checked against the answer in the database, and using some similarity measures[15] and scaling, a score out of 10*difficulty is assigned to the interviewee's response. The interpretation of the score is entirely subjective, however, and requires the interviewer's understanding of the various ways of answering the question. This also implies the condition of the question being limited in terms of interpretations and kinds of responses it can generate.

The two answers are treated as two separate strings. Each string is tokenized using nltk library's TreebankWordTokenizer and each token is converted to its stem using nltk library's Snowball Stemmer. The words are detokenized back into a string. For cosine similarity, a list of terms, L occurring in the two strings are created. Count vectors are created for each string, that is, each string is represented by a vector containing occurrence count for each term in the list of terms L. Thus, we have two vectors A and B. Given n terms, there are n values in each of A and B, written as A_i and B_i where $1 \leq i \leq n$. The cosine similarity is calculated as follows:

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}},$$

For cosine similarity considering only two grams, the above process is repeated using two consecutive terms as a single entity for count vector creation. For jaccard similarity, the count of words common to both strings is found and divided by the number of terms in list of terms L. [16], [17]

$$Jaccard\ Index = \frac{count\ of\ common\ terms}{total\ number\ of\ terms}$$

The above scores are all in range [0, 1]. While cosine similarity calculates a good enough score, Jaccard similarity is given some weight to reduce the score in case the sizes of the strings differ vastly despite there being many common terms. Cosine similarity for 2-grams is given more weight than its actual value. The pseudocode for calculation of the final score out of 10*difficulty is given as follows:

```
function score_answer(response, expected_response, difficulty):
     cosine1 = cosine_similarity_1gram(response, expected_response)
     cosine2 = cosine_similarity_2gram(response, expected_response)
     jaccard = jaccard_similarity(response, expected_response)
     score = (cosine1 * 6) + (jaccard * 2.4)
     if cosine2 < 0.2:
            score += cosine2
     else if cosine2 < 0.3:
            score += 0.4
     else if cosine2 < 0.5:
            score += 0.8
     else if cosine2 < 0.7:
            score += 1
     else:
            score += 1.6
     return ceiling(score) * difficulty
```

5.6 Browser Interface, Backend and Database

The frontend implementation uses ReactJS and NodeJS frameworks. Navigation on a page works through tabs for creating test, taking test and seeing responses and viewing report (results/analysis) on navigation bar. All the functions including the clicks, navigation, inputs on the website are actions that are of importance in determining the behavior, responses and updates made to the database and this is how they move forward in their journey. The backend implementation uses NodeJS and Django (Python) frameworks. The file system interaction is handled by the NodeJS, for example, saving the images in folders, question-answers in .json files in separate folders for each user for each test. The processing of inputs and their responses in terms of score is generated by the python server and returned to the frontend. [35], [36], [37]

Three-tier client server architecture is used because data is stored in database (File System) can be accessed and modified by the test creator and the test taker with the help of the GUI between them. Frontend output is the response to any event or action performed on the website that can be seen in the form of routing to different pages, the change of questions or images on next and previous buttons, starting, submitting and moving through the phases of the test, etc. All the inputs to the NodeJS server result in either publishing the data to files and folders or fetching it from there. The question-answers are saved as .json file, images are saved in the folder inside

Creator→Test. The responses of the user in different phases are also saved in the .json files. The picture information is saved in the form of BLOB. These responses are sent to python server for evaluation and determination of score by using Machine Learning models or Chatbot system.

6. Results

The **FER model** trained on FER-2013 dataset shows 63% accuracy on validation set. The model was tested on a few images outside of the dataset. The images and results are given in Figures 28 to 31.

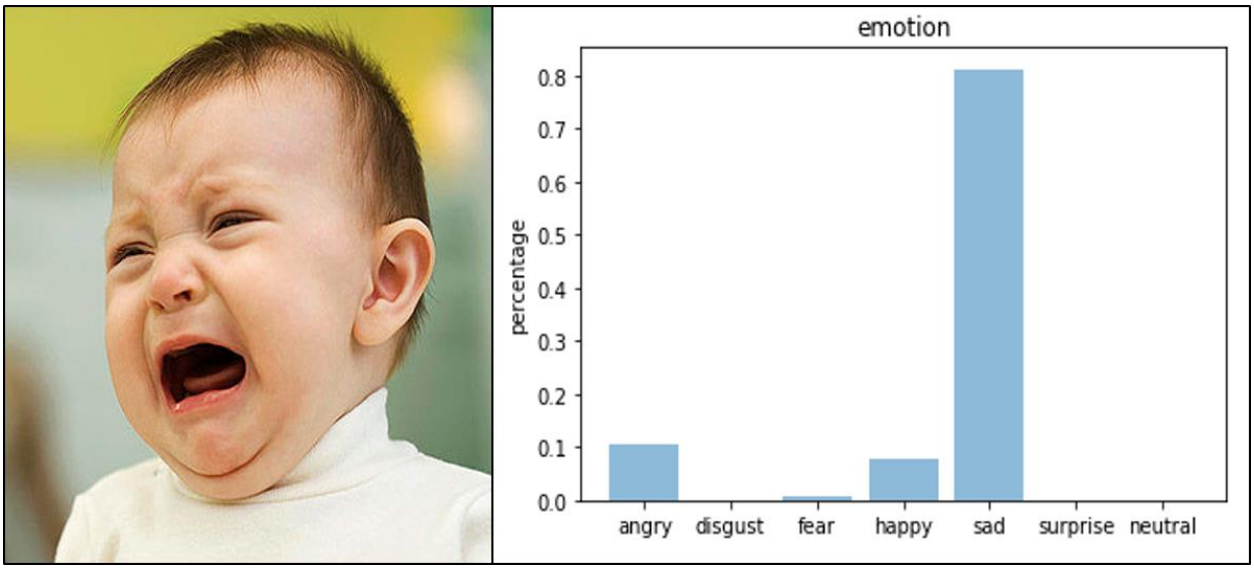


Figure 28

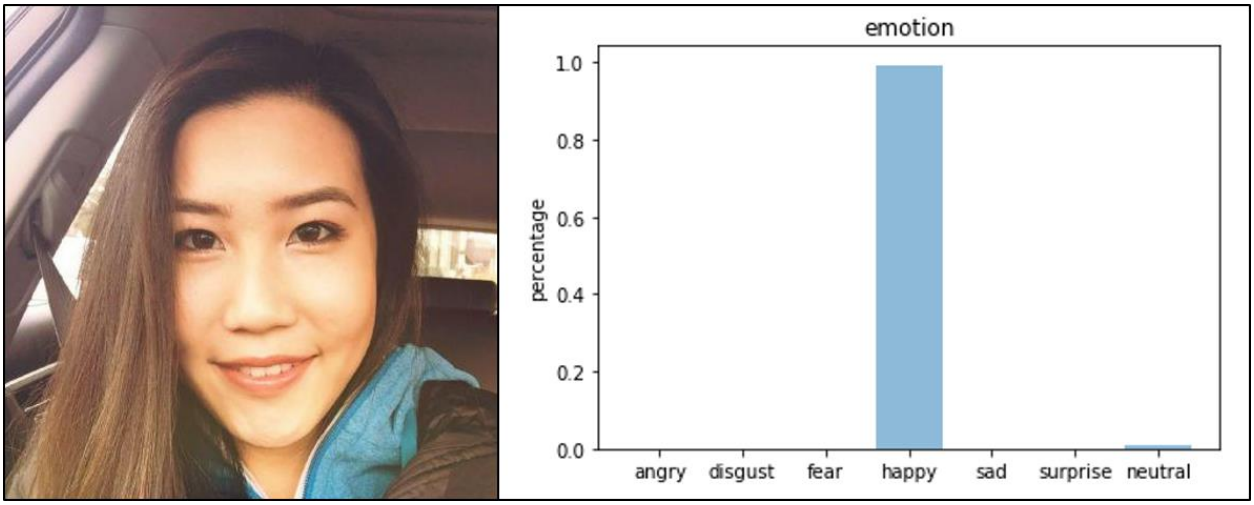


Figure 29

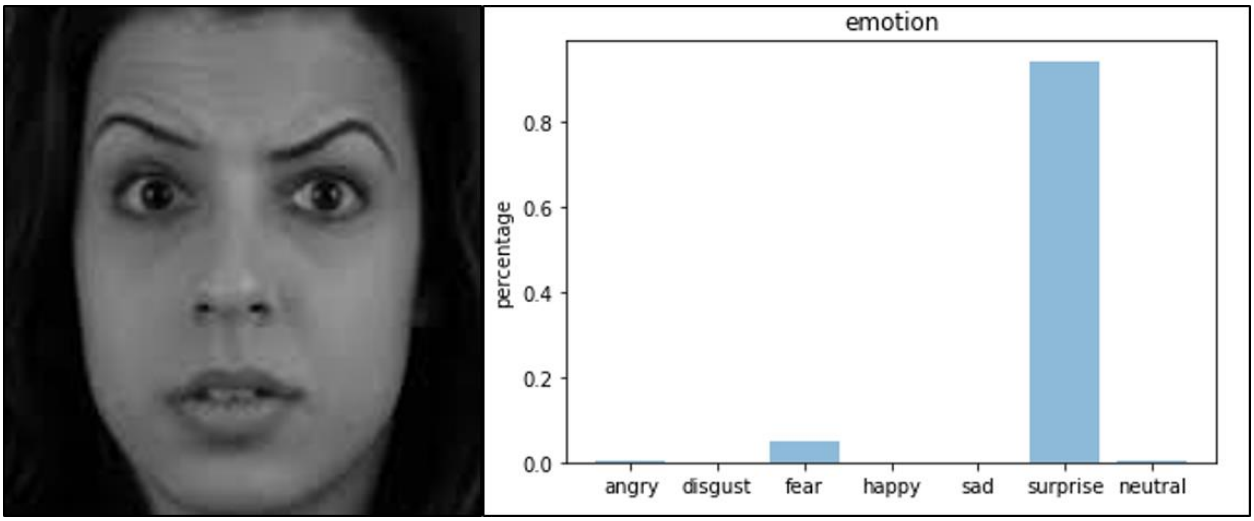


Figure 30

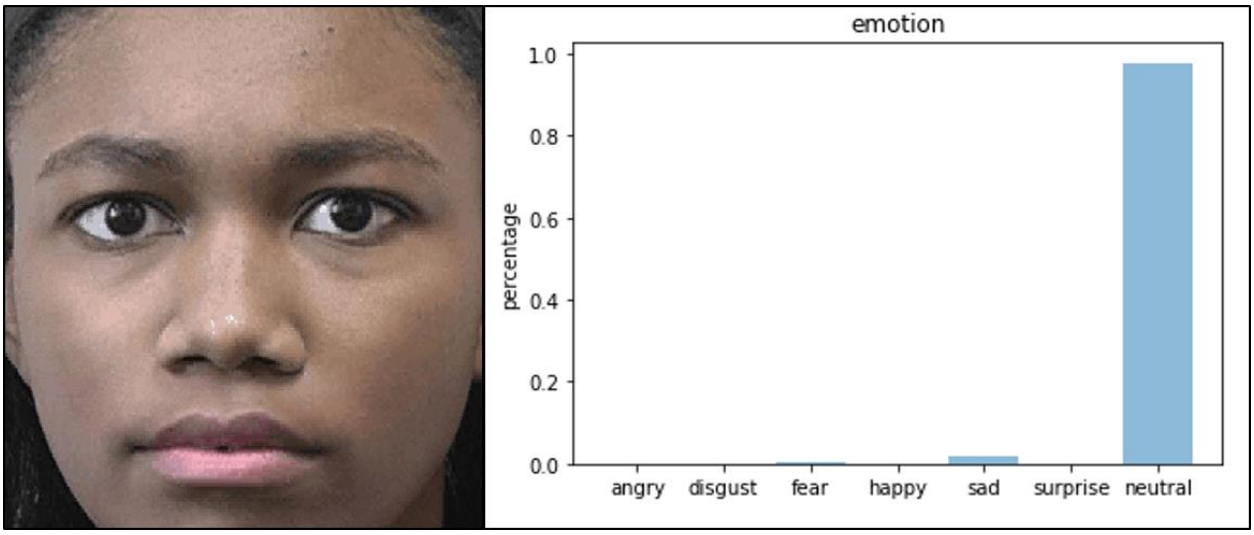
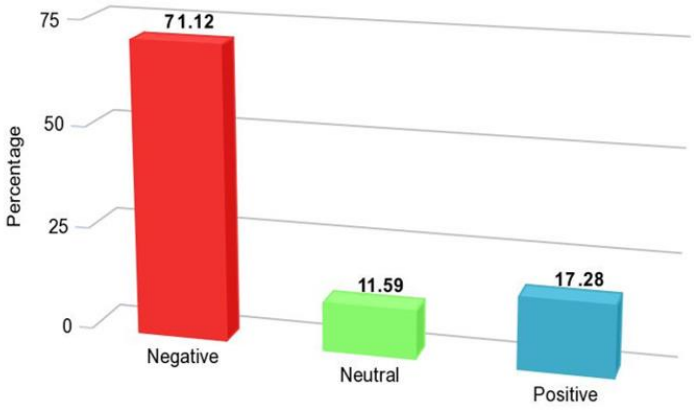


Figure 31

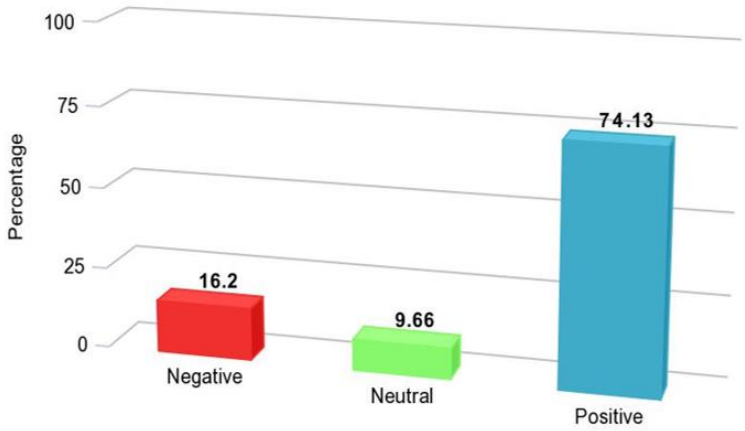
The **SER model** shows validation accuracy of 82.35% on RAVDESS Emotional Songs and 63.54% on SAVEE dataset.

The **Sentiment Classification model's** lexicon is built using Amazon review dataset, Twitter Airline Sentiment Dataset and Stanford Movie Review dataset. The validation accuracy is 76.24%. Figure 32 and 33 show the result on two unseen text inputs (not in dataset).



Text example 1
The lion is cowering before the hyenas, tired, weak and afraid.

Figure 32



Text example 2
A cheerful bird, sitting in the rain and enjoying every minute of it.

Figure 33

Test cases for **confidence score and certainty score determination**:
Output format: (confidence_score, certainty_score)

```
calculate_score("I am absolutely sure this works!")
>>> (1.0, 1.0)
calculate_score("We are not so sure about this")
>>> (-0.14285714285714285, -0.2)
```

Test case for **scoring answers using similarity measures**:

```
score_answer(["AI has always been the friend of all humans.", "AI an
d humans have always been friendly"], 5)
>>> 35
```

The score is 35/50 or 0.7 on similarity scale. Any score above 0.5 is a very good score.

Overall, the psychometric analysis tool functions fairly accurately in order to determine a person's emotional state and polarity of attitude. This tool can be used for psychological evaluation by recruiters, psychologists and people preparing for interviews.

There is great scope for research in the field as more and more testing agencies are switching to online statistical and psychometric analysis tools for standardized testing and personality and skill evaluation for certain jobs. For example, a line of research at NTA relevant to the use of psychometrics in high stake tests will promote new and improved psychometric and statistical methods and capabilities through innovation and development of foundational knowledge [41].

7. Future Work

The goals of the project were kept within what was believed to be attainable within the given timeframe and with the given resources and datasets. However, there is a significant scope for improvement and refinement by implementing certain changes and enhancements, which are listed as follows:

Registration and login: Currently, the tool can be used indiscriminately by any user. Registration and login functionality for test creator and test giver can be added for security and restricted use. Using an API, a server can be created to store the registration information (written in the form by users) in a database table.

Flexibility in setting tests and retaining results: Restrictions on the test could be imposed by fixing number and order of questions and images given to the user. Time constraint can be added

by fixing the response time per question. Multiple user responses can also be stored for pattern analysis.

Bag of Audio Words for Speech Emotion Recognition: Speech emotion recognition is a challenging task and the datasets are not available in bigger sizes or numbers. The bag-of-audio-words (BoAW) representations of acoustic low-level descriptors (LLDs) can been employed for greatly improving the performance on the classification of emotion task. [42]

Addition of more traditional psychometrics tests: More tests can be added based on Item Response Theory (IRT), Rasch Model or Likert Scale [43] for an extensive psychometric analysis. Apart from confidence and certainty score, more text based detections such as persuasive versus dismissive speech, amiable vs unfriendly word choice and inconsistency check can be added.

8. References

[1] National Council on Measurement in Education- http://www.ncme.org/ncme/NCME/ Resource_ Center/Glossary/NCME/Resource_Center/Glossary1.aspx?hkey=4bb87415-44dc-4088-9ed9-e8515326a061#anchorPArchived 2017-07-22 at the Wayback Machine.

[2] Koza, John R.; Bennett, Forrest H.; Andre, David; Keane, Martin A. (1996). Automated Design of Both the Topology and Sizing of Analog Electrical Circuits Using Genetic Programming. Artificial Intelligence in Design '96. Springer, Dordrecht. pp. 151–170

[3] Bishop, C.M. (2006), Pattern Recognition and Machine Learning, Springer, ISBN 978-0-387-31073-2 Zeiler, M.D. & Fergus, R.(2013). Visualizing and understanding convolutional networks. European Conference on Computer Vision(ECCV). 8689. 818-833.

[4] Jaswal, Deepika & Vishvanathan, Sowmya & Kp, Soman. (2014). Image Classification Using Convolutional Neural Networks. International Journal of Scientific and Engineering Research. 5. 1661-1668. 10.14299/ijser.2014.06.002.

[5] Huynh, Phung & Tran, Tien-Duc & Kim, Yong-Guk. (2016). Convolutional Neural Network Models for Facial Expression Recognition Using 3D-BUFE Database. 10.1007/978-981-10-0557-2_44.

[6] Zhang, Ting. (2018). Facial Expression Recognition Based on Deep Learning: A Survey. 345-352. 10.1007/978-3-319-69096-4_48.

[7] Schmidhuber, Juergen. (2014). Deep Learning in Neural Networks: An Overview. Neural Networks. 61. 10.1016/j.neunet.2014.09.003.

[8] Chernykh, Vladimir & Sterling, Grigoriy & Prihodko, Pavel. (2017). Emotion Recognition From Speech With Recurrent Neural Networks.

[9] Hochreiter, Sepp & Schmidhuber, Jürgen. (1997). Long Short-term Memory. Neural computation. 9. 1735-80. 10.1162/neco.1997.9.8.1735.

[10] Cen, Ling & Dong, Minghui & Li, Haizhou & Chan, Paul. (2010). Machine Learning Methods in the Application of Speech Emotion Recognition. 10.5772/8613.

[11] Desai, Mitali & Mehta, Mayuri. (2016). Techniques for sentiment analysis of Twitter data: A comprehensive survey. 149-154. 10.1109/CCAA.2016.7813707.

[12] Das, Bijoyan & Chakraborty, Sarit. (2018). An Improved Text Sentiment Classification Model Using TF-IDF and Next Word Negation.

[13] Effrosynidis, Dimitrios & Symeonidis, Symeon & Arampatzis, Avi. (2017). A Comparison of Pre-processing Techniques for Twitter Sentiment Analysis. 21st International Conference on Theory and Practice of Digital Libraries (TPDL 2017). 10.1007/978-3-319-67008-9_31.

[14] Yun-tao, Zhang & Ling, Gong & Yong-cheng, Wang. (2005). An improved TF-IDF approach for text classification. Journal of Zhejiang University - Science A: Applied Physics & Engineering. 6. 49-55. 10.1007/BF02842477.

[15] Khuat, Tung & Duc Hung, Nguyen & Thi My Hanh, Le. (2015). A Comparison of Algorithms used to measure the Similarity between two documents. International Journal of Advanced Research in Computer Engineering & Technology (IJARCET). 4. 1117-1121.

[16] Shoaib, Muhammad & Daud, Ali & Khiyal, Malik. (2014). An improved Similarity Measure for Text Documents.. Journal of Basic and Applied Scientific Research. 4. 215-223. 10.13140/2.1.4814.4006.

[17] Computer Engineering and Applications Vol. 5, No. 1, February 2016. Comparison Jaccard similarity, Cosine Similarity and Combined Both of the Data Clustering With Shared Nearest Neighbor Method Lisna Zahrotun.

[18] https://en.wikipedia.org/wiki/Levenshtein_distance

[19] https://keras.io/getting-started/sequential-model-guide/

[20] https://keras.io/layers/recurrent/

[21] https://www.sqlite.org/docs.html

[22] https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_objdetect/py_face_detection /py_face_detection.html

[23] https://people.csail.mit.edu/hubert/pyaudio/docs/

[24] https://www.kaggle.com/c/challenges-in-representation-learning-facial-expression-recognition -challenge/data

[25] https://tspace.library.utoronto.ca/handle/1807/24487

[26] https://smartlaboratory.org/ravdess

[27] http://kahlan.eps.surrey.ac.uk/savee/

[28] http://jmcauley.ucsd.edu/data/amazon/

[29] https://nlp.stanford.edu/sentiment/index.html

[30] https://www.kaggle.com/crowdflower/twitter-airline-sentiment

[31] Zhang, Lei & Wang, Shuai & Liu, Bing. (2018). Deep Learning for Sentiment Analysis : A Survey. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery. 10.1002/widm.1253.

[32] https://chatterbot.readthedocs.io/en/stable/

[33] https://pypi.org/project/thesaurus/

[34] P. Kataria, K. Rode, A. Jain, P. Dwivedi, S. Bhingarkar. User Adaptive Chatbot for Mitigating Depression. International Journal of Pure and Applied Mathematics. Volume 118 No. 16 2018, 349-361.

[35] https://reactjs.org/docs/getting-started.html

[36] https://nodejs.org/docs/latest-v8.x/api/

[37] https://docs.djangoproject.com/en/2.1/

[38] https://www.npmjs.com/package/react-webcam

[39] https://www.npmjs.com/package/react-mic

[40] https://pypi.org/project/SpeechRecognition/

[41] https://nta.ac.in/Research

[42] Schmitt, M., Ringeval, F., & Schuller, B.W. (2016). At the Border of Acoustics and Linguistics: Bag-of-Audio-Words for the Recognition of Emotions in Speech. INTERSPEECH.

[43] http://www.assessmentpsychology.com/psychometrics.htm