EMOTION RECOGNITION

A PROJECT REPORT

Submitted By

(SACHIN LAL)
(2100290140114)

(MUSKAN CHOUDHARY)
(2100290140092)

(VIDUSHI SHUKLA)
(2100290140145)

(AVNI TYAGI)
(2100290140042)

Submitted in partial fulfilment of the Requirements for the Degree of

MASTER OF COMPUTER APPLICATION

Under the Supervision of
Ms. Divya Singhal (Assistant Professor)

Submitted to
DEPARTMENT OF COMPUTER APPLICATIONS
KIET GROUP OF INSTITUTIONS, GHAZIABAD , DELHI-NCR UTTAR PRADESH-201206

(24 FEB 2023)

DECLARATION

I hereby declare that the work presented in this report entitled “EMOTION RECOGNITION”, was carried out by me and my team mates. I have not submitted the matter embodied in this report for the award of any other degree or diploma of any other University or Institute.

I have given due credit to the original authors/sources for all the words, ideas, diagrams, graphics, computer programs, experiments, results, that are not my original contribution. I have used quotation marks to identify verbatim sentences and given credit to the original authors/sources.

I affirm that no portion of my work is plagiarized, and the experiments and results reported in the report are not manipulated. In the event of a complaint of plagiarism and the manipulation of the experiments and results, I shall be fully responsible and answerable.

Name	: Sachin Lal (2100290140114)
	Muskan Choudhary (2100290140092)
	Vidushi Shukla (2100290140145)
	Avni Tyagi (2100290140042)
Branch	: Master of Computer Application

(Candidate Signature)

CERTIFICATE

Certified that **SACHIN LAL (2100290140114), MUSKAN CHOUDHARY (2100290140092), VIDUSHI SHUKLA (2100290140145), AVNI TYAGI (2100290140042)** have carried out the project work having **"EMOTION RECOGNITION"** for Master of Computer Applications from Dr. A.P.J. Abdul Kalam Technical University (AKTU) (formerly UPTU), Technical University, Lucknow under my supervision. The project report embodies original work, and studies are carried out by the student himself / herself and the contents of the project report do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution.

Date:

SACHIN LAL (2100290140114)
MUSKAN CHOUDHARY (2100290140092)
VIDUSHI SHUKLA (2100290140145)
AVNI TYAGI (2100290140092)

This is to certify that the above statement made by the candidate is correct to the best ofmy knowledge.

Date:

Ms. Divya Singhal
Assistant Professor
Department of Computer Applications
KIET Group of Institutions, Ghaziabad

Signature of Internal Examiner **Signature of External Examiner**

Dr. Arun Tripathi
Head, Department of Computer Applications
KIET Group of Institutions, Ghaziabad

ABSTRACT

Our Mood Recognition System identifies emotions to the best of its capabilities depending on the internet and the hardware implemented in the system. communication that very in complexity, intensity, and meaning. Purposed system depends upon human faces we know face also reflects the human brain activities or emotions.

The addition or absence of one or more facial actions may alter its interpretation. In addition, some facial expressions may have a similar gross morphology but indicate varied meaning for different expression intensities. In order to capture the subtlety of facial expression in non-verbal communication,

I will use an existing simulator which will be able to capture human emotions by reading or comparing mood expressions. This algorithm automatically extracts features and their motion information, discriminate subtly different facial expressions,and estimate expression intensity.

ACKNOWLEDGEMENT

Success in life is never attained single-handedly. My deepest gratitude goes to my project supervisor, **Ms. Divya Singhal,** for her guidance, help and encouragement throughout my research work. Their enlightening ideas, comments, and suggestions.

Words are not enough to express my gratitude to **Dr. Arun Kumar Tripathi**, Professor and Head, Department of Computer Applications, for his insightful comments and administrativehelp at various occasions.

Fortunately, I have many understanding friends, who have helped me a lot on many critical conditions.

Finally, my sincere thanks go to my family members and all those who have directly and indirectly provided me with moral support and other kind of help. Without their support, completion of this work would not have been possible in time. They keep my life filled with enjoyment and happiness.

SACHIN LAL

MUSKAN CHOUDHARY

VIDUSHI SHUKLA

AVNI TYAGI

TABLE OF CONTENTS

DECLARATION	ii
CERTIFICATE	iii
ABSTRACT	iv
ACKNOWLEDGEMENT	v
LIST OF TABLES	vi
1. CHAPTER: Introduction	1-4
1.1 Project Description	2
1.2 Project Scope	3
1.3 Hardware Requirements	4
2. CHAPTER: Literature Review	5-12
2.1 Facial Expression Recognition	6
2.2 Speech Emotion Recognition	8
2.3 Physiological Emotion Recognition	4
2.4 Multi modal Emotion Recognition	4
3. CHAPTER: Database Design	5-10
3.1 Flow Chart	5

3.2 Use Case Diagram 7

4. CHAPTER: Form Design 11-14

4.1 Data Set and Data Collection Module 11

4.2 Output Form (Screenshots) 13

5. CHAPTER: Coding 15-16

5.1 Input (Screenshot) 15

6. CHAPTER: Testing 17-22

6.1 Unit Testing 17

6.2 Integration Testing 18

6.3 Software verification and Validation 19

6.4 Black Box Testing 20

6.5 White Box Testing 21

6.6 System Testing 22

CHAPTER 1

INTRODUCTION

Emotion recognition is the process of identifying human emotion. People vary widely in their accuracy at recognizing the emotions of others. Use of technology to help people with emotion recognition is a relatively nascent research area. Generally, the technology works best if it uses multiple modalities in context. To date, the most work has been conducted on automating the recognition of facial expression from video, spoken expressions from audio, written expressions fromtext, and physiology asmeasured by wearables.

Emotion recognition is a technique used in software that allows a program to "read" the emotions on a human face using advanced image processing. Companies have been experimenting with combining sophisticated algorithms with image processing techniques that have emerged in the pastten years to understand more about what an image or a video of a person's face tells us about how he/she is feeling.

Human emotion recognition plays an important role in the interpersonal relationship. The automatic recognition of emotions has been an active research topic from early eras. Therefore, there are several advances made in this field. Emotions are reflected from speech, hand and gestures of thebody and throughfacial expressions. Hence extracting and understanding of emotion has a high importance of the interactionbetween human and machine communication. This paper describes the advances made in this field and the various approaches used for recognition of emotions. The main objective of the paper is to propose real time implementation of emotion recognition system.

Emotions govern our daily lives; they are a big part of the human experience, and inevitably they affect our decision -making. We tend to repeat actions that make us feel happy, but we avoid those that make us angry or sad.

Information spreads quickly via the Internet — a big part of it as text — and as we know, emotions tend to intensify if left undealt with.

Thanks to natural language processing, this subjective information can be extracted from written **sources**such as reviews, recommendations, publications on social media, transcribed conversations, etc., allowing us to understand the emotions expressed by the author of the text and therefore act accordingly.

1.1 PROJECT DESCRIPTION

AI can detect the emotions of a person through their facial expressions. Detected emotions can fall intoany of the six main data of emotions: happiness, sadness, fear, surprise, disgust, and anger.

Emotion detection is the process of identifying human emotions through various methods such as facial expressions, speech analysis, and physiological signals. The purpose of emotion detection is to improve human-computer interaction, understand customer feedback, and monitor mental health.

A typical emotion detection project involves the following steps:

- Data collection: Gather datasets of facial expressions, speech samples, or physiological signals that represent different emotions such as happiness, anger, sadness, fear, or surprise.

- Data preprocessing: Clean and prepare the data for analysis by removing noise, outliers, and missing values. Normalize the data to a common scale and divide it into training, validation, and testing sets.

- Feature extraction: Extract meaningful features from the data such as the intensity, duration, or frequency of certain facial muscles, speech patterns, or heart rate variability.

- Model selection: Choose a suitable machine learning algorithm such as Support Vector Machines, Random Forests, or Convolutional Neural Networks that can classify the input features into different emotional categories.

- Model training: Train the selected model on the training set using an appropriate optimization algorithm such as Gradient Descent, Stochastic Gradient Descent, or Adam.

- Model evaluation: Evaluate the performance of the trained model on the validation set by calculating various metrics such as accuracy, precision, recall, and F1-score.

- Model tuning: Adjust the hyperparameters of the model such as learning rate, regularization strength, or number of hidden layers to optimize the performance on the validation set.

- Model testing: Test the final model on the testing set to estimate its performance on unseen data.

- Deployment: Integrate the trained model into a software application or a web service that can receive input data such as images, audio, or biosignals and output the corresponding emotional category.

Overall, emotion detection projects require a combination of skills in data science, machine learning, signal processing, and software engineering.

1.2 PROJECT SCOPE

Emotion recognition has wide scope in many areas such as human computer interaction, biometric security etc. So it provides insight into artificial intelligence or machine intelligence that uses various supervised and unsupervised machine-learning algorithms to simulate the human brain.

Emotion recognition systems recognize emotions from facial expressions, text data, body movements, voice, brain or heart signals. Along with basic emotions, attitude, control over emotions and power of activation of emotion can also be examined for analyzing sentiments.

The scope of an emotion detection project can vary depending on the specific goals and requirements of the project. Here are some examples of different project scopes:

- Facial expression recognition: The project focuses on detecting emotions from facial expressions captured from images or videos. The scope includes data collection, preprocessing, feature extraction, model selection, training, evaluation, and testing using machine learning algorithms such as Convolutional Neural Networks, Support Vector Machines, or Decision Trees.

- Speech emotion recognition: The project focuses on detecting emotions from speech signals captured from audio recordings. The scope includes data collection, preprocessing, feature extraction, model selection, training, evaluation, and testing using machine learning algorithms such as Hidden Markov Models, Gaussian Mixture Models, or Long Short-Term Memory Networks.

- Physiological signal emotion recognition: The project focuses on detecting emotions from physiological signals captured from biosensors such as electroencephalography (EEG), electrocardiography (ECG), or galvanic skin response (GSR). The scope includes data collection, preprocessing, feature extraction, model selection, training, evaluation, and testing using machine learning algorithms such as Support Vector Machines, Random Forests, or Artificial Neural Networks.

- Multimodal emotion recognition: The project focuses on integrating multiple sources of data such as facial expressions, speech signals, and physiological signals to improve the accuracy and robustness of emotion detection. The scope includes data collection, preprocessing, feature extraction, model selection, training, evaluation, and testing using machine learning algorithms such as Ensemble Learning, Deep Fusion, or Multi-Task Learning.

- Real-time emotion recognition: The project focuses on detecting emotions in real-time from streaming data such as live video or audio feeds. The scope includes developing efficient algorithms and architectures that can handle the high computational and memory demands of real-time processing, as well as designing user interfaces and feedback mechanisms that can provide timely and meaningful information to the users.

In summary, the scope of an emotion detection project depends on the data sources, the computational requirements, and the intended applications of the project.

1.3 HARDWARE REQUIREMENTS

- Central Processing Unit (CPU) - Intel Core i5 6th gen or AMD processor equivalent.
- RAM - 8 GB minimum, 16 GB or higher is recommended.
- Graphics Processing Unit (GPU) - NVIDIA GeForce GTX960 or higher
- Inbuilt Camera or Webcam Support
- Operating System (OS) - Ubuntu or Microsoft Windows 10
- Storage - 20 GB

CHAPTER-2

LITERATURE REVIEW

Emotion detection has been the subject of extensive research in recent years, and several studies have been conducted on this topic. Here are some examples of previous studies on emotion detection:

1-A study conducted by Ekman and Friesen (1971) focused on the recognition of facial expressions of emotions. They showed participants photographs of facial expressions and asked them to identify the emotions portrayed. The results showed that participants were able to recognize six basic emotions - happiness, sadness, anger, fear, surprise, and disgust - from facial expressions.

2-A study conducted by Scherer (1984) investigated the acoustic cues of emotions in speech. The study found that emotions such as happiness, sadness, anger, and fear could be recognized from speech based on various acoustic cues such as pitch, loudness, and speech rate.

3-A study conducted by Picard and colleagues (2001) investigated the use of physiological signals such as skin conductance, heart rate, and facial electromyography (EMG) for emotion detection. They found that physiological signals could provide valuable information for recognizing emotions such as anxiety, anger, and sadness.

4-A study conducted by Hu and colleagues (2014) used a deep learning approach to recognize emotions from facial expressions. They used a convolutional neural network (CNN) to extract features from facial images and achieved state-of-the-art performance in emotion recognition.

5-A study conducted by Mohammad and Turney (2013) focused on emotion detection from text. They proposed a lexicon-based approach that used a set of emotion-related words to classify emotions in text. The study showed that their approach could achieve high accuracy in emotion detection.

6-A study conducted by Alghowinem and colleagues (2013) investigated the use of a multimodal approach for emotion detection. They combined facial expressions, speech, and physiological signals to recognize emotions and achieved higher accuracy than using any single modality.

Overall, these studies demonstrate the importance of emotion detection and the different modalities that can be used for emotion recognition, including facial expressions, speech, physiological signals, and text. The studies also show that various techniques can be used for emotion detection, including feature-based methods, deep learning-based methods, and multimodal approaches

Here is a brief literature review on emotion detection:

2.1 FACIAL EXPRESSION RECOGNITION

This is one of the most common methods for detecting emotions. Researchers have developed various techniques to detect emotions from facial expressions, including feature-based approaches, appearance-based approaches, and hybrid approaches. Feature-based approaches extract geometric features such as the position and orientation of facial landmarks, while appearance-based approaches use image processing techniques to extract texture and color features from facial images. Hybrid approaches combine both feature and appearance-based techniques for improved accuracy.

Facial expression recognition is a widely used method for emotion detection as facial expressions are one of the most visible and easily recognizable indicators of emotional states. Over the years, several studies have been conducted on facial expression recognition for emotion detection, and various techniques have been proposed. Here are some examples of facial expression recognition techniques for emotion detection:

- Feature-based methods: Feature-based methods involve extracting facial features such as eye movements, eyebrow positions, and mouth shape, and using machine learning algorithms to classify emotions. These methods are simple and computationally efficient, but they require handcrafted features and may not be able to capture the complex dynamics of facial expressions. Examples of feature-based methods include the Facial Action Coding System (FACS) and the

Emotional Facial Action Coding System (EMFACS).

- Template-based methods: Template-based methods involve using predefined templates of facial expressions to classify emotions. These templates are usually created by experts in facial expression recognition, and the method involves matching the input facial expression to the closest template. Template-based methods are simple and easy to implement, but they may not be able to capture the individual variations in facial expressions. Examples of template-based methods include the Facial Action Coding System for Kids (FACS-Kids) and the Children's Emotion Coding System (CECS).

- Deep learning-based methods: Deep learning-based methods involve using convolutional neural networks (CNNs) to automatically learn features from facial images and classify emotions. These methods have shown significant improvement over traditional methods and have achieved state-of-the-art performance in facial expression recognition. Examples of deep learning-based methods include the Deep Emotion Recognition Framework (DeepER) and the Convolutional Neural Network Emotion Recognition System (CN-NER).

- Multimodal methods: Multimodal methods involve combining facial expressions with other modalities such as speech and physiological signals to improve emotion detection accuracy. These methods can capture more comprehensive information about emotions and may be more robust to individual variations in facial expressions. Examples of multimodal methods include the Multimodal Emotion Recognition System (MERS) and the Multimodal Approach for Emotion Recognition (MAER).

In conclusion, facial expression recognition is a powerful method for emotion detection, and several techniques have been proposed for this purpose. Feature-based, template-based, deep learning-based, and multimodal methods are some of the popular techniques used in facial expression recognition for emotion detection.

2.2 SPEECH EMOTION RECOGNITION

Researchers have also developed various techniques for detecting emotions from speech signals. These include acoustic-based approaches, prosody-based approaches, and language-based approaches. Acoustic-based approaches use statistical modeling techniques to extract features such as pitch, energy, and spectral features from speech signals. Prosody-based approaches focus on the variations in pitch, duration, and intensity in speech signals. Language-based approaches use natural language processing techniques to analyze the content of speech signals, such as the use of emotional words or phrases.

Speech emotion recognition (SER) is another widely used method for emotion detection. Speech carries a lot of information about emotions, including the tone, pitch, and rhythm of speech, and various techniques have been proposed for recognizing emotions from speech. Here are some examples of speech emotion recognition techniques:

- Prosodic feature-based methods: Prosodic feature-based methods involve extracting features such as pitch, intensity, and duration from speech and using machine learning algorithms to classify emotions. These methods are simple and easy to implement, but they may not be able to capture the complex variations in speech that are indicative of emotions.

- Acoustic model-based methods: Acoustic model-based methods involve building models of acoustic features such as spectral information and using machine learning algorithms to classify emotions. These methods are more complex than prosodic feature-based methods, but they can capture more comprehensive information about speech.

- Linguistic feature-based methods: Linguistic feature-based methods involve extracting features such as lexical choices and syntactic structures from speech and using machine learning algorithms to classify emotions. These methods can capture more abstract information about emotions and are less dependent on the specific acoustic characteristics of speech.

- Deep learning-based methods: Deep learning-based methods involve using neural networks such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs) to automatically learn features from speech and classify emotions. These methods have shown significant improvement over traditional methods and have achieved state-of-the-art performance in speech emotion recognition.

- Multimodal methods: Multimodal methods involve combining speech with other modalities such as facial expressions and physiological signals to improve emotion detection accuracy. These methods can capture more comprehensive information about emotions and may be more robust to individual variations in speech. Examples of multimodal methods include the Multimodal Emotion Recognition System (MERS) and the Multimodal Approach for Emotion Recognition (MAER).

In conclusion, speech emotion recognition is a powerful method for emotion detection, and several techniques have been proposed for this purpose. Prosodic feature-based, acoustic model-based, linguistic feature-based, deep learning-based, and multimodal methods are some of the popular techniques used in speech emotion recognition

2.3 PHYSIOLOGICAL SIGNAL EMOTION RECOGNITION

This method involves detecting emotions from physiological signals such as electroencephalography (EEG), electrocardiography (ECG), or galvanic skin response (GSR). Researchers have developed various techniques to analyze these signals, including time-domain, frequency-domain, and nonlinear analysis techniques. These techniques can extract features such as heart rate variability, skin conductance level, and EEG power spectral density to detect emotions.

Physiological signal emotion recognition (PSER) is another approach to emotion detection that involves measuring physiological signals such as heart rate, skin conductance, and respiration, and using machine learning algorithms to classify emotions. Here are some examples of PSER techniques:

- Heart rate variability (HRV)-based methods: HRV-based methods involve analyzing the variability of heart rate over time and using machine learning algorithms to classify emotions.

HRV is a non-invasive and easy to measure physiological signal that is sensitive to emotional changes. HRV-based methods have shown promising results in emotion recognition.

- Skin conductance (SC)-based methods: SC-based methods involve measuring the electrical conductance of the skin and using machine learning algorithms to classify emotions. SC is an indicator of sympathetic nervous system activity and is sensitive to emotional changes. SC-based methods have been used successfully in emotion recognition, particularly for detecting arousal and stress.

- Respiration-based methods: Respiration-based methods involve analyzing respiratory patterns and using machine learning algorithms to classify emotions. Respiration is affected by emotional states, and respiration-based methods have been used successfully in emotion recognition.

- Multimodal methods: Multimodal methods involve combining physiological signals with other modalities such as facial expressions and speech to improve emotion detection accuracy. These methods can capture more comprehensive information about emotions and may be more robust to individual variations in physiological signals. Examples of multimodal methods include the Multimodal Emotion Recognition System (MERS) and the Multimodal Approach for Emotion Recognition (MAER).

- In conclusion, physiological signal emotion recognition is a promising approach to emotion detection, and several techniques have been proposed for this purpose. HRV-based, SC-based, respiration-based, and multimodal methods are some of the popular techniques used in physiological signal emotion recognition.

2.4 MULTIMODAL EMOTION RECOGNITION

Researchers have also explored the use of multiple modalities such as facial expressions, speech signals, and physiological signals for improved accuracy and robustness in emotion detection. These techniques involve combining features from multiple modalities using various fusion techniques such as early fusion, late fusion, and hybrid fusion.

Overall, emotion detection is a rapidly evolving field with a wide range of techniques and applications. While significant progress has been made in detecting emotions using different modalities, there are still many challenges and limitations that need to be addressed, including the development of more robust and accurate models, the availability of larger and more diverse datasets, and the ethical considerations of emotion detection in various applications.

Multimodal emotion recognition (MER) is an approach to emotion detection that involves combining information from multiple modalities, such as facial expressions, speech, and physiological signals, to improve the accuracy and robustness of emotion detection. MER aims to capture the complexity and diversity of human emotions by utilizing multiple sources of information.

One of the most popular MER systems is the Multimodal Emotion Recognition System (MERS), which combines facial expressions, speech, and physiological signals such as skin conductance and heart rate variability. MERS uses computer vision and speech processing techniques to analyze facial expressions and speech, and machine learning algorithms to classify emotions. MERS has been shown to achieve high accuracy in recognizing emotions in various contexts, such as movies and naturalistic interactions.

Another example of MER is the Multimodal Approach for Emotion Recognition (MAER), which combines facial expressions, speech, and electroencephalogram (EEG) signals. MAER uses deep learning models to extract features from each modality and combines them using a fusion method to classify emotions. MAER has been shown to achieve high accuracy in recognizing emotions in controlled laboratory settings.
MER has several advantages over single-modality emotion recognition. First, it can capture more comprehensive and diverse information about emotions. Second, it can be more robust to individual variations in each modality. Third, it can improve the accuracy and reliability of emotion detection. However, MER also faces some challenges, such as the need for sophisticated data fusion techniques and the difficulty of integrating data from different modalities with different temporal resolutions.

In conclusion, multimodal emotion recognition is a promising approach to emotion detection, and several systems have been proposed for this purpose. MERS and MAER are two examples of popular MER systems that combine facial expressions, speech, and physiological signals to achieve high accuracy in recognizing emotions.

Chapter 3

Feasibility Study

A feasibility study is a detailed analysis that considers all of the critical aspects of a proposed project in order to determine the likelihood of it succeeding.

Success in business may be defined primarily by return on return on investment, meaning that the project will generate enough profit to justify the investment. However, many other important factors may be identified on the plus or minus side, such as community reaction and environmental impact.

A feasibility study is an important step in any project, including an emotion detection project. It helps to determine the technical, economic, operational, and legal feasibility of the project. Here are some key aspects to consider in a feasibility study for an emotion detection project:

Based on the results of the feasibility study, the project team can make informed decisions about the viability and scope of the emotion detection project. If the feasibility study indicates that the project is viable and has potential benefits, the team can proceed with the project planning and implementation. If the study indicates that the project is not feasible or has significant risks and limitations, the team can consider alternative approaches or abandon the project altogether.

Before starting the project, feasibility study is carried out to measure the viable of the system. Feasibility is necessary to determine is creating a new or improved system is friendly with the cost, benefits, operation, technology and time. Following feasibility is given below:

Why Do I Need a Feasibility Study?

Feasibility studies are important for a communications service provider to determine whether your broadband project will succeed or not. It should be the first action taken when wanting to begin a new project. It is one, if not the most important factor in determining whether the project can and should move forward. Also, if you are applying for broadband loans and grants, a feasibility study is normally required.

3.1 TECHNICAL FEASIBILITY

Technical feasibility is carried out to determine whether the project is feasible in terms of software, hardware, personnel, and expertise, to handle the completion of the project. It considers determining resources for the proposed system.

As the system is developed using python, it is platform independent. Therefore, the users of the systemcan have average processing capabilities, running on any platform. The technology is one ofthe latest hence the system is also technically feasible.

This involves questions such as whether the technology needed for the system exists, how difficult it will be to build, and whether the firm has enough experience using that technology. The assessment is based on outline design 7 of system requirements in terms of input, processes, output, fields, programs and procedures. This can be qualified in terms of volume of data, trends, frequency of updating in order to give an introduction to the technical system. The application is the fact that it has been developed on windows XP platform and a high configuration of 1 GB RAM on Intel Pentium Dual core processor. This is technically feasible. The technical feasibility assessment is focused on gaining an understanding of the present technical resources of the organization and their applicability to the expected needs of the proposed system. It is an evaluation of the hardware and software and how it meets the need of the proposed system. The technical feasibility of a fake news detection system depends on various factors such as the available resources, the complexity of the algorithms used, and the accuracy of the results. Here are some factors to consider:-

- Data Availability: The availability of labeled datasets is crucial in building an effective fake news detection system. If a dataset is not available or is insufficient, it can limit the system's accuracy.
- Algorithms and Models: The choice of algorithms and models used in the system plays a crucial role in determining its effectiveness. Some algorithms and models require a considerable amount of computational resources, which can affect the system's feasibility.

- Feature Selection: Selecting the right features to detect fake news is essential. If the system doesn't consider the right features, it can lead to inaccurate results.
- Scalability: The system should be able to handle large volumes of data and scale according to the requirements. If the system is not scalable, it can affect its efficiency and feasibility.
- Integration: The fake news detection system should integrate with the existing infrastructure to ensure that it can be used effectively. If the system is not compatible with the existing infrastructure, it can affect its feasibility.
- Infrastructure: Developing a fake news detection system requires significant computational resources, including high-performance servers, data storage, and networking infrastructure.
- Scalability: As the volume of data increases, the fake news detection system must be able to scale up to handle the additional workload. This requires designing the system to be scalable and efficient.

Overall, the technical feasibility of a fake news detection system depends on several factors, including the availability of data, choice of algorithms and models, feature selection, scalability, and integration. By carefully considering these factors, it is possible to build a feasible and effective fake news detection system.

3.2 ECONOMIC FEASIBILITY

Economic feasibility defines whether the expected benefit equals or exceeds the expected costs. It is also commonly referred to as cost/benefit analysis.

The procedure is to determine the benefits and the savings expected from the system and compare them with the costs.

A proposed system is expected to outweigh the costs. This is a small project with no cost for development. The system is easy to understand and use. Therefore, there is no need to spend on training to use the system. This system has the potential to grow by adding functionalities like adding new emotions other than added ones. This can hence, the project could have economic benefits in the future.

Establishing the cost-effectiveness of the proposed system i.e. if the benefits do not outweigh the costs then it is not worth going ahead. In the fast paced world today there is a great need of online social networking facilities. Thus the benefits of this project in the current scenario make it economically feasible. The purpose of the economic feasibility assessment is to determine the positive economic benefits to the organization that the proposed system will provide. It includes quantification and identification of all the benefits expected. This assessment typically involves a cost/benefits analysis. The economic feasibility of a fake news detection system refers to its ability to generate sufficient revenue to cover the costs of development, implementation, and maintenance, as well as provide a return on investment. Here are some key economic considerations for the development of a fake news detection system:

- Target Market: The fake news detection system must have a clearly defined target market. This may include news organizations, social media platforms, governments, or other organizations that have a vested interest in identifying and combatting fake news.
- Revenue Model: The fake news detection system must have a clearly defined revenue model that generates sufficient revenue to cover the costs of development, implementation, and maintenance. This may include subscription fees, licensing fees, or revenue sharing models.
- Cost of Development: The cost of developing the fake news detection system must be reasonable and justifiable based on the expected return on investment. This includes the cost of software development, hardware, and infrastructure, as well as any additional costs associated with training, support, and maintenance.
- Market Competition: The fake news detection system may face competition from other similar products or services. Developers must assess the competition and develop a strategy to differentiate their product and provide added value to users.
- Return on Investment: Developers must assess the potential return on investment of the fake news detection system. This includes estimating the revenue potential and comparing it to the cost of development and ongoing maintenance.
- Economic Impact: The fake news detection system may have a positive economic impact by reducing the spread of fake news and improving the accuracy and reliability of news and information. This impact should be considered in the economic feasibility analysis.

In conclusion, the economic feasibility of a fake news detection system is critical to its long-term viability. The system must have a clearly defined target market, revenue model, and return on investment, and must be competitive in a market that may have existing solutions. By addressing these economic considerations, developers can ensure that the fake news detection system is economically feasible and provides value to users.

3.3 OPERATIONAL FEASIBILITY

Operational feasibility is the measure of how well a proposed system solves the problems and takes advantage of the opportunities identified during scope definition and how it satisfies the requirements identified in the requirements analysis phase of system development. The operational feasibility assessment focuses on the degree to which the proposed development projects fits in with the existing business environment and objectives with regard to development schedule, delivery date, corporate culture and existing business processes. To ensure success, desired operational outcomes must be imparted during design and development. These include such design-dependent parameters as reliability, maintainability, supportability, usability, producibility, disposability, sustainability, affordability, and others. These parameters are requiredto be considered at the early stages of design if desired operational behaviors are to be realized.

A system design and development require appropriate and timely application of engineering and management efforts to meet the previously mentioned parameters. A system may serve its intended purpose most effectively when its technical and operating characteristics are engineeredinto the design.

Therefore, operational feasibility is a critical aspect of systems engineering that needs to be anintegralpart of the early design phases.

Operational feasibility is the measure of how well a proposed system solves the problems, and takes advantage of the opportunities identified during scope definition and how it satisfies the requirements identified in the requirements analysis phase of system development. The operational feasibility assessment focuses on the degree to which the proposed development projects fits in with the existing business environment and objectives with regard to development schedule, deliverydate, corporate culture and existing business processes. To ensure success, desired operational outcomes must be imparted during design and development. These include such design-dependent parameters as reliability, maintainability, supportability, usability, producibility, disposability, sustainability, affordability and others. These parameters are required to be considered at the early stages of design if desired operational behaviours are to be realised. A system design and development requires appropriate and timely application of engineering and management efforts to meet the previously mentioned parameters. A system may serve its intended purpose most effectively when its technical and operating characteristics are engineered into the design. Therefore, operational feasibility is a critical aspect of systems engineering that needs to be an integral part of the early design phases. Operational feasibility of a fake news detection system refers to whether it can be implemented and operated effectively in the real-world environment. Here are some key operational considerations for the development of a fake news detection system:-

- User Acceptance: The success of a fake news detection system relies on its acceptance by end-users, including journalists, fact-checkers, and the general public. Developers must consider the needs of these end-users and design a system that is user-friendly and easy to understand.
- Cost: Developing and maintaining a fake news detection system can be expensive, requiring a significant investment in hardware, software, and personnel. The cost of the system must be balanced against the benefits it provides in detecting and preventing the spread of fake news.
- Integration: The fake news detection system should be integrated with existing news platforms and social media platforms to make it more accessible and effective. Integration also ensures that the system can be easily used by end-users, without requiring significant changes to their existing workflows.
- Maintenance and Support: The fake news detection system must be maintained and updated regularly to ensure its accuracy and effectiveness. This requires a dedicated team of personnel to monitor and maintain the system and provide technical support to end-users.
- Legal and Ethical Considerations: Fake news detection systems must comply with legal and ethical standards, including privacy laws and regulations, data protection laws, and ethical considerations related to the use of personal data. Developers must ensure that the system is designed and operated in compliance with these standards.

Operational feasibility is a critical consideration in the development of a fake news detection system. Developers must consider user acceptance, cost, integration, maintenance and support, and legal and ethical considerations to ensure the system can be implemented and operated effectively in the real-world environment.

3.4 BEHAVIORAL FEASIBILITY

Establishing the cost-effectiveness of the proposed system i.e., if the benefits do not outweigh the costs, then it is not worth going ahead. In the fast-paced world today there is a great need ofonline social networking facilities. Thus, the benefits of this project in the current scenario makeit economically feasible. The purpose of the economic feasibility assessment is to determine the positive economic benefits to the organization that the proposed system will provide. It includes quantification and identification of all the benefits expected. This assessment typically involves a cost/benefits analysis.

A behavioral study of a fake news detection system refers to an examination of how users interact with the system and whether it influences their behavior regarding the consumption and sharing of news. A behavioral study of a fake news detection system can provide valuable insights into how users interact with the system and whether it is effective in promoting behavior change. Here are some key considerations for conducting a behavioral study of a fake news detection system:

- User Behavior: The study should examine how users consume and share news articles, especially those that are potentially fake. This includes their search and browsing behavior, as well as their social media engagement patterns.
- User Perception: The study should examine how users perceive the accuracy and trustworthiness of news articles. This includes their assessment of the veracity of headlines, content, and sources.
- User Interaction: The study should examine how users interact with the fake news detection system, including their willingness to input articles for analysis and their understanding of the system's output.
- Impact on User Behavior: The study should examine whether the fake news detection system influences users' behavior regarding the consumption and sharing of news articles. This includes whether users are more likely to fact-check articles before sharing them or if they are less likely to trust articles that are flagged as potentially fake.
- Effectiveness of the System: The study should examine the effectiveness of the fake news detection system in identifying potentially fake news articles. This includes whether the system accurately identifies fake news articles and whether users trust the system's output.

- User Satisfaction: The study should examine user satisfaction with the fake news detection system. This includes whether users find the system easy to use and whether they perceive the system as providing value.
- User Experience: The study should evaluate users' experience using the fake news detection system, including the ease of use, usefulness, and satisfaction. This can be done through usability testing or user feedback. Evaluating user experience can help developers identify areas for improvement and optimize the system for better user engagement.
- Behavioral Intention: The study should assess users' intention to use the fake news detection system, including their motivation, intention, and willingness to adopt the system. This can be done through surveys or behavioral intention scales. Understanding behavioral intention can help developers identify potential adoption barriers and develop strategies to overcome them.

In conclusion, a behavioral study of a fake news detection system is critical to understanding how users interact with the system and whether it influences their behavior regarding the consumption and sharing of news. By addressing these behavioral considerations, researchers can improve the design and implementation of fake news detection systems, increasing their effectiveness in combating the spread of misinformation.

CHAPTER 4
DATABASE DESIGN

4.1 FLOW CHART

Flowchart is a graphical representation of an algorithm. Programmers often use it as a program-planning tool to solve a problem. It makes use of symbols which are connected amongthem to indicate the flow of information and processing. The process of drawing a flowchart for an algorithm is known as "flow charting".

Basic Symbols used in Flowchart Designs

Terminal: The oval symbol indicates Start, Stop and Halt in a program's logic flow. A pause/halt is generally used in a program logic under some error conditions. Terminal is the first and last symbols in the flowchart.

Input/Output: A parallelogram denotes any function of input/output type. Program instructions that take input from input devices and display output on output devices are indicated with parallelogram in a flowchart.

Processing: A box represents arithmetic instructions. All arithmetic processes such as adding, subtracting, multiplication and division are indicated by action or process symbol.

Decision: Diamond symbol represents a decision point. Decision based operations such as yes/no question or true/false are indicated by diamond in flowchart.

Connectors: Whenever flowchart becomes complex or it spreads over more than one page, itisuseful to use connectors to avoid any confusions. It is represented by a circle.

Flow lines: Flow lines indicate the exact sequence in which instructions are executed. Arrows represent the direction of flow of control and relationship among different symbols of flowchart.

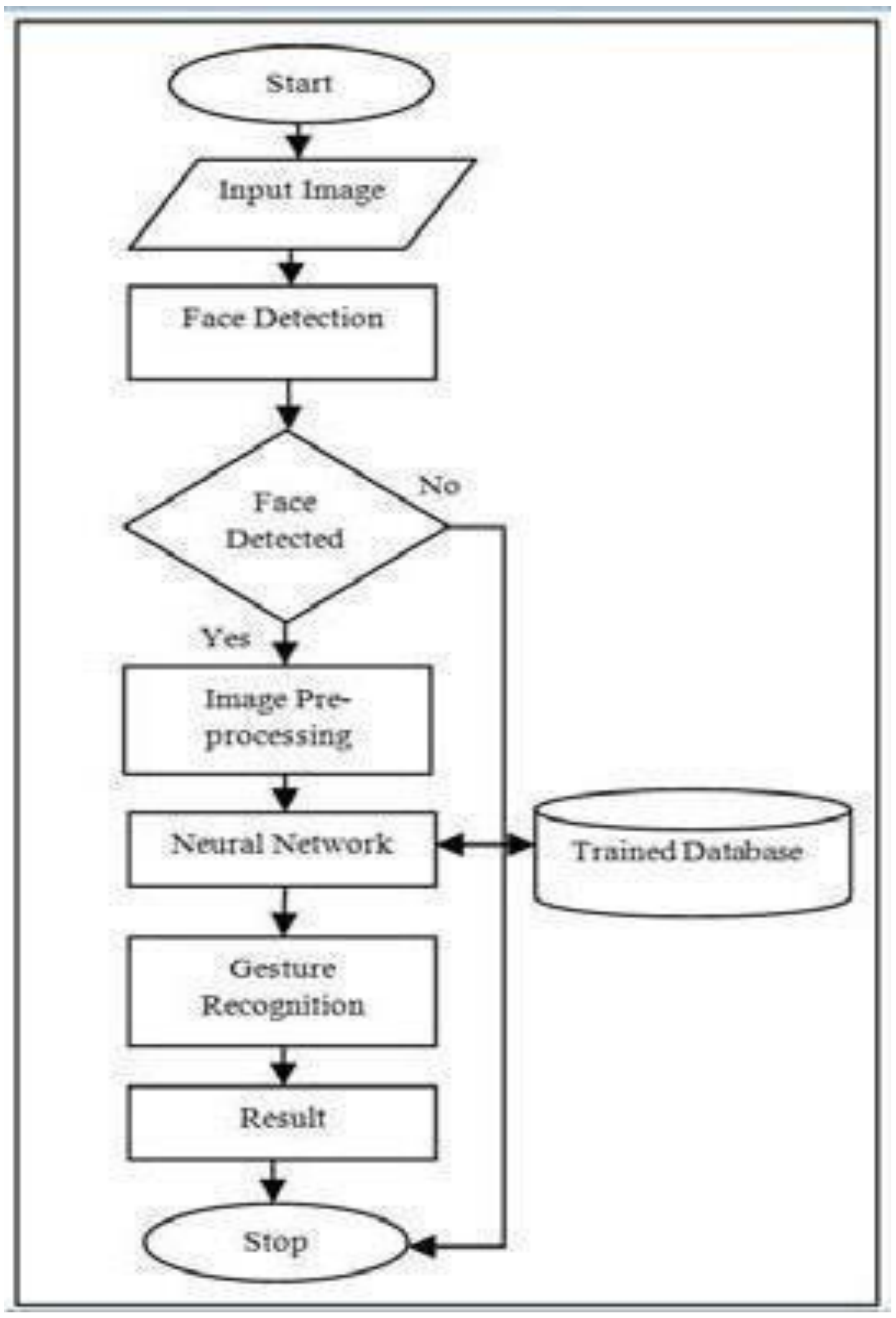


Figure 1 Flow Chart

4.2 USE CASE DIAGRAM

In the Unified Modeling Language (UML), a use case diagram can summarize the details of your system'susers (also known as actors) and their interactions with the system. To build one, you'll use aset of specialized symbols and connectors. An effective use case diagram can help your team discuss and represent:

Scenarios in which your system or application interacts with people, organizations, or external systems
A use case diagram is a visual representation of the interactions between actors and a system in various scenarios. A use case diagram is a valuable tool in project reports for illustrating the functionality of a system and its interactions with actors. It helps to identify the different use cases of the system, the actors involved, and their relationships. In a project report, a use case diagram can be used to provide a visual representation of the requirements and features of the system. The diagram can be accompanied by a brief description of each use case, including its purpose and the steps involved. Moreover, the use case diagram can help stakeholders to understand the system's functionalities and requirements better. It can also serve as a basis for testing and validation of the system. By visualizing the system's interactions, stakeholders can identify potential issues and requirements that may need to be addressed. Overall, including a use case diagram in a project report is a useful way to illustrate the system's functionality and requirements to stakeholders. It helps to ensure that everyone involved in the project has a clear understanding of the system and its interactions with the actors.

A use case diagram is a type of UML diagram that is used to illustrate the different ways that users interact with a system. It shows the various use cases, actors, and their relationships. In a project report for a fake news detection system, a use case diagram can be used to show the different types of users that interact with the system and the tasks that they perform.

For example, a use case diagram for a fake news detection system might include actors such as news readers, news curators, and administrators. News readers might use the system to check the credibility of a news article, while news curators might use the system to review and evaluate news content. Administrators might use the system to manage user accounts and perform maintenance tasks.

The use cases themselves might include tasks such as submitting news content, reviewing news content, and checking the credibility of news content. These tasks would be represented by rectangular boxes on the diagram. Arrows would be used to show the relationships between the actors and the use cases they perform.

Goals that your system or application helps those entities (known as actors) achieve

Use case Diagram Components

To answer the question, "What is a use case diagram?" you need to first understand its building blocks. Common components include:

Actors: The users that interact with a system. An actor can be a person, an organization, or an outside system that interacts with your application or system. They must be external objects that produce or consume data.

System: A specific sequence of actions and interactions between actors and the system. Asystem may also be referred to as a scenario.

Goals: The result of most use cases. A successful diagram should describe the activities and variants used to reach the goal.

Use case diagram symbols and notation

The notation for a use case diagram is straightforward and doesn't involve as many types ofsymbolsas other UML diagrams.

Use cases: Horizontally shaped ovals that represent the different uses that a user might have.

Actors: Stick figures that represent the people employing the use cases.

Associations: A line between actors and use cases. In complex diagrams, it is important to know which actors are associated with which use cases.

System boundary boxes: A box that sets a system scope to use cases. All use cases outside the box would be considered outside the scope of that system. For example, Psycho Killer is outsidethe scope of occupations in the chainsaw example found below.

Packages: A UML shape that allows you to put different elements into groups. Just as with component diagrams, these groupings are represented as file folders.

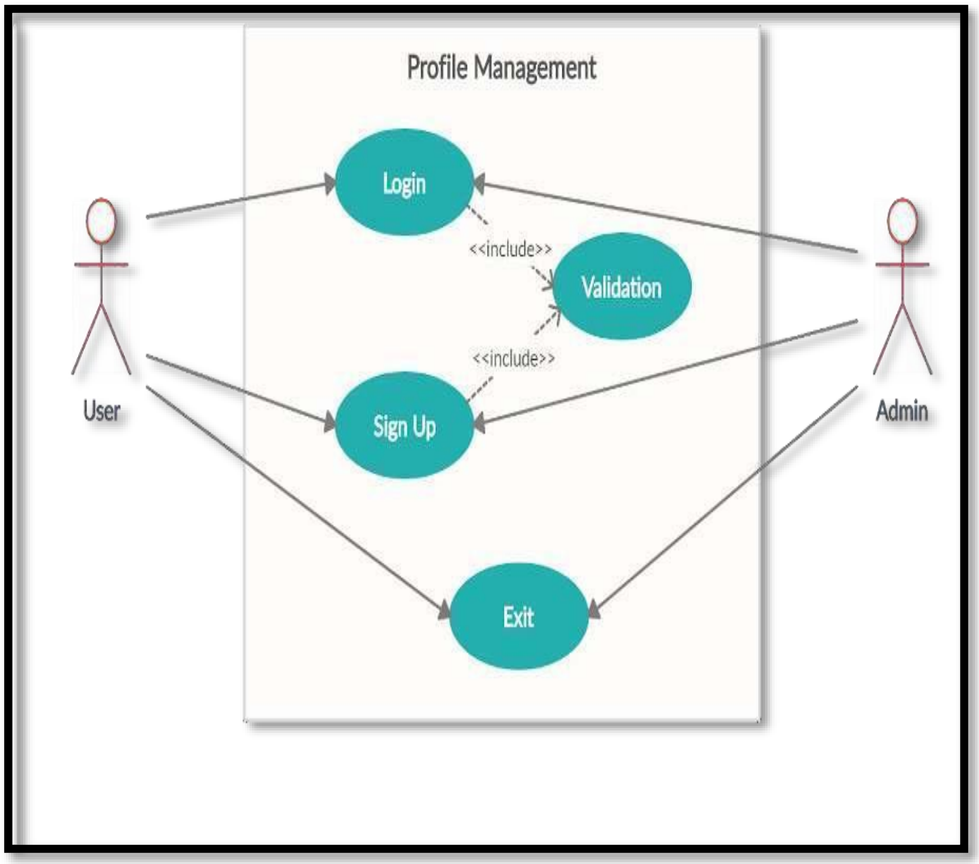


Figure 2 Profile Management
Use Case

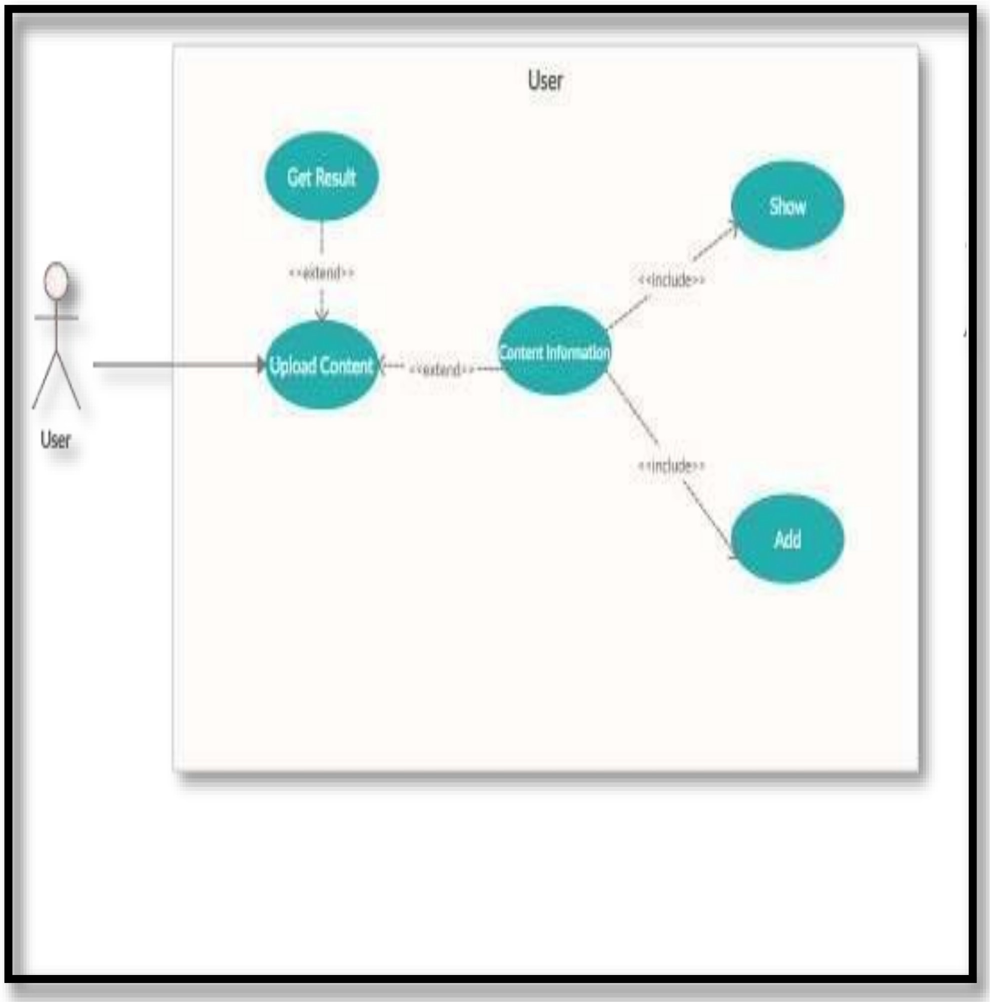


Figure 3 User Use Case

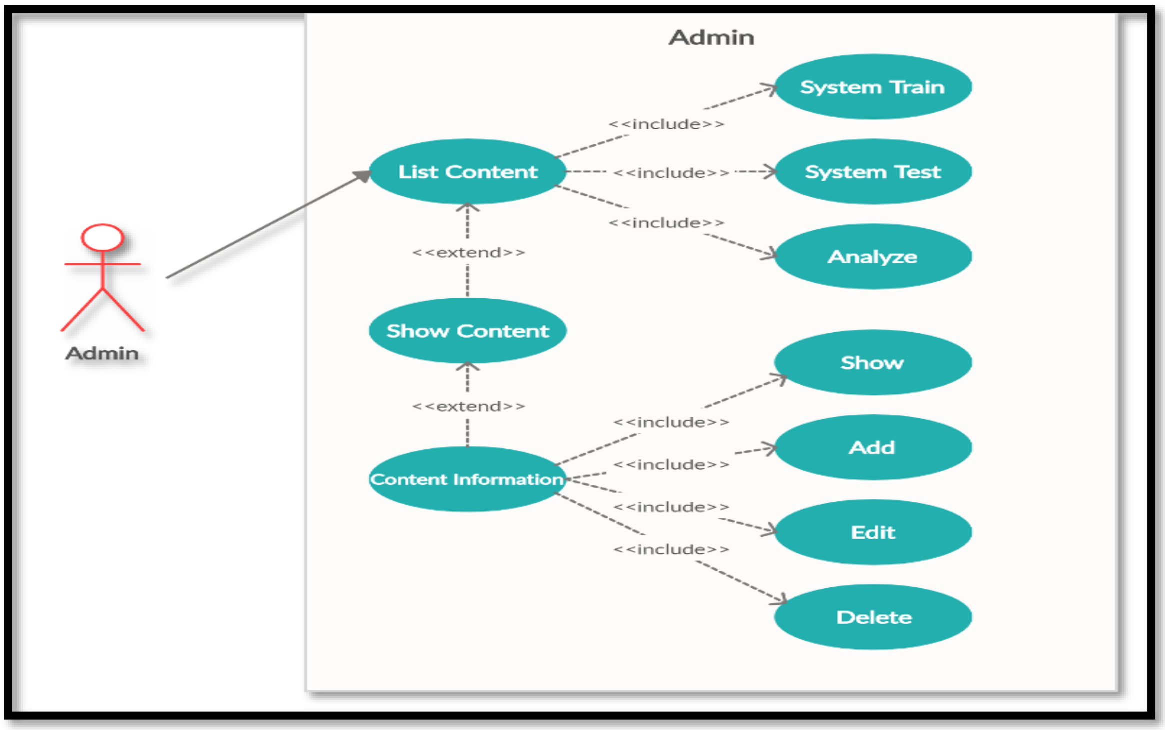


Figure 4 Admin Use Case

CHAPTER 5

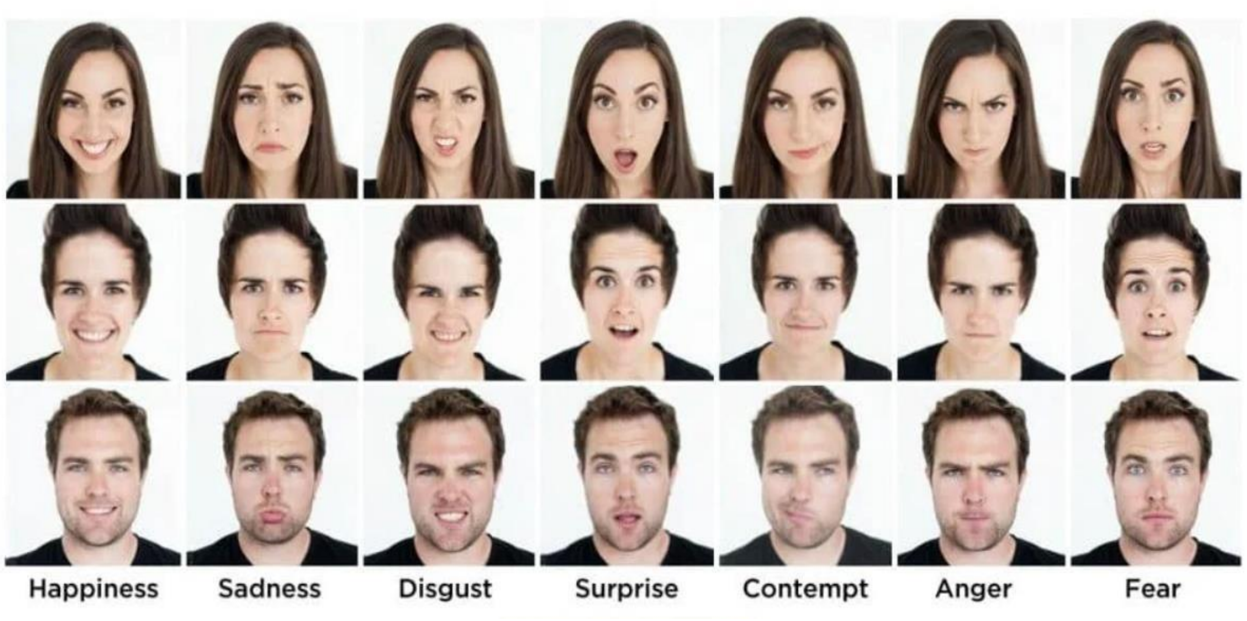
FORM DESIGN

5.1 DATA SET AND DATA COLLECTION MODULE

This module consists of the data that we have used in the formation of this project or thesystemi.e., Mood Recognition System.

The data set that we have used is the Keras data set.

Data set consist of various emotions with each picture consisting of specified emotion.



Using all this data collecting from the data set available in the Keras Data set, we haveused in our Code as by implementing libraries to recognize the emotion of a person.

Different modules that we have used in this Recognition system are: -

- Active Module
- Detecting Module
- Emotions Module
- Exit Module

Active Module: - This module is used for stating the Recognition System window where aperson is supposed to keep the face with required brightness and face.

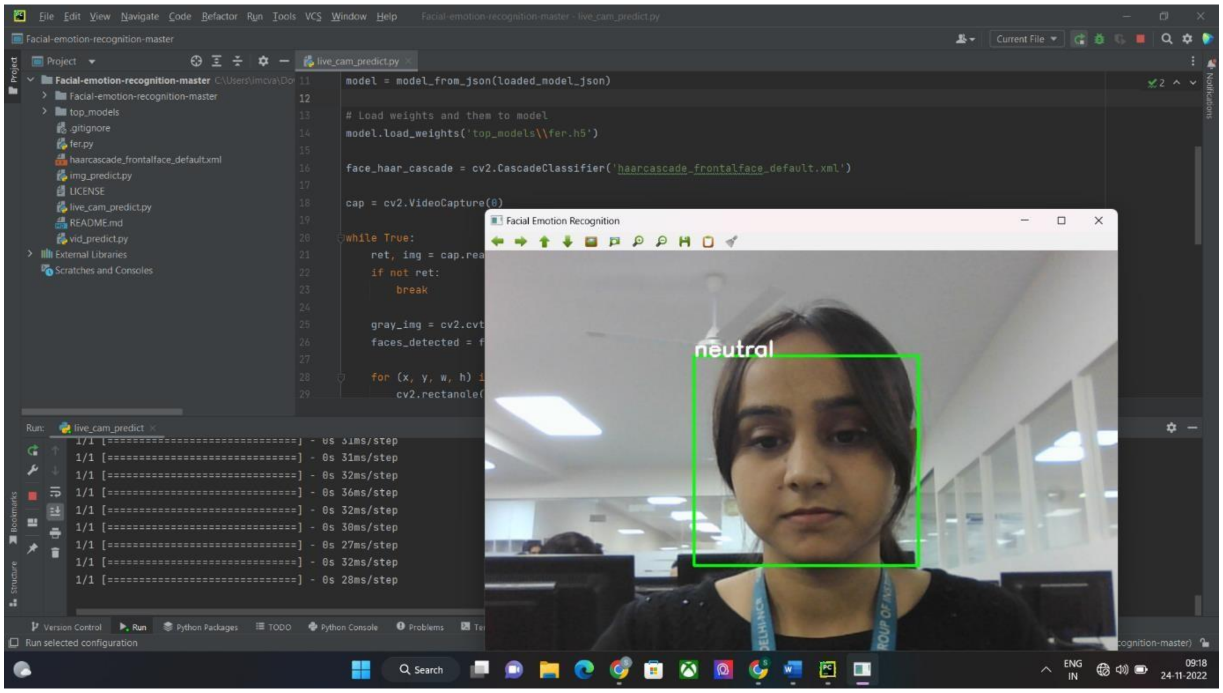
Detecting Module: - This module is used for detecting the face in the Recognition Systemwindow where a person is supposed to keep the face with required brightness and face.

Emotions Module: - This module is used for detecting the face in the Recognition Systemwindow where a person is supposed to keep the face with required brightness and face.
After recognizing the face, it will present the emotions of the person, if he is happy, sad,angry, feared, confused, surprised etc.

Exit Module: - This module is used for closing the window of the Recognition System,whereas person was detecting his/her emotions.

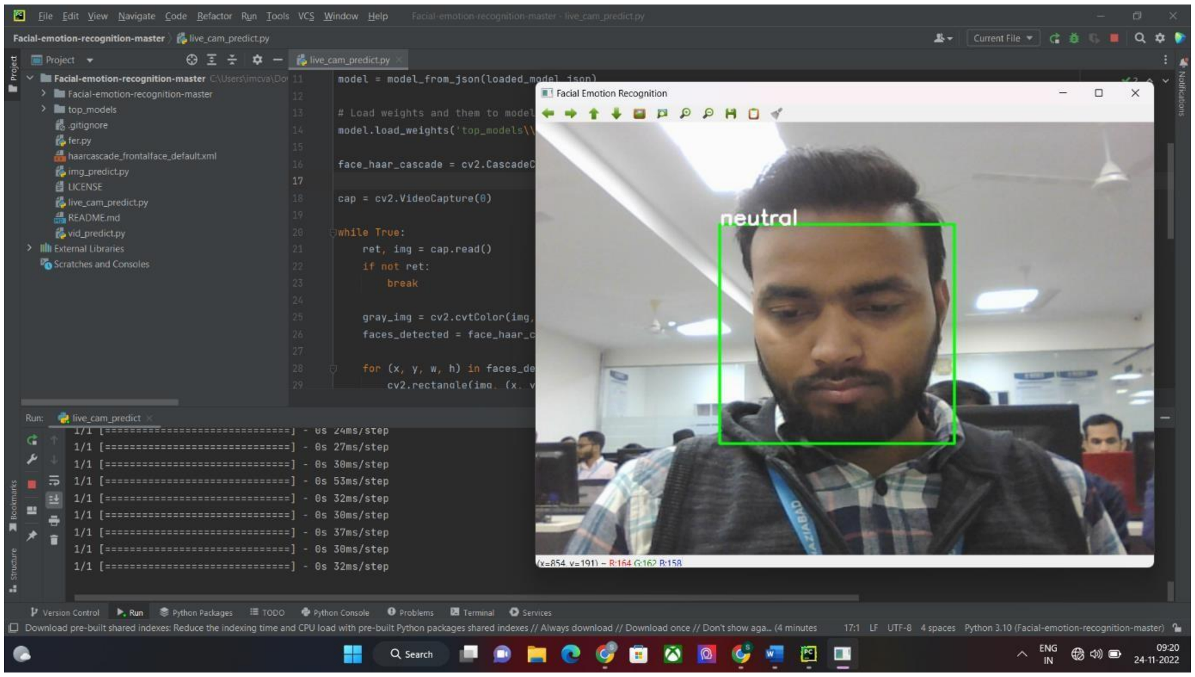
5.2OUTPUT FORM (SCREENSHOT)

Output 1 : System look(Mood- Neutral)



```
model = model_from_json(loaded_model_json)

# Load weights and them to model
model.load_weights('top_models\\fer.h5')

face_haar_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')

cap = cv2.VideoCapture(0)

while True:
    ret, img = cap.rea
    if not ret:
        break

    gray_img = cv2.cvt
    faces_detected = f

    for (x, y, w, h) i
        cv2.rectangle(
```

```
1/1 [==============================] - 0s 31ms/step
1/1 [==============================] - 0s 31ms/step
1/1 [==============================] - 0s 32ms/step
1/1 [==============================] - 0s 36ms/step
1/1 [==============================] - 0s 32ms/step
1/1 [==============================] - 0s 30ms/step
1/1 [==============================] - 0s 27ms/step
1/1 [==============================] - 0s 32ms/step
1/1 [==============================] - 0s 28ms/step
```

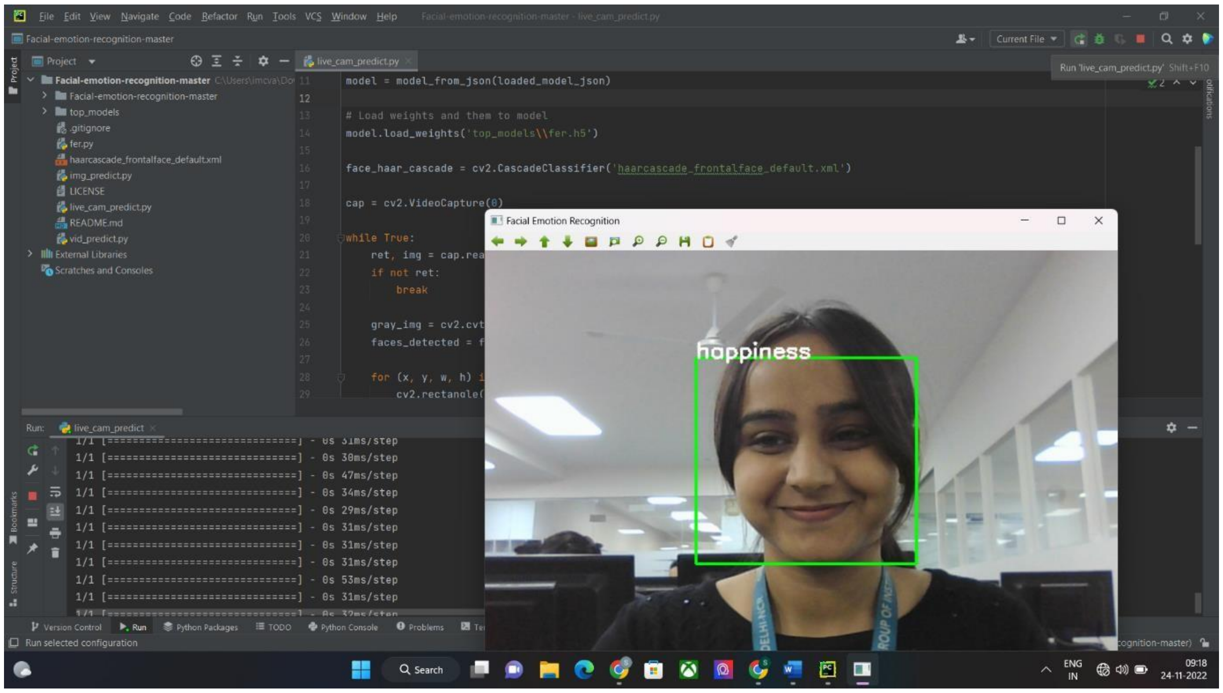


```
model = model_from_json(loaded_model_json)

# Load weights and them to model
model.load_weights('top_models\\

face_haar_cascade = cv2.CascadeC

cap = cv2.VideoCapture(0)

while True:
    ret, img = cap.read()
    if not ret:
        break

    gray_img = cv2.cvtColor(img,
    faces_detected = face_haar_c

    for (x, y, w, h) in faces_de
        cv2.rectangle(img, (x, y
```

```
1/1 [==============================] - 0s 24ms/step
1/1 [==============================] - 0s 27ms/step
1/1 [==============================] - 0s 30ms/step
1/1 [==============================] - 0s 53ms/step
1/1 [==============================] - 0s 32ms/step
1/1 [==============================] - 0s 30ms/step
1/1 [==============================] - 0s 37ms/step
1/1 [==============================] - 0s 30ms/step
1/1 [==============================] - 0s 32ms/step
```

Facial Emotion Recognition
neutral
Download pre-built shared indexes: Reduce the indexing time and CPU load with pre-built Python packages shared indexes // Always download // Download once // Don't show aga... (4 minutes

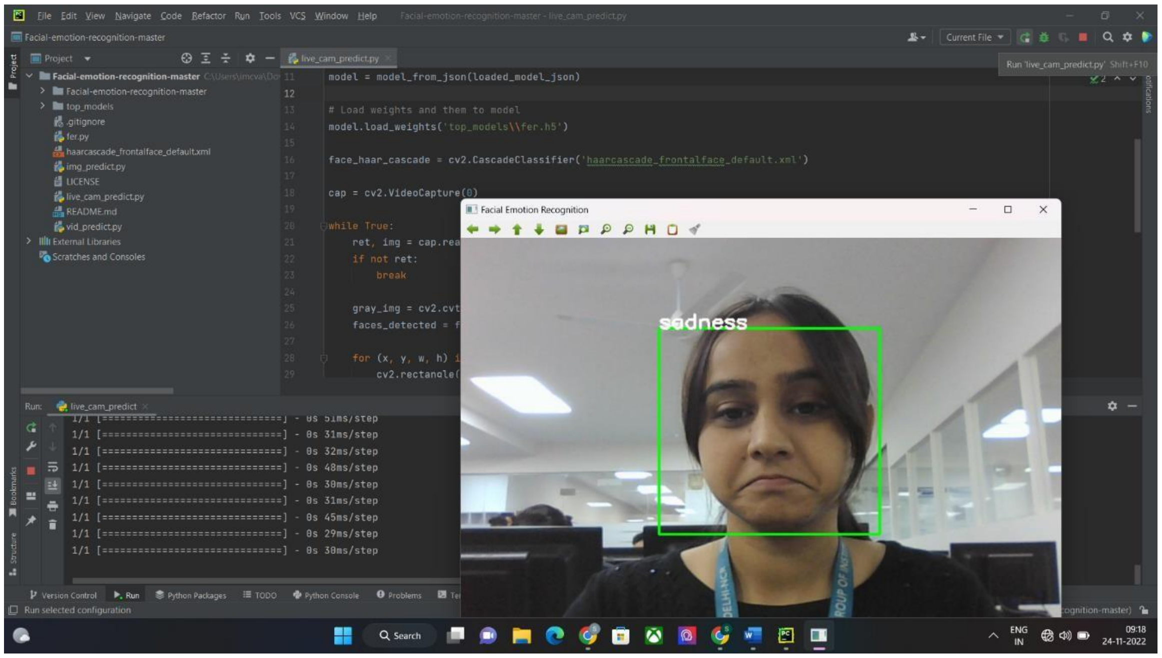
Output 2 : System look(Mood- Happiness)



```
model = model_from_json(loaded_model_json)

# Load weights and them to model
model.load_weights('top_models\\fer.h5')

face_haar_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')

cap = cv2.VideoCapture(0)

while True:
    ret, img = cap.rea
    if not ret:
        break

    gray_img = cv2.cvt
    faces_detected = f

    for (x, y, w, h) i
        cv2.rectangle(
```

```
1/1 [==============================] - 0s 31ms/step
1/1 [==============================] - 0s 30ms/step
1/1 [==============================] - 0s 47ms/step
1/1 [==============================] - 0s 34ms/step
1/1 [==============================] - 0s 29ms/step
1/1 [==============================] - 0s 31ms/step
1/1 [==============================] - 0s 31ms/step
1/1 [==============================] - 0s 31ms/step
1/1 [==============================] - 0s 53ms/step
1/1 [==============================] - 0s 31ms/step
1/1 [==============================] - 0s 32ms/step
```

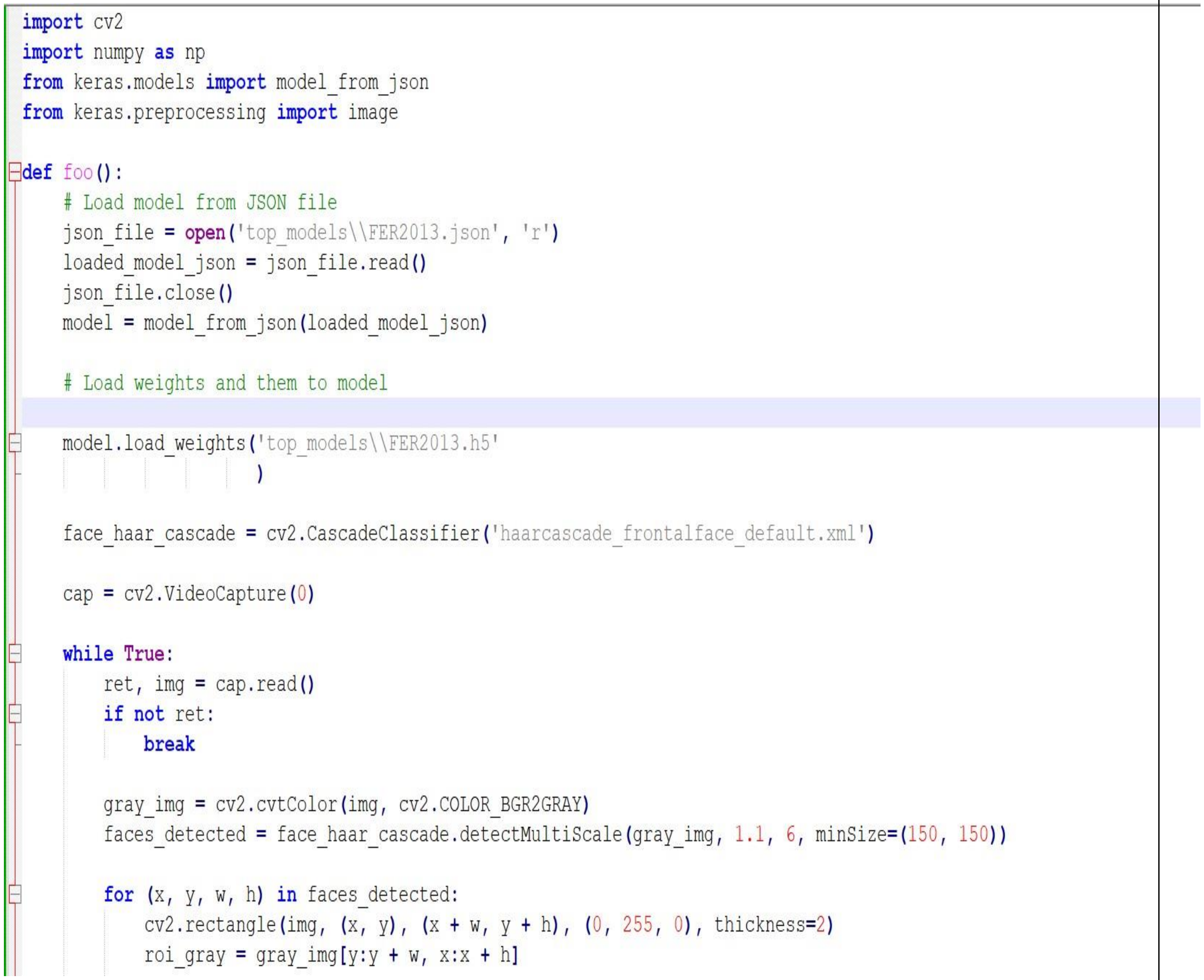
Output 3 : System look(Mood- Sadness)



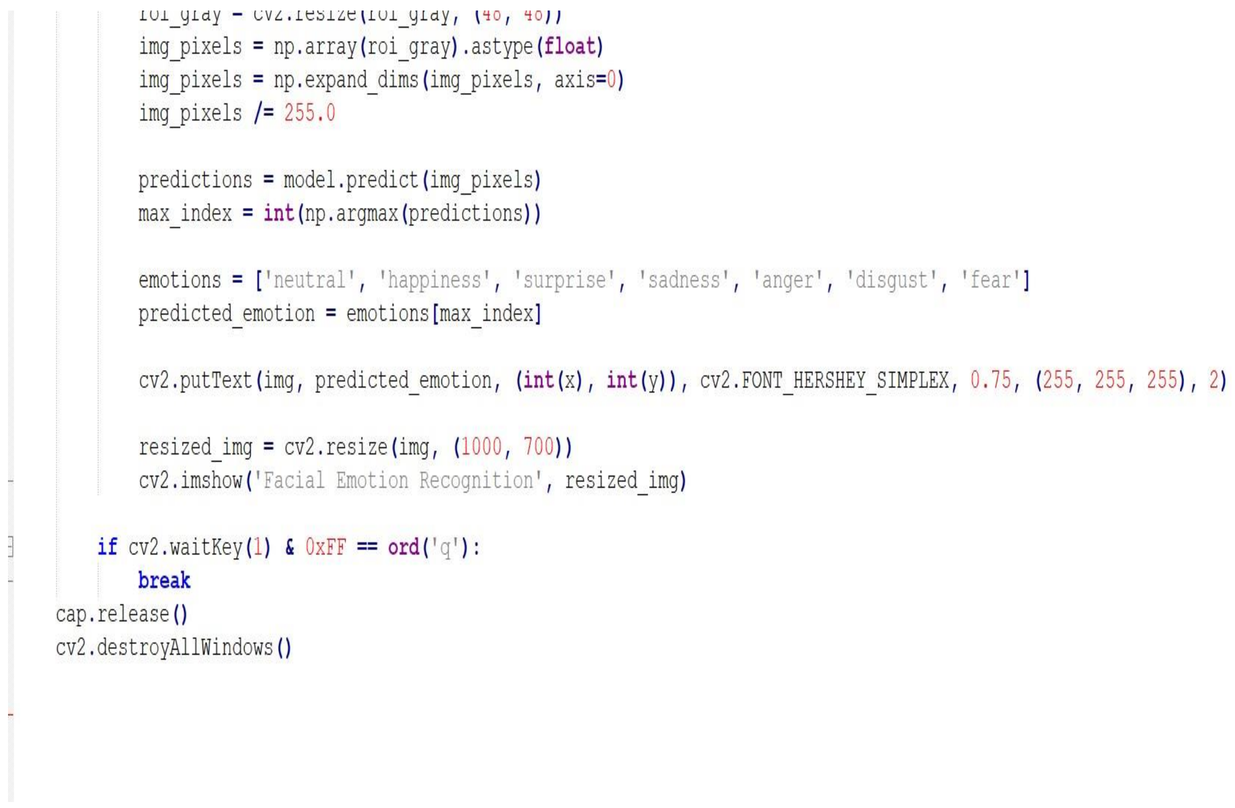
```
model = model_from_json(loaded_model_json)

# Load weights and them to model
model.load_weights('top_models\\fer.h5')

face_haar_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')

cap = cv2.VideoCapture(0)

while True:
    ret, img = cap.rea
    if not ret:
        break

    gray_img = cv2.cvt
    faces_detected = f

    for (x, y, w, h) i
        cv2.rectangle(
```

```
1/1 [==============================] - 0s 31ms/step
1/1 [==============================] - 0s 31ms/step
1/1 [==============================] - 0s 32ms/step
1/1 [==============================] - 0s 48ms/step
1/1 [==============================] - 0s 30ms/step
1/1 [==============================] - 0s 31ms/step
1/1 [==============================] - 0s 45ms/step
1/1 [==============================] - 0s 29ms/step
1/1 [==============================] - 0s 30ms/step
```

CHAPTER 6
CODING



```
import cv2
import numpy as np
from keras.models import model_from_json
from keras.preprocessing import image

def foo():
    # Load model from JSON file
    json_file = open('top_models\\FER2013.json', 'r')
    loaded_model_json = json_file.read()
    json_file.close()
    model = model_from_json(loaded_model_json)

    # Load weights and them to model

    model.load_weights('top_models\\FER2013.h5'
                       )

    face_haar_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')

    cap = cv2.VideoCapture(0)

    while True:
        ret, img = cap.read()
        if not ret:
            break

        gray_img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
        faces_detected = face_haar_cascade.detectMultiScale(gray_img, 1.1, 6, minSize=(150, 150))

        for (x, y, w, h) in faces_detected:
            cv2.rectangle(img, (x, y), (x + w, y + h), (0, 255, 0), thickness=2)
            roi_gray = gray_img[y:y + w, x:x + h]
```



```
        roi_gray = cv2.resize(roi_gray, (48, 48))
        img_pixels = np.array(roi_gray).astype(float)
        img_pixels = np.expand_dims(img_pixels, axis=0)
        img_pixels /= 255.0

        predictions = model.predict(img_pixels)
        max_index = int(np.argmax(predictions))

        emotions = ['neutral', 'happiness', 'surprise', 'sadness', 'anger', 'disgust', 'fear']
        predicted_emotion = emotions[max_index]

        cv2.putText(img, predicted_emotion, (int(x), int(y)), cv2.FONT_HERSHEY_SIMPLEX, 0.75, (255, 255, 255), 2)

        resized_img = cv2.resize(img, (1000, 700))
        cv2.imshow('Facial Emotion Recognition', resized_img)

    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
cap.release()
cv2.destroyAllWindows()
```

CHAPTER 7
TESTING

7.1 UNIT TESTING

In computer programming, unit testing is a software testing method by which individual units of source code, sets of one or more computer program modules together with associated control data, usage procedures, and operating procedures, are tested to determine whether they are fit for use. Intuitively, one can view a unit as the smallest testable part of an application. In procedural programming, a unit could be an entire module, but it is more commonly an individual function or procedure. In object-oriented programming, a unit is often an entire interface, such as a class, but could be an individual method. Unit tests are short code fragments created by programmers or occasionally by white box testers during the development process. It forms the basis for componenttesting. Ideally, each test case is independent from the others. Substitutes such as method stubs, mock objects, fakes, and test harnesses can be used to assist testing a module in isolation. Unit testsare typically written and run by software developers to ensure that code meets its design and behaves as intended.

Benefits

The goal of unit testing is to isolate each part of the program and show that the individual parts are correct. A unit test provides a strict, written contract that the piece ofcode must satisfy. As a result, it offers several benefits.

1. **Find problems early:** Unit testing finds problems early in the development cycle. In test- driven development (TDD), which is frequently used in both extreme programming and scrum, unit tests are created before the code itself is written. When the tests pass, that code is considered complete. The same unit tests are run against that function frequently as the larger code base is developed either as the code is changed or via an automated process with the build. If the unit tests fail, it is a bug either in the changed code or the tests themselves. The unit tests then allow the location of the fault or failure to be easily traced. Since the unit tests alert the development team of the problem before handing the code off to testers or clients, it is still early in the development process.

2. **Facilitates Change:** Unit testing allows the programmer to refactor code or upgrade system libraries later, and make sure the module still works correctly (e.g., in regression testing). The procedure is to write test cases for all functions and methods so that whenever a change causes a fault, it can be quickly identified. Unit tests detect changes which may break a design contract.

3. **Simplifies Integration:** Unit testing may reduce uncertainty in the units themselves and can be used in a bottom-up testing style approach. By testing the parts of a program first and then testing the sum of its parts, integration testing becomes much easier.

4. **Documentation:** Unit testing provides a sort of living documentation of the system. Developers looking to learn what functionality is provided by a unit, and how to use it, can look at the unit tests to gain a basic understanding of the unit's interface (API). Unit test cases embody characteristics that are critical to the success of the unit. These characteristics can indicate appropriate/inappropriate use of a unit as well as negative behaviors that are to be trapped by the unit. A unit test case, in and of itself, documents these critical characteristics, although many software development environments do not rely solely upon code to document the product in development unit. A unit test case, in and of itself, documents these critical characteristics, although many software development environments do not rely solely upon code to document the product in development.

7.2 INTEGRATION TESTING

Integration testing (sometimes called integration and testing, abbreviated I&T) is the phase in software testing in which individual software modules are combined and tested as a group. It occurs after unit testing and before validation testing. Integration testing takes as its input modules that have been unit tested, groups them in larger aggregates, applies tests defined in an integration test plan to those aggregates, and delivers as its output the integrated system ready for system testing.

Purpose

The purpose of integration testing is to verify functional, performance, and reliability requirements placed on major design items. These "design items", i.e., assemblages (or groups of units), are exercised through their interfaces using black-box testing, success and error cases being simulated via appropriate parameter and data inputs. Simulated usage of shared data areas and inter-process communication is tested and individual subsystems are exercised through their input interface. Testcases are constructed to test whether all the components within assemblages interact correctly, for example across procedure calls or process activations, and this is done after testing individual modules, i.e., unit testing. The overall idea is a "building block" approach, in which verified assemblages are added to a verified base which is then used to support the integration testing of further assemblages. Software integration testing is performed according to the software development life cycle (SDLC) after module and functional tests. The cross-dependencies for software integration testing are schedule for integration testing, strategy and selection of the tools used for integration, define the cyclomatic complexity of the software and software architecture, reusability of modules and life cycle and versioning management. Some different types of integration testing are big-bang, top- down, and bottom-up, mixed (sandwich) and risky- hardest. Other Integration Patterns are collaboration integration, backbone integration, layer integration, client-server integration, distributed services integration and high-frequency integration.

- **Big Bang**

In the big-bang approach, most of the developed modules are coupled together to form a complete software system or major part of the system and then used for integration testing. This method is very effective for saving time in the integration testing process. However, if the test cases and their results are not recorded properly, the entire integration process will be more complicated and may prevent the testing team from achieving the goal of integration testing. A type of big-bang integration testing is called "usage model testing" which can be used in both software and hardware integration testing. The basis behind this type of integration testing is to run user-like workloads in integrated user-like environments. In doing the testing in this manner, the environment is proofed, while the individual components are proofed indirectly through their use. Usage Model testing takes an optimistic approach to testing because it expects to have few problems with the individual components. The strategy relies heavily on the component developers to do the isolated unit testing for their product. The goal of the strategy is to avoid redoing the testing done by the developers, and instead flesh-out problems caused by the interaction of the components in the environment. For integration testing, Usage Model testing can be more efficient and provides better test coverage than traditional focused functional integration testing. To be more efficient and accurate, care must be used in defining the user-like workloads for creating realistic scenarios in exercising the environment. This gives confidence that the integrated environment will work as expected for the target customers.

- **Top-down And Bottom-up**

Bottom-up testing is an approach to integrated testing where the lowest level components are tested first, then used to facilitate the testing of higher-level components. The process is repeated until the component at the top of the hierarchy is tested. All the bottom or low-level modules, procedures or functions are integrated and then tested. After the integration testing of lower-level integrated modules, the next level of modules will be formed and can be used for integration testing. This approach is helpful only when all or most of the modules of the same development level are ready. This method also helps to determine the levels of software developed and makes it easier to report testing progress in the form of a percentage. Top-down testing is an approach to integrated testing where the top integrated modules are tested, and the branch of the module is tested step by step until the end of the related module. Sandwich testing is an approach to combine top-down testing with bottom-up testing.

7.3 SOFTWARE VERIFICATION AND VALIDATION

Introduction

In software project management, software testing, and software engineering, verification and validation (V&V) is the process of checking that a software system meets specifications and that it fulfills its intended purpose. It may also be referred to as software quality control. It is normallythe responsibility of software testers as part of the software development lifecycle. Validation checks that the product design satisfies or fits the intended use (high-level checking), i.e., the software meets the user requirements. This is done through dynamic testing and other forms of review. Verification and validation are not the same thing, although they are often confused. Boehm succinctly expressed the difference between

- Validation: Are we building the right product?

- Verification: Are we building the product, right?

According to the Capability Maturity Model (CMMI-SWv1.1)

Software Verification: The process of evaluating software to determine whether the products of a given development phase satisfy the conditions imposed at the start of that phase.

Software Validation: The process of evaluating software during or at the end of the development process to determine whether it satisfies specified requirements.

In other words, software verification is ensuring that the product has been built according to the requirements and design specifications, while software validation ensures that the product meets the user's needs, and that the specifications were correct in the first place. Software verification ensures that "you built it right". Software validation ensures that "you built the right thing". Software validation confirms that the product, as provided, will fulfill its intended use.

From Testing Perspective

- Fault – wrong or missing function in the code.
- Failure – the manifestation of a fault during execution.
- Malfunction – according to its specification the system does not meet its specified functionality

Both verification and validation are related to the concepts of quality and of software quality assurance. By themselves, verification and validation do not guarantee software quality; planning, traceability, configuration management and other aspects of software engineering are required. Within the modeling and simulation (M&S) community, the definitions of verification, validation and accreditation are similar:

- M&S Verification is the process of determining that a computer model, simulation, or federation of models and simulations implementations and their associated data accurately represent the developer's conceptual description and specifications.
- M&S Validation is the process of determining the degree to which a model, simulation, or federation of models and simulations, and their associated data are accurate representations of the real

world from the perspective of the intended use(s).

Classification of Methods

In mission-critical software systems, where flawless performance is absolutely necessary, formal methods may be used to ensure the correct operation of a system. However, often for non- mission-critical software systems, formal methods prove to be very costly, and an alternative method of software V&V must be sought out. In such cases, syntactic methods are often used.

Test Cases

A test case is a tool used in the process. Test cases may be prepared for software verification and software validation to determine if the product was built according to the requirements of the user. Other methods, such as reviews, may be used early in the life cycle to provide for software validation.

7.4 BLACK-BOX TESTING

Black-box testing is a method of software testing that examines the functionality of an application without peering into its internal structures or workings. This method of test can be applied virtually to every level of software testing: unit, integration, system and acceptance. It typically comprises most if not all higher-level testing but can also dominate unit testing as well.

Test Procedures

Specific knowledge of the application's code/internal structure and programming knowledge in general is not required. The tester is aware of what the software is supposed to do but is not awareof how it does it. For instance, the tester is aware that a particular input returns a certain, invariable output but is not aware of how the software produces the output in the first place.

Test Cases

Test cases are built around specifications and requirements, i.e., what the application is supposed to do. Test cases are generally derived from external descriptions of the software, including specifications, requirements and design parameters. Although the tests used are primarily functional in nature, non-functional tests may also be used. The test designer selects both valid and invalid inputs and determines the correct output, often with the help of an oracle or a previous

result that is known to be good, without any knowledge of the test object's internal structure.

7.5 White-Box Testing

White-box testing (also known as clear box testing, glass box testing, transparent box testing, and structural testing) is a method of testing software that tests internal structures or workings of an application, as opposed to its functionality (i.e., black-box testing). In white box testing an internal perspective of the system, as well as programming skills, are used to design test cases. The tester chooses inputs to exercise paths through the code and determine the appropriate outputs. This is analogous to testing nodes in a circuit, e.g., in-circuit testing (ICT). White-box testing can be applied at the unit, integration and system levels of the software testing process. Although traditional testers tended to think of white box testing as being done at the unit level, itis used for integration and system testing more frequently today. It can test paths within a unit, paths between units during integration, and between subsystems during a system–level test. Though this method of test design can uncover many errors or problems, it has the potential to miss unimplemented parts of the specification or missing requirements.

Levels

1) **Unit testing:** White-box testing is done during unit testing to ensure that the code is working as intended, before any integration happens with the previously tested code. White box testing during unit testing catches any defects early on and aids in any defects that happen later on after the code is integrated with the rest of the application and therefore prevents any type of errors later on.

2) **Integration testing:** White box testing at this level are written to test the interactions of each interface with each other. The Unit level testing made sure that each code was tested and working accordingly in an isolated environment and integration examines the correctness of the behavior in an open environment through the use of white box testing for any interactions of interfaces thatare known to the programmer.

3) Regression testing: White-box testing during regression testing is the use of recycled white- box test cases at the unit and integration testing levels.

Procedures

White-box testing's basic procedures involve the tester having a deep level of understanding of the source code being tested. The programmer must have a deep understanding of the application to know what kinds of testcases to create so that every visible path is exercised for testing. Once the source code is understood then the source code can be analyzed for test cases to be created. These are the three basic steps that white box testing takes in order to create test cases:

- Input involves different types of requirements, functional specifications, detailed designing of documents, proper source code, security specifications. This is the preparation stage of Whitebox testing to lay out all the basic information.

- Processing involves performing risk analysis to guide the whole testing process, proper test plan, execute test cases and communicate results. This is the phase of building test cases to make sure they thoroughly test the application the given results are recorded accordingly.

- Output involves preparing a final report that encompasses all the above preparations and results.

Advantages

White-box testing is one of the two biggest testing methodologies used today. It has several major advantages:

- Side effects of having the knowledge of the source code are beneficial to thorough testing.
- Optimization of code by revealing hidden errors and being able to remove these possible

defects.

- Gives the programmer introspection because developers carefully describe any new implementation.
- Provides traceability of tests from the source, allowing future changes to the software to beeasily captured in changes to the tests.
- White box testing gives clear, engineering-based, rules for when to stop testing.

Disadvantages

- Although white-box testing has great advantages, it is not perfect and contains some disadvantages:
- White-box testing brings complexity to testing because the tester must have knowledge of the program, including being a programmer. White-box testing requires a programmer with a high level of knowledge due to the complexity of the level of testing that needs to be done.
- On some occasions, it is not realistic to be able to test every single existing condition of the application and some conditions will be untested.
- The tests focus on the software as it exists, and missing functionality may not be discovered.

7.6 SYSTEM TESTING

System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. System testing falls within the scope of black-box testing, and as such, should require no knowledge of the inner design of the code or logic. As a rule, system testing takes, as its input, all the "integrated" software components that have passed integration testing and the software system itself integrated with anyapplicable hardware system(s). The purpose of integration testing is to detect any inconsistencies between the software units that are integrated together (called assemblages) or between any of the assemblages and the hardware. System testing is a more limited type of testing; it seeks to detect defects both within the "inter-assemblages" and also within the system as a whole. System testing is performed on the entire system in the context of a Functional Requirement Specification(s) (FRS)and/or a System Requirement Specification (SRS). System testing tests not only the design, but also the behavior and even the believed expectations of the customer. It is also intended to test up to and beyond the bounds defined in the software/hardware requirements specification(s).

CHAPTER-8
FUTURE SCOPE

The future scope of emotion detection is promising, as the technology is continually evolving and advancing. Some of the potential developments and applications of emotion detection technology include:

- Improved accuracy: Emotion detection technology will continue to improve in accuracy, thanks to the development of advanced machine learning algorithms and the use of multimodal emotion recognition.

- Real-time emotion detection: Emotion detection technology will become faster and more efficient, allowing for real-time detection of emotions in various contexts, such as customer service interactions or mental health diagnosis.

- Personalized marketing: Emotion detection technology can help companies tailor their marketing strategies by analyzing customers' emotional responses to products and services.

- Mental health diagnosis: Emotion detection technology can aid in the diagnosis and treatment of mental health conditions by analyzing a patient's emotional responses and physiological signals.

- Education and training: Emotion detection technology can be used in education and training contexts to assess student engagement and improve teaching methods.

- Virtual and augmented reality: Emotion detection technology can be integrated into virtual and augmented reality experiences to create more immersive and engaging experiences.

- Ethical considerations: There will be an increasing focus on addressing the ethical considerations related to the use of emotion detection technology, including privacy concerns and the potential misuse of the technology.

- Human-computer interaction: Emotion detection systems could be integrated into virtual assistants, chatbots, and other AI-powered systems, allowing them to better understand and respond to users' emotional states.

- Law enforcement: Emotion detection systems could be used in security and law enforcement settings to monitor individuals' emotional states for signs of aggression or other potentially dangerous behaviors.

Emotion detection technology has significant potential for a wide range of applications, and its future development will continue to be driven by advancements in machine learning, natural language processing, and computer vision.
the future scope of emotion detection systems is wide-ranging and has the potential to significantly impact numerous industries and fields. As these systems continue to improve and become more sophisticated, we can expect to see increasingly innovative and transformative applications emerge.

CHAPTER-9
CONCLUSION

Emotion detection is an increasingly important field in artificial intelligence, as it can have a wide range of applications such as improving customer experience, mental health diagnosis, and personalized marketing. The field involves using various techniques such as natural language processing, computer vision, and machine learning algorithms to analyze and interpret human emotions expressed through facial expressions, speech, text, and physiological signals.

While there have been significant advancements in emotion detection technology, it still faces several challenges. For instance, emotions are complex and can be expressed differently based on cultural backgrounds, context, and personal experiences. Additionally, there are concerns about privacy and the ethical implications of using emotion detection technology in certain contexts.

Emotion detection has the potential to provide valuable insights into human behavior and improve various industries, but it is crucial to continue developing and refining the technology while also addressing concerns related to privacy and ethics.

Emotion recognition is a field of artificial intelligence that focuses on the detection and interpretation of human emotions expressed through various modalities such as facial expressions, speech, text, and physiological signals.

The potential applications of emotion recognition technology are vast, ranging from mental health diagnosis to customer experience improvement. However, the field faces challenges such as the complexity of human emotions, cultural differences, and ethical considerations related to privacy and the potential misuse of the technology.

Despite these challenges, there have been significant advancements in emotion recognition technology, thanks to the use of machine learning algorithms, natural language processing, and computer vision. Additionally, the development of multimodal emotion recognition that combines multiple sources of emotion signals is promising.

In conclusion, emotion recognition technology has the potential to improve various industries and provide valuable insights into human behavior. However, it is essential to continue refining the technology, address challenges related to privacy and ethics, and ensure that it is used responsibly and ethically.

CHAPTER 10

BIBLIOGRAPHY

Here is a bibliography for emotion detection project from where reference is taken:

1. Youtube channel named "EDUREKA" is used for learning purpose
2. Deep learning by Phd scholar.
3. Akshit madan
4. Geekyhumans.com , Towardsdatascience.com websites are used in this.
5. DSDL club of KIET.