A PROJECT REPORT ON

"Multi-Modal Emotion Detection System"

SUBMITTED TO THE SAVITRIBAI PHULE PUNE UNIVERSITY,

PUNE IN THE PARTIAL FULFILLMENT FOR THE AWARD OF THE

DEGREE OF

BACHELOR OF ENGINEERING IN INFORMATION TECHNOLOGY

BY

AKANKSHA UNDE	**BI90458582**
SHASHANK KULKARNI	**BI90458546**
HARSHADA SUTAR	**BI90458576**
PRADNYA SURYAWANSHI	**BI90458575**

UNDER THE GUIDANCE OF

GUIDE	**HEAD OF DEPARTMENT**
DR. BHARATI P. VASGI	**DR. RUPALI M. CHOPADE**

DEPARTMENT OF INFORMATION TECHNOLOGY

MARATHWADA MITRA MANDAL'S COLLEGE OF ENGINEERING

KARVENAGAR, PUNE - 411052, MAHARASHTRA, INDIA 2021-22

2022-23

CERTIFICATE

This is to certify that the Project Report entitled

MULTI-MODEL DEPRESSION DETECTION SYSTEM USING DEEP LEARNING

Submitted by

AKANKSHA UNDE	**B190458582**
SHASHANK KULKARNI	**B190458546**
HARSHADA SUTAR	**B190458576**
PRADNYA SURYAWANSHI	**B190458575**

is a bonafide work carried out by them under the supervision of **Dr Bharati P. Vasgi** and it is approved by the partial fulfillment of the requirement of Savitribai Phule Pune University for the award of the Degree of Bachelor of Engineering (Information Technology)

This project report has not been submitted to any other Institute or University for the award of any degree or diploma.

Dr. Bharati P. Vasgi
Internal Guide
Department of Information Technology

Dr. Rupali Chopade
Head of Department
Department of Information Technology

External Examiner
Date :

Dr. V. N. Gohokar
Principal
Marathwada Mitra Mandal's College of Engineering

Date :

Place :

ACKNOWLEDGEMENT

It is our proud privilege and duty to acknowledge the kind of help and guidance received from several people in the preparation of this report. It would not have been possible to prepare this report, in this report and in this form without their valuable help, co-operation and guidance.

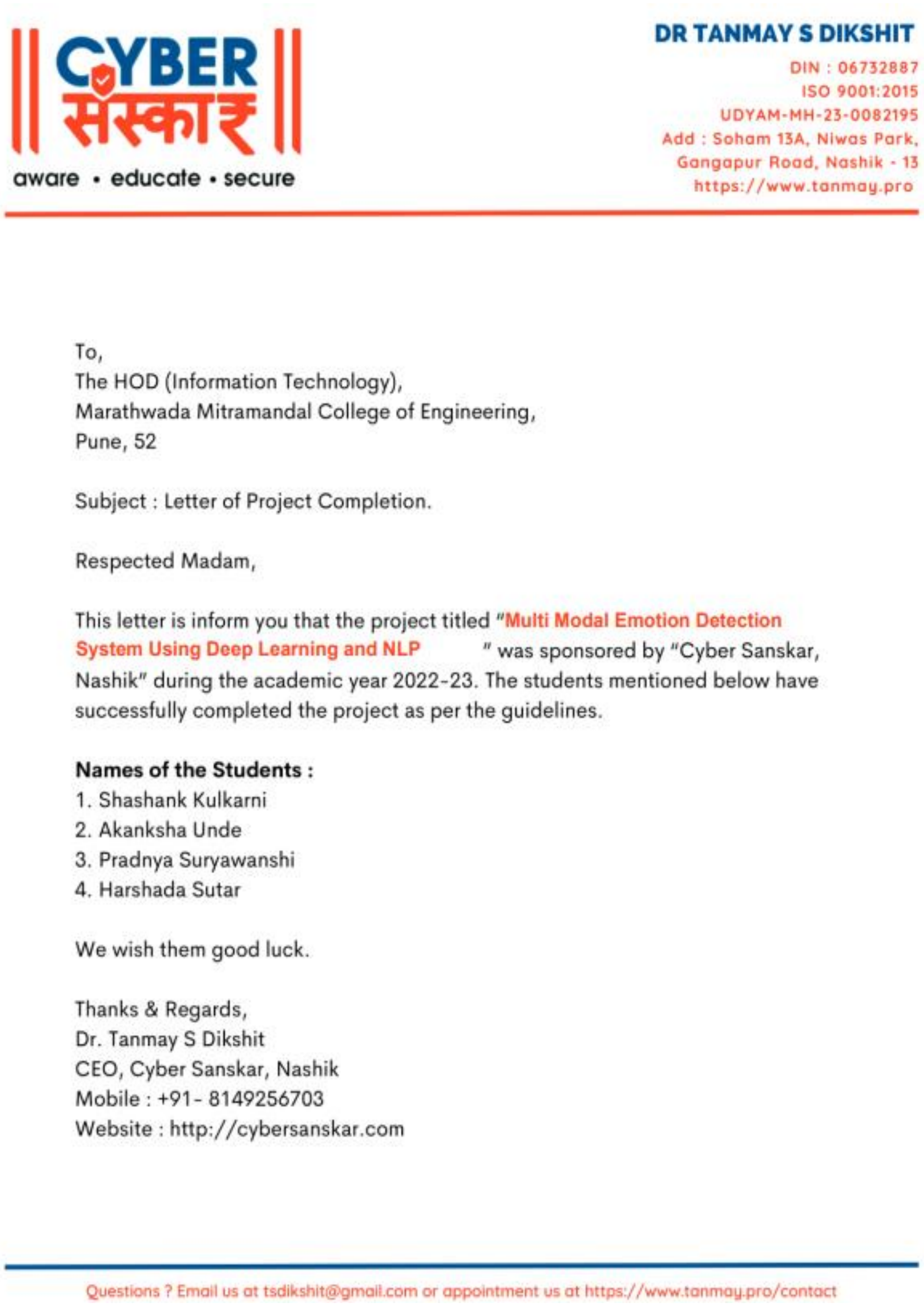
Our sincere thanks to **Dr. Rupali Chopade**, Head of Department of Information Technology, for her valuable suggestions and guidance throughout the preparation of this report.

We express our sincere gratitude to our guide, **Dr. Bharati P. Vasgi** for guiding us in the investigation of this project and in carrying out experimental work. We hold her in esteem for the guidance, encouragement and inspiration received from her.

Finally we wish to thank our parents for financing our studies and helping usthroughout our life for achieving perfection and excellence. Their personal help in making thisreport and project presentation is gratefully acknowledged.

Akanksha Unde
(B190458582)
Shashank Kulkarni
(B190458546)
Harshada Sutar
(B190458576)
Pradnya Suryawanshi
(B190458575)
(B.E. Information Technology)

SPONSORSHIP LETTER



CYBER संस्कार
aware • educate • secure

DR TANMAY S DIKSHIT
DIN : 06732887
ISO 9001:2015
UDYAM-MH-23-0082195
Add : Soham 13A, Niwas Park,
Gangapur Road, Nashik - 13
https://www.tanmay.pro

To,
The HOD (Information Technology),
Marathwada Mitramandal College of Engineering,
Pune, 52

Subject : Letter of Project Completion.

Respected Madam,

This letter is inform you that the project titled "Multi Modal Emotion Detection System Using Deep Learning and NLP " was sponsored by "Cyber Sanskar, Nashik" during the academic year 2022-23. The students mentioned below have successfully completed the project as per the guidelines.

Names of the Students :
1. Shashank Kulkarni
2. Akanksha Unde
3. Pradnya Suryawanshi
4. Harshada Sutar

We wish them good luck.

Thanks & Regards,
Dr. Tanmay S Dikshit
CEO, Cyber Sanskar, Nashik
Mobile : +91- 8149256703
Website : http://cybersanskar.com

Questions ? Email us at tsdikshit@gmail.com or appointment us at https://www.tanmay.pro/contact

ABSTRACT

Emotions are fundamental aspects of human communication and play a crucial role in our daily lives. Accurately detecting and understanding emotions can have profound implications in various domains, including human-computer interaction, mental health analysis, and personalized user experiences. However, existing emotion detection systems often focus on individual modalities, such as facial expressions or textual cues, which limit their effectiveness and real-world applicability. To address this, The Multi-Modal EmotionDetection System presented in this project integrates multi-modal data to effectively detect and classify emotions. By leveraging Convolutional Neural Networks (CNN) for real-time emotion recognition from webcam input and utilizing the Natural Language Toolkit (NLTK) for text processing and analysis, the system aims to capture a more comprehensive understanding of emotional states. The motivation behind this project stems from the need for a reliable and versatile emotion

This project encompasses several key components, including data collection, preprocessing, feature extraction,and fusion of multi-modal information. A diverse dataset comprising facial expression images and corresponding textual data is collected and annotated. The collected data is then preprocessed to ensure its quality and consistency. Feature extraction techniques tailored to each modality are employed to extract meaningful representations from the input data. To effectively combine the modlities, fusion strategies are employed, enabling the integration of facial expressions and textual features. Convolutional Neural Networks(CNNs) are trained to learn the underlying patterns and relationships within the fused data, facilitating accuracy.

CONTENT

INDEX

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Emotions play a significant role in human communication and have a profound impact on our decision-making, behavior, and overall well-being. Understanding and accurately detecting emotions are crucial in various domains, including human-computer interaction, mental health analysis, and personalized user experiences. However, traditional emotion detection systems often rely on single-modal approaches, such as analyzing facial expressions or processing textual data, which limits their effectiveness and real-world applicability. To address this limitation, this project introduces a Multi-Modal Emotion Detection System that combines facial expressions and textual data to enhance the accuracy and holistic understanding of emotions.

The motivation behind this project stems from the need for a more comprehensive and versatile emotion detection system capable of integrating multiple modalities. By leveraging Convolutional Neural Networks (CNN) for real-time emotion recognition from webcam input and utilizing the Natural Language Toolkit (NLTK) for text processing, our system aims to capture a more nuanced and context-aware understanding of emotional states. By fusing visual and textual cues, the Multi-Modal Emotion Detection System overcomes the limitations of single-modality approaches and offers a more robust and accurate emotion detection capability.

The project involves several key components, including data collection, preprocessing, feature extraction, and fusion of multi-modal information. A diverse dataset comprising facial expression images and corresponding textual data is collected and annotated. To ensure data quality and consistency, the collected data undergoes preprocessing steps. Feature extraction techniques tailored to each modality are then applied to extract meaningful representations from the input data. The fusion of facial expressions and textual features is achieved through carefully designed fusion strategies, enabling a holistic understanding of emotions.

The proposed Multi-Modal Emotion Detection System holds great potential for real-time emotion monitoring, sentiment analysis, and adaptive user interfaces. By combining real-time webcam-based facial expression analysis with textual data processing, the system can accurately recognize and categorize emotions, providing valuable insights into users' emotional states. Furthermore, this project serves as a foundation for future research and development in the field of multi-modal emotion detection.

1.1 Motivation

Overall, the motivation for this project lies in the need for a more advanced and versatile emotion detectionsystem that can accurately interpret emotions from multiple modalities. By leveraging the power of multi-modal data analysis, we aim to overcome the limitations of existing approaches and unlock the potential for significant advancements in psychology, healthcare, and human-computer interaction.

1.2 Aim & Objective

- Develop a Multi-Modal Emotion Detection System that combines facial expressions and textual data for enhanced emotion recognition and understanding.
- Overcome the limitations of single-modal emotion detection systems by leveraging multiple modalities to capture a comprehensive range of emotional cues.
- Enhance the accuracy of emotion classification by fusing and analyzing information from facial expressions and textual cues.
- Explore the potential applications of the Multi-Modal Emotion Detection System in psychology, healthcare, and human-computer interaction.
- Objectives:

- Collect and curate a diverse dataset comprising facial expression images and corresponding textual data.
- Preprocess the collected data to ensure consistency, quality, and compatibility for subsequent analysis.
- Develop feature extraction techniques tailored to each modality (facial expressions and textual cues) to extract meaningful representations.
- Implement fusion strategies to integrate and combine the modalities effectively, enabling a more comprehensive understanding of emotions.

1.3 Problem Statement

Current emotion detection systems struggle to accurately detect emotions from multiple modalities. A more sophisticated multi-modal system is needed to better interpret complex emotional cues and enable accurate classification, with potential applications in psychology, healthcare, and human-computer interaction.

CHAPTER 2
LITERATURE SURVEY

It is a critical examination and analysis of existing literature, scholarly articles, books, and other relevant sources on a specific topic. It involves gathering, evaluating, and synthesizing information from various published works to provide a comprehensive understanding of the current state of knowledge on the subject.

2.1 Tabular Data

Sr. No	Title	Author	Problem Statement	Solution	Limitation
1	A novel multimodal depression detection approach based on task-based mechanisms.	Thati, R.P., Dhadwal, A.S., Kumar	To identify depression at an early stage by mining online social behaviors.	This study investigated multi modal features extracted from MCS and task/interview-based mechanism to identify depressed and non-depressed participants. For this purpose, the user data was collected in a unique way by acquiring their smartphones usage data, emotion and speech elicitation mechanisms.	Different statistical analysis techniques and their comparative study could reveal the more scientific significance of the work

2	Cloud edge Collaborative Depression Detection using negative emotions recognition and cross scale facial feature analysis	Yao Yu et. al (IEEE transaction on Industrial Informatics), 2022	To identify an intelligent method for multi-scene automatic depression symptom Detection.	They propose a shallow model Edge-ER on the edge server and a deep model on the cloud server, using an efficient and convenient cloud-edge collaboration	The potential of multi-model data such as voice, head posture, text to improve the performance of depression detection

				framework.	
3	Depression Intensity Estimation via social media a Deep Learning Approach	Shreya Ghosh et. al (IEEE transaction on computational social system),2021	To predict depression users as well as estimate their depression Intensity.	A rich set of features are used to train a small, long short-term memory network using Swish as an activation function	Development of automatic preliminary assessment methods based on social data.
4	Early Depression Detection from Social Network Using Deep Learning Techniques	Faisal Mohammad Shah et al. (IEEE Region 10 Symposium), 2020	To detect depression in early stage by analyzing posts to save a person from depression related disease.	In this research work, a hybrid model has been proposed that can detect depression by analyzing user's textual posts.	Takes too long time to detect users as depressed.
5	Automatic Assessment of Depression Based on Visual Cues: A Systematic Review	Anastasia Pampouchidou et al (IEEE conference),2020	Visual manifestations of depression, various procedures used for data collection, and existing datasets summarized	The review outlines methods and algorithms for visual feature extraction, dimensionality reduction, decision methods for classification and regression approaches, as well as different fusion strategies.	Visual cues need to be supplemented by information from other modalities to achieve clinically useful results.

2.1 Literature Paper -Summary

1. **Thati, R.P., Dhadwal, A.S., Kumar, P. et al. A novel multi-modal depression detection approach based on mobile crowd sensing and task-based mechanisms. Multimedia Tools Application (2022).**

Problem Statement: To identify depression at an early stage by mining online social behaviors.
The paper presents a novel approach for detecting depression using multimodal data and leveraging task-based mechanisms. Depression is a complex mental health condition that can be challenging to diagnose accurately. Traditional approaches often rely on self-reporting and clinical assessments, which can be subjective and prone to biases.
The proposed approach combines multiple modalities, such as facial expressions, speech patterns, and physiological signals, to capture a more comprehensive view of an individual's emotional state. By integrating data from various sources, the model aims to improve the accuracy and reliability of depression detection.
One of the key contributions of the paper is the incorporation of task-based mechanisms. It suggests that analyzing how individuals perform specific tasks or activities can provide valuable insights into their mental well-being. The approach involves designing specific tasks that are known to elicit emotional responses and measuring the corresponding multimodal data during task performance.
The authors outline a framework that includes data collection, preprocessing, feature extraction, and classification stages. Machine learning algorithms, such as deep neural networks or support vector machines, are employed to train models on the extracted features and classify individuals into depression and non-depression categories.

2. **Cloud edge Collaborative Depression Detection using negative emotions recognition and cross scale facial feature analysis, Yao Yu et. al (IEEE transaction on Industrial Informatics), 2022**

Problem Statement: To identify an intelligent method for multi-scene automatic depression symptom Detection
This paper presents a novel approach to collaborative depression detection utilizing cloud edge computing, negative emotions recognition, and cross-scale facial feature analysis. Depression is a widespread mental health condition that often goes undiagnosed or misdiagnosed. Traditional methods rely on self-reporting or subjective assessments, leading to limited accuracy.
The proposed approach leverages cloud edge computing to distribute the computational load and enable real-time analysis. It combines negative emotions recognition techniques with cross-scale facial feature analysis to capture and analyze facial expressions associated with depression. By incorporating multiple scales of facial features, including micro-expressions and macro-expressions, a more comprehensive assessment of emotional states is achieved.
The paper outlines the system architecture, which includes data acquisition from individuals through cameras or smart devices, real-time processing at the edge using machine learning algorithms, and cloud-based collaboration for further analysis and feedback. The negative emotions recognition module utilizes computer vision and

machine learning techniques to classify emotional states based on facial expressions.
Furthermore, the cross-scale facial feature analysis incorporates deep learning models to extract features from different scales of facial expressions. These features are then combined and analyzed to provide a holistic understanding of an individual's emotional well-being and the likelihood of depression.

3. **Depression Intensity Estimation Viasocial media a Deep learning Approach, Shreya Ghosh et. al (IEEE transaction on computational social system). 2021**

   Problem Statement: To predict depression users as well as to predict depression intensity,

This paper introduces a deep learning approach for estimating the intensity of depression through social media analysis. Social media platforms have become valuable sources of information, providing insights into individuals' mental health states. This study focuses on leveraging deep learning techniques to analyze social media content and estimate the intensity of depression.
The proposed approach involves collecting and preprocessing social media data, such as posts, comments, and likes, from individuals who self-disclose their depression status. The collected data is then used to train deep learning models, such as recurrent neural networks (RNNs) or convolutional neural networks (CNNs), to learn patterns and features indicative of depression intensity.
The paper emphasizes the importance of feature representation in social media text analysis and discusses various techniques, including word embeddings, sentiment analysis, and topic modeling, to capture meaningful information from textual data.

4. **Early depression detection from social network using deep learning techniques, Faisal Mohammad Shah et al.(IEEE Region 10 Symposium), 2020**

   Problem Statement: To detect depression in early stage by analyzing posts to save patient from ealy depression diseases.

This paper presents a deep learning-based approach for early detection of depression using social network data. Social networks have become integral parts of people's lives, offering a wealth of information that can be harnessed for mental health assessment. The study focuses on leveraging deep learning techniques to analyze social network activities and identify early signs of depression.
The proposed approach involves collecting and preprocessing data from social networks, such as user posts, interactions, and network structures. Deep learning models, such as recurrent neural networks (RNNs) or graph neural networks (GNNs), are trained on this data to capture patterns and features indicative of depression.
The paper discusses various aspects of the deep learning approach, including network representation learning, sentiment analysis, and temporal modeling, to effectively extract meaningful information from social network data.

CHAPTER 3

SYSTEM REQUIREMENTS

3.1 SOFTWARE REQUIREMENTS

Operating System(OS):

An operating system (OS) is a software that serves as the foundation for managing computer hardware and software resources. It acts as an intermediary between users and the computer hardware, enabling users to interact with the system and run applications.

The primary functions of an operating system include:

Hardware management: The OS manages the computer's hardware resources such as the central processing unit (CPU), memory, disk storage, input/output devices (keyboard, mouse, monitor), and network connections. It allocates resources to different programs and ensures they operate smoothly without conflicts.

Process management: The OS handles the execution of programs or processes. It manages the creation, scheduling, and termination of processes, allowing multiple programs to run concurrently and ensuring each program gets a fair share of the CPU's processing time.

Memory management: The OS controls the allocation and deallocation of memory resources. It keeps track of which parts of the memory are in use by different programs and ensures that processes have the necessary memory to execute their tasks.

File system management: The OS provides a way to organize and manage files on storage devices such as hard drives or solid-state drives. It allows users to create, read, write, and delete files, as well as organize them into directories or folders for easier organization and retrieval.

Device management: The OS handles communication between software and hardware devices. It provides device drivers that act as intermediaries, allowing applications to interact with hardware components such as printers, scanners, or network adapters.

User interface: The OS provides a user-friendly interface for users to interact with the computer system. This can be in the form of a graphical user interface (GUI) with windows, icons, menus, and buttons, or a command-line interface (CLI) where users type commands to perform tasks.

Security: The OS includes security mechanisms to protect the computer system and its data. It provides user authentication, access control, and encryption features to safeguard against unauthorized access, malware, and other security threats.

IDE- Visual Studio Code:

Visual Studio Code (VS Code) is a popular integrated development environment (IDE) developed by Microsoft. It provides a comprehensive set of tools and features to aid software development across various programming languages and platforms. Here are some key aspects of Visual Studio Code:

Cross-platform: Visual Studio Code is available for Windows, macOS, and Linux operating systems, making it accessible to developers on different platforms.

Lightweight and customizable: Unlike some other IDEs, Visual Studio Code is lightweight and designed for efficiency. It has a minimalistic interface and allows users to customize various aspects of the editor, such as themes, keyboard shortcuts, and extensions, to tailor it to their preferences and workflow.

Support for multiple programming languages: VS Code supports a wide range of programming languages out of the box, including popular ones like JavaScript, Python, C++, Java, and many more. It provides syntax highlighting, intelligent code completion, and other language-specific features to enhance productivity.

IntelliSense: IntelliSense is a powerful code completion feature in VS Code. It offers context-aware suggestions as you type, providing information about available variables, functions, and API documentation. This helps reduce typing errors and speeds up coding.

Built-in terminal: Visual Studio Code has a built-in terminal that allows developers to execute commands directly within the IDE. This eliminates the need to switch between the editor and an external terminal window, making it convenient for running build scripts, executing commands, and interacting with the development environment.

Kaggle:

Kaggle is an online platform that hosts data science and machine learning competitions, offers datasets for practice, and provides a collaborative environment for data scientists and machine learning practitioners. It was founded in 2010 and acquired by Google in 2017.

Here are some key aspects of Kaggle:

Data science competitions: Kaggle hosts a variety of data science competitions, where individuals or teams compete to develop the best predictive models for a given problem. These competitions cover a wide range of domains, such as image classification, natural language processing, and time series analysis. Competitors submit their models and predictions, and the leaderboard ranks them based on performance metrics.

Datasets and kernels: Kaggle offers a vast collection of datasets that can be freely accessed and used for analysis, research, and practice. Users can upload and share their datasets as well. Additionally, Kaggle provides a feature called "kernels," which are interactive Jupyter notebooks that allow users to work on data analysis, modeling, and visualization in a collaborative and shareable environment.

Collaboration and learning: Kaggle provides a community-driven platform for collaboration and learning. Users can form teams, join discussions, and share insights and code with fellow data scientists. The community aspect of Kaggle allows users to learn from others, gain exposure to different techniques and approaches, and participate in forums and competitions to improve their skills.

Learning resources: Kaggle offers various learning resources, including tutorials, courses, and notebooks, to help users enhance their data science and machine learning knowledge. These resources cover a wide range of topics and skill levels, from beginner to advanced, making it a valuable platform for both newcomers and experienced practitioners.

Job opportunities and hiring: Kaggle has a job board where companies can post data science and machine learning job openings. It provides a platform for recruiters to discover and hire talented individuals based on their competition rankings, kernel contributions, and expertise showcased on the platform.

Kaggle has become a popular platform for data scientists, machine learning enthusiasts, and researchers due to its diverse set of competitions, extensive dataset library, collaborative environment, and the opportunity to learn and showcase skills in a competitive setting.
[15:04, 02/06/2023] Pradnya: Python is a high-level, interpreted programming language known for its simplicity, readability, and versatility. It was created by Guido van Rossum and first released in 1991. Python is widely used for various purposes, including web development, data analysis, scientific computing, artificial intelligence, and automation. Here are some key aspects of Python:

Readable and expressive syntax: Python's syntax is designed to be easily readable and understandable, emphasizing code readability and simplicity. It uses indentation (whitespace) to define code blocks, which enhances the visual structure of the code. Python code is often considered more readable than languages with explicit delimiters like braces.

Extensive standard library: Python comes with a large standard library that provides a wide range of modules and functions for various tasks, such as file I/O, networking, string manipulation, regular expressions, and more. The standard library reduces the need for external dependencies and makes it easier to develop applications without reinventing the wheel.

Dynamically typed: Python is a dynamically typed language, which means you don't need to declare variable types explicitly. Variables are dynamically assigned types based on the assigned values. This dynamic nature allows for flexibility but also requires careful handling to prevent potential type-related errors.

Object-oriented programming (OOP) support: Python supports object-oriented programming principles, allowing developers to define classes and create objects. It provides features such as inheritance, polymorphism, and encapsulation, enabling modular and reusable code.

Large ecosystem of libraries and frameworks: Python has a vast ecosystem of third-party libraries and frameworks that extend its functionality. For example, NumPy and pandas are popular libraries for scientific computing and data analysis, Django and Flask are widely used for web development, and TensorFlow and PyTorch are prominent frameworks for machine learning and deep learning.

Cross-platform compatibility: Python is available on various platforms, including Windows, macOS, Linux, and more. Python code written on one platform can often run on another without significant modifications, making it highly portable.

Interpretation and scripting: Python is an interpreted language, which means that the code is executed line by line without prior compilation. This allows for rapid development and testing, as code changes can be immediately executed. Python is also commonly used for scripting tasks, where it can automate repetitive operations or act as a glue language to connect different components of a system.

Python's popularity has grown steadily over the years due to its simplicity, versatility, and extensive community support. It is widely adopted in various industries and domains, and its ease of use makes it an excellent choice for beginners entering the world of programming.

Libraries used:

Keras: Keras is a high-level deep learning library that provides a user-friendly interface for building and training neural networks. It is built on top of other low-level frameworks, such as TensorFlow, Theano, or CNTK, and allows you to quickly prototype and develop deep learning models. Keras offers a wide range of pre-built layers, optimizers, and loss functions, making it easier to design and train neural networks.

NumPy: NumPy is a fundamental library for numerical computing in Python. It provides support for large, multi-dimensional arrays and matrices, along with a collection of mathematical functions to operate on these arrays efficiently. NumPy is widely used in scientific computing, data analysis, and machine learning tasks as it offers fast numerical operations and advanced mathematical functions.

Pandas: Pandas is a powerful library for data manipulation and analysis. It provides data structures such as DataFrame, which allows you to store and manipulate structured data in a tabular form. Pandas provides functionalities to load, clean, transform, and analyze data, making it a popular choice for data preprocessing and exploratory data analysis tasks.

TensorFlow: TensorFlow is an open-source deep learning framework developed by Google. It provides a comprehensive ecosystem of tools, libraries, and resources for building and deploying machine learning models. TensorFlow supports both high-level APIs like Keras, as well as low-level APIs that give you more control over the model's architecture and training process. It includes a wide range of pre-built neural network layers, optimizers, and utilities for efficient computation on both CPUs and GPUs.

NLTK: NLTK (Natural Language Toolkit) is a library for natural language processing (NLP) in Python. It provides a wide range of tools and resources for tasks like tokenization, stemming, tagging, parsing, and semantic reasoning. NLTK is widely used in academic and research settings for exploring linguistic data and building NLP models.

Matplotlib: Matplotlib is a plotting library in Python that provides a wide variety of visualization options. It allows you to create line plots, scatter plots, bar plots, histograms, and more. Matplotlib is highly customizable, enabling you to control aspects like colors, labels, annotations, and axes. It is commonly used for data visualization and generating publication-quality plots.

OpenCV: OpenCV (Open Source Computer Vision Library) is a popular computer vision library that provides tools and algorithms for image and video processing. It includes functions for image manipulation, feature detection, object recognition, and more. OpenCV is widely used in computer vision applications, including robotics, surveillance, and augmented reality.

These libraries play crucial roles in different areas of data analysis, machine learning, deep learning, natural language processing, and computer vision, providing essential tools and functions to support various tasks and workflows.

Operating System	Windows 7 and above
RAM	4GB or Higher
Language	Python
IDE	VS-Code, Kaggle

Table 2: Software Requirements

3.2 HARDWARE REQUIREMENTS

System Type:

32-bit System: A 32-bit system, also known as x86 (or IA-32), is a type of computer architecture that operates on data units that are 32 bits wide. This architecture has a memory address space that can directly access up to 4 gigabytes (GB) of RAM. In a 32-bit system, each process is limited to a maximum of 4 GB of memory.

64-bit System: A 64-bit system, also known as x64 (or IA-64), is a computer architecture that operates on data units that are 64 bits wide. This architecture has a much larger memory address space, capable of directly accessing up to 18.4 million terabytes (TB) of RAM.

Processor:

The Intel Core i5 processor is a popular line of CPUs produced by Intel Corporation. The specific model you mentioned has a base clock speed of 2 GHz. Here's an explanation of what that means:

Core i5: The Core i5 is a mid-range processor line within Intel's product lineup. It offers a good balance between performance and cost, suitable for a wide range of tasks including general computing, multimedia, and moderate gaming.

Clock speed: The clock speed refers to the rate at which the processor's internal clock cycles, measured in gigahertz (GHz). In the case of the processor you mentioned, it has a base clock speed of 2 GHz. This means that the processor's cores can execute instructions at a frequency of 2 billion cycles per second.

Turbo Boost: Many Intel Core i5 processors also include a feature called Turbo Boost. Turbo Boost allows the processor to dynamically increase its clock speed above the base frequency when there is a demand for more processing power. The exact turbo frequencies vary depending on the specific model.

Cores and threads: The Core i5 processors typically have multiple cores, which are independent processing units within the CPU. Each core can execute tasks simultaneously, providing parallel processing capabilities. The number of cores in a Core i5 processor may vary depending on the specific model. Additionally, each core can handle multiple threads, allowing for efficient multitasking.

Performance: The performance of a processor depends on various factors, including clock speed, the number of cores, architecture, cache size, and other architectural enhancements. While clock speed is an important factor, it's not the sole determinant of overall performance. The specific model of the Core i5 processor, along with its generation, will have a significant impact on its performance capabilities.

It's worth noting that the performance of a processor is not solely determined by clock speed or the specific model. Other factors such as the software being run, system memory (RAM), storage type, and the overall system configuration also play a role. If you require more detailed information about the specific processor model you have, it's advisable to refer to the official Intel documentation or consult the specifications provided by the computer manufacturer.

I/O Devices:

Input/output (I/O) devices are peripheral devices that allow users to interact with a computer system by providing input or receiving output. These devices enable the exchange of data and information between the computer and the external world. Here are some common examples of I/O devices:

Keyboard: A keyboard is an input device that allows users to enter alphanumeric characters, symbols, and commands by pressing keys. It is one of the primary input devices for text-based input.

Mouse: A mouse is a pointing device that allows users to move a cursor or pointer on the screen and select or interact with objects and graphical user interface elements through clicks and movements.

Monitor/Display: A monitor or display is an output device that presents visual information generated by the computer. It shows text, images, videos, and graphical user interfaces to users.

Printer: A printer is an output device that produces hard copies of documents or images from the computer onto paper or other printable materials.

Scanner: A scanner is an input device that captures images, documents, or photographs and converts them into digital format, allowing the computer to process and store the information.

Speakers/Headphones: Speakers and headphones are output devices that produce sound and allow users to listen to audio generated by the computer, such as music, system sounds, or voice recordings.

Webcam: A webcam is an input device that captures video and audio, allowing users to conduct video conferencing, record videos, or take pictures.

Microphone: A microphone is an input device that captures audio or voice, allowing users to input sound into the computer, such as for voice recognition, voice recording, or online communication.

External Storage Devices: External storage devices, such as USB flash drives, external hard drives, and memory cards, provide additional storage capacity and allow users to transfer data between computers.

System Type	64 bit or 32 bit
Processor	Intel core i5. 2GHz
Storage capacity	256 GB
I/O Devices	Mouse, Keyboard and WebCam

Table 3: Hardware Requirements

CHAPTER 4
SYSTEM DESIGN

Requirement analysis, also known as requirements engineering or gathering, is a crucial phase in the Software Development Life Cycle (SDLC). It involves understanding and documenting the needs, expectations, and constraints of stakeholders to define the system requirements accurately. Here's an explanation of the requirement analysis phase in the SDLC:

4.1 SDLC Model

Identifying Stakeholders:

The first step in requirement analysis is identifying the stakeholders who will be involved in the development process. Stakeholders can include clients, end-users, business analysts, subject matter experts, and other individuals or groups with a vested interest in the software project.

Gathering Requirements:

Requirements gathering involves collecting information about the desired functionalities, features, and constraints of the software system. This can be done through various techniques, such as interviews, workshops, surveys, and document analysis. The goal is to elicit as much relevant information as possible from stakeholders to capture their needs accurately.

Requirements Documentation:

Once the requirements are gathered, they need to be documented in a structured manner. This includes creating requirement documents or specifications that describe the functional and non-functional aspects of the software. Requirements documentation serves as a reference for the development team and ensures a common understanding among stakeholders.

Requirements Analysis and Prioritization:

After gathering the requirements, they are analyzed to identify any inconsistencies, ambiguities, or conflicts. This step involves validating the requirements and ensuring they are complete, consistent, and feasible. Additionally, requirements are prioritized based on their importance and criticality to the system. This helps in making informed decisions during the development process.

Requirements Validation:

Requirements validation involves reviewing and verifying the documented requirements to ensure they meet the stakeholders' needs and expectations. This can be done through techniques such as requirements walkthroughs, inspections, or prototyping. The goal is to identify any gaps, errors, or misunderstandings in the requirements early on, reducing the likelihood of costly changes during later stages of the SDLC.

Requirements Management:

Throughout the SDLC, requirements may evolve due to changing business needs, new insights, or external factors. Requirements management involves tracking and controlling changes to the requirements, ensuring that they are properly communicated, reviewed, and approved by stakeholders. This helps in maintaining a clear understanding of the project scope and managing potential risks.

Planning :

Planning is a crucial phase in the Software Development Life Cycle (SDLC) that lays the foundation for a successful software development project. It involves defining project goals, determining project scope, estimating resources and timelines, and creating a comprehensive plan for the entire development process. Here's an explanation of the planning phase in the SDLC:

Defining Project Goals and Objectives:

The planning phase begins by clearly defining the goals and objectives of the software development project. This includes understanding the business requirements, identifying the problem to be solved, and defining the expected outcomes or deliverables. It is essential to have a clear understanding of the project's purpose and align it with the organization's overall objectives.

Project Scope Definition:

The project scope defines the boundaries of the software development project, including what is included and what is excluded. It identifies the features, functionalities, and constraints of the software system to be developed. The scope statement helps manage stakeholders' expectations, set project boundaries, and ensure that the project stays focused on its intended goals.

Resource Estimation and Allocation:

In the planning phase, it is necessary to estimate and allocate the required resources for the project. This includes determining the human resources (developers, testers, project managers), infrastructure (hardware, software, development tools), and other resources (budget, time) needed for the project. Resource estimation helps in creating a realistic project plan and ensures that the necessary resources are available to execute the project successfully.

Timeline and Milestone Planning:

A project plan should include a timeline with specific milestones and deliverables. Milestones are key points in the project timeline that mark the completion of significant phases or achievements. They help track progress, ensure timely completion of tasks, and provide opportunities for evaluation and feedback. The timeline and milestones should be realistic and consider dependencies, potential risks, and the availability of resources.

Risk Assessment and Mitigation:

Risk assessment is an essential part of the planning phase. It involves identifying potential risks and uncertainties that may impact the project's success. Risks can be related to technology, resources, stakeholders, external factors, or any other aspect of the project. Once risks are identified, appropriate mitigation strategies and contingency plans can be developed to minimize their impact and increase the chances of project success.

Communication and Stakeholder Engagement:

Planning also includes establishing effective communication channels and engaging with stakeholders throughout the project. Clear and regular communication helps manage expectations, gather feedback, and address concerns or changes in requirements. It is important to involve stakeholders in the planning phase to ensure their buy-in and alignment with the project goals.

Project Documentation:

Throughout the planning phase, it is essential to create and maintain project documentation. This includes the project plan, scope statement, resource allocation, risk assessment, and any other relevant documentation. Proper documentation ensures that the project plan is well-documented, accessible to all stakeholders, and serves as a reference throughout the software development lifecycle.

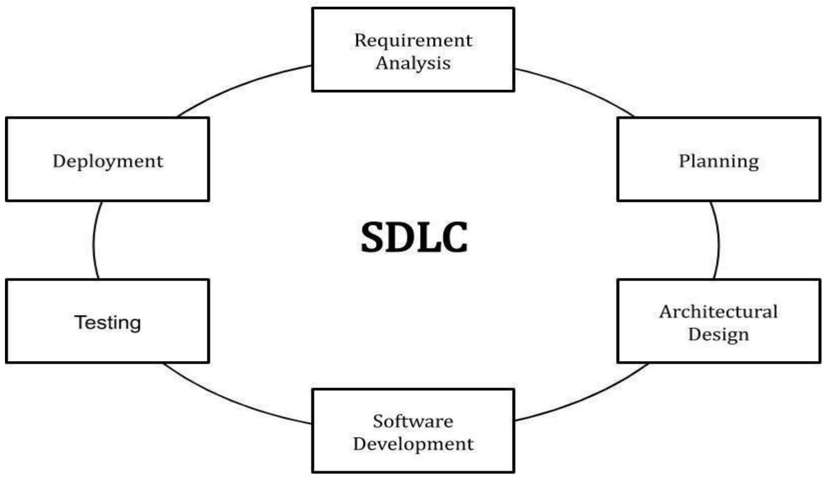


Figure 1 : SDLC Model

4.2 System Design

- The Multi-Modal Emotion Detection System is designed to integrate computer vision and natural language processing techniques for real-time emotion detection from video and text inputs. The architecture comprisestwo main components: a Convolutional Neural Network (CNN) for video analysis and the Text2Emotion algorithm for text analysis. The outputs of these components are then combined and processed to provide a comprehensive emotional analysis of the input.

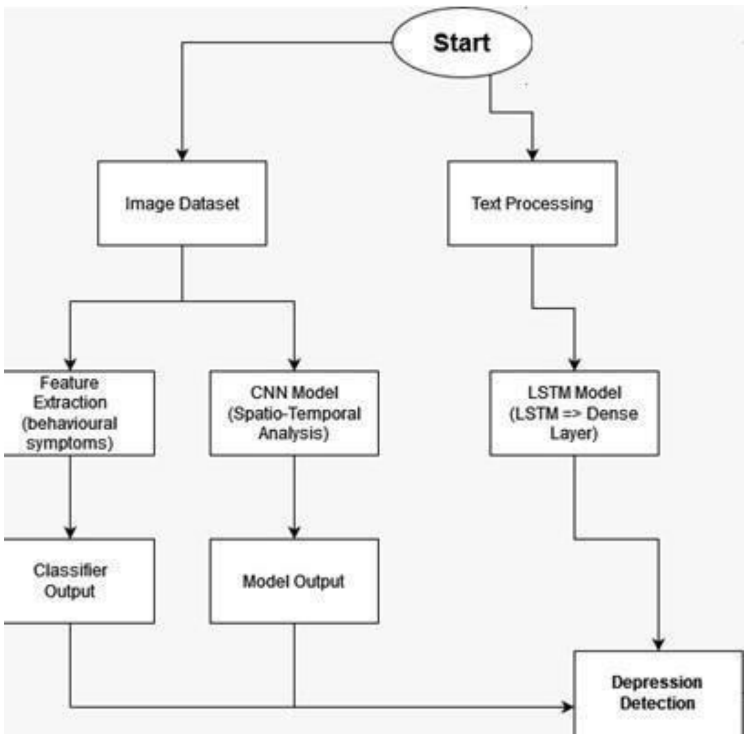

Start
Image Dataset
Text Processing
Feature Extraction (behavioural symptoms)
CNN Model (Spatio-Temporal Analysis)
LSTM Model (LSTM => Dense Layer)
Classifier Output
Model Output
Depression Detection

Figure 2: System Architecture

Video Analysis Component:

- The video analysis component utilizes a Convolutional Neural Network (CNN) architecture specifically designed for facial expression recognition.
- The system captures video input in real-time from a webcam or video source and preprocesses the frames to extract facial regions of interest.
- The CNN model takes the preprocessed facial images as input and applies convolutional, pooling, and fully connected layers to learn and extract relevant features.
- The trained CNN model then classifies the facial expressions into different emotion categories such as

happiness, sadness, anger, and surprise.

- The output of the video analysis component provides real-time emotion predictions based on the detected facial expressions.

Text Analysis Component:

- The text analysis component utilizes the Text2Emotion algorithm, which applies natural language processing techniques to analyze textual input.
- Textual data, such as user-generated text or social media posts, is preprocessed to remove noise and irrelevant information.
- The Text2Emotion algorithm then employs techniques like tokenization, sentiment analysis, and keyword matching to extract emotional cues from the text.
- By analyzing the sentiment, emotional keywords, and contextual information within the text, the algorithm determines the emotional states expressed in the text.
- The output of the text analysis component provides emotion labels associated with the input text.

The integration of computer vision and natural language processing techniques in the system architecture enables a comprehensive understanding of emotions by leveraging both visual and textual cues. By combining and processing the outputs of the video analysis and text analysis components, the system offers a powerful tool for real-time emotion detection and analysis across multiple modalities.

4.3 DATA FLOW DIAGRAM

A Data Flow Diagram (DFD) is a graphical representation that shows the flow of data within a system. It depicts how data moves from input sources to output destinations through various processes. Level 0 DFD provides an overview of the entire system by showing the interactions between the main external entities and the system itself.

The diagram represents the entire system's flow of data and how each module communicates with theother. It starts with the user providing input as text or image. The system will check the text or image and accordingly respond and will give the output.

4.3.1 DFD LEVEL 0

Level 0 DFD for a user interacting with a depression detector system. The purpose of this system is to analyze user input and determine if the user might be experiencing symptoms of depression. Here are the main components of the Level 0 DFD.

User: The user is the external entity that interacts with the depression detector system. They provide input and receive output from the system.

Depression Detector System: This is the main system being represented. It takes user input and processes it to identify potential signs of depression. The system includes various processes and data stores to perform its tasks.

Input Data: The user provides input data to the system, which typically consists of text, voice, or other forms of communication. This input may include responses to specific questions or prompts related to their mental health.

Output Data: The depression detector system produces output data that provides information about the user's potential depression status. This output can be in the form of a simple message or a more detailed report.

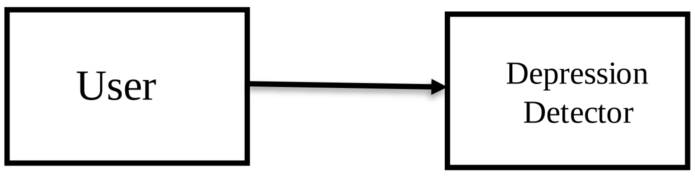


Figure 3: DFD 0

4.3.2 DFD LEVEL 1

DFD level 1 Multimodal machine learning refers to the use of multiple modalities or sources of data to improve the performance of machine learning models. A modality refers to a distinct type of data, such as text, speech, images, videos, or sensor readings. By combining information from different modalities, multimodal machine learning aims to leverage the complementary nature of these data sources to enhance the accuracy, robustness, and understanding of the models.

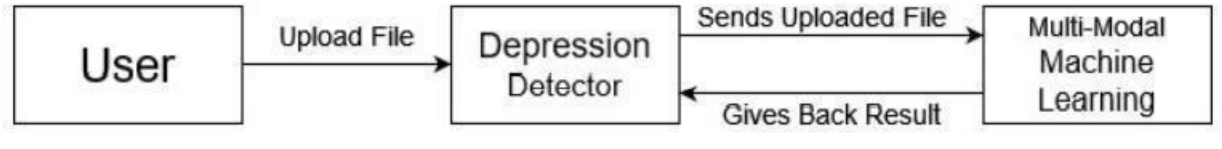


Figure 4: DFD 1

4.3.3 DFD LEVEL 2

A Convolutional Neural Network (CNN) is a type of deep learning model specifically designed for processing and analyzing visual data, such as images or videos. CNNs have gained significant popularity and achieved state-of-the-art performance in various computer vision tasks, including image classification, object detection, and image segmentation. Here's an explanation of the key components and working principles of a CNN model:

Convolutional Layers: CNNs employ convolutional layers as their primary building blocks. These layers apply a set of learnable filters (also called kernels) to the input data in order to extract meaningful features. Convolution involves sliding the filters over the input data and computing element-wise multiplications and summations, which capture spatial relationships and local patterns in the data.

LSTM stands for Long Short-Term Memory, which is a type of recurrent neural network (RNN) architecture. LSTM models are designed to address the vanishing gradient problem in traditional RNNs and are particularly effective in handling sequential data with long-term dependencies. Here's an explanation of the key components and workings of an LSTM model:

Sequential Data and Time Steps: LSTM models are suitable for processing sequential data, such as time series data, natural language text, or speech signals. The data is divided into multiple time steps, where each time step represents an individual element in the sequence.

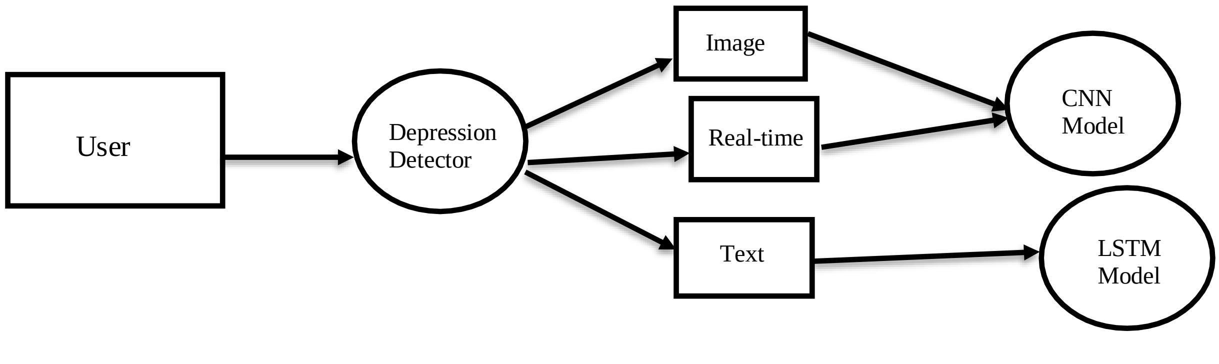


Figure 5: DFD 2

4.4 UML Diagram

5.3.1 Use Case Diagram

A use case diagram is a type of behavioral diagram in the field of software engineering. It is used to depict the interactions between actors (users or external systems) and a system or a software application. Use case diagrams provide a high-level view of the system's functionality and illustrate the various ways in which users interact with it.

The main components of a use case diagram are:

Actors: An actor represents a role or a user interacting with the system. Actors can be individuals, other systems, or even external entities. They are usually depicted as stick figures or labeled boxes. Actors are connected to use cases to show their involvement in the system.

Use Cases: A use case represents a specific functionality or behavior of the system from the user's perspective. It describes a sequence of steps or interactions between the user and the system to achieve a specific goal. Use cases are depicted as ovals or labeled ellipses. They are connected to actors to show which actors are involved in each use case.

Relationships: Relationships in a use case diagram define the associations between actors and use cases. The main relationship types are:

Association: It represents a general connection between an actor and a use case. It indicates that the actor is involved in the use case but does not specify the nature of the involvement.

Generalization: It represents an inheritance relationship between use cases or actors. It indicates that the child use case or actor inherits the behavior of the parent use case or actor

Include: It represents a relationship between two use cases where one use case includes the behavior of another use case. It is used when a common sequence of steps is shared by multiple use cases.

Extend: It represents a relationship between two use cases where one use case extends the behavior of another use case. It is used when additional steps are optional and can be added to the base use case.

Use case diagrams are often used in the early stages of software development to capture and communicate the functional requirements of a system. They help stakeholders, including developers, designers, and users, to understand the system's behavior, identify key functionalities, and validate the system's requirements. Use case diagrams provide a visual representation of the system's interactions, making it easier to analyze and design the system's architecture and user interface.

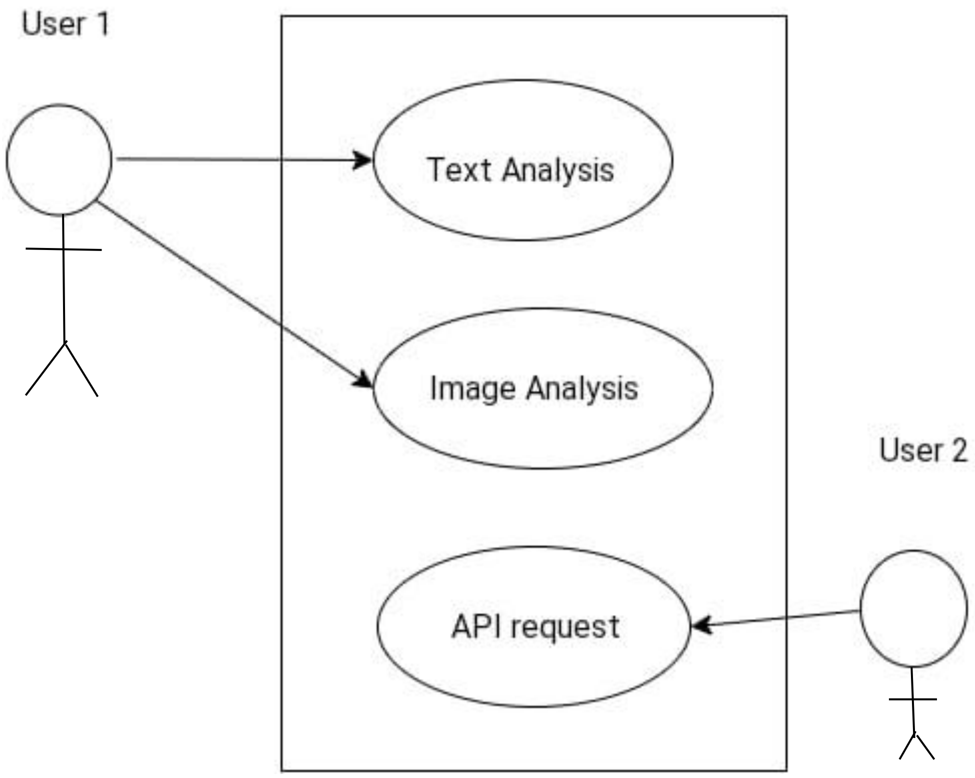


Figure 6: Use Case Diagram

4.3.2 Activity Diagram

An activity diagram is a type of UML (Unified Modeling Language) diagram that represents the flow of activities or processes within a system, such as a software application or a business process. It visually depicts the sequence of actions, decisions, and control flows that occur from the start to the end of a particular activity or process.

Here are the main elements and concepts used in an activity diagram:

Initial Node: This is represented by a filled-in circle and indicates the starting point of the activity or process.

Activity or Action: An activity is represented by a rectangular shape with rounded corners and represents a specific action or task within the system. It can be a simple action like "Calculate total" or a more complex action like "Validate user input."

Decision Node: This is represented by a diamond shape and represents a decision point where the flow branches based on a condition or a set of conditions. It typically has multiple outgoing arrows indicating different possible paths.

Control Flow Arrow: These arrows show the flow of control or the sequence of activities from one node to another. They indicate the order in which actions or decisions are executed.

Merge Node: A merge node is represented by a diamond shape with a plus sign inside it and is used to merge multiple control flows back into a single flow. It is often used to indicate that multiple parallel activities or paths converge into a single path.

Fork Node: A fork node is represented by a black bar that splits a single control flow into multiple parallel flows. It indicates that multiple activities can occur simultaneously or in parallel.

Join Node: A join node is represented by a black bar that merges multiple parallel control flows back into a single flow. It is used to synchronize the execution of parallel activities.

Final Node: This is represented by a filled-in circle with a hollow circle inside it and indicates the end of an activity or process.

Activity diagrams are helpful in visualizing complex workflows, identifying dependencies between activities, and understanding the overall structure and behavior of a system. They are widely used in software development, business process modeling, and system analysis and design to communicate and document the flow of activities or processes.

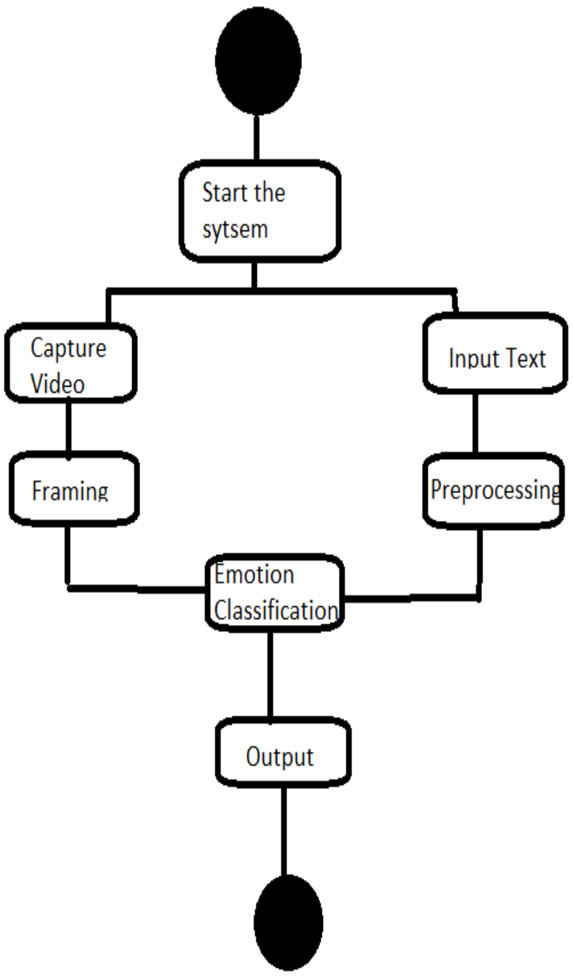


Figure 8: Activity Diagram

4.3.3 Sequence Diagram

A sequence diagram is a type of interaction diagram in Unified Modeling Language (UML) that illustrates the flow of messages or interactions between objects or components within a system. It represents the dynamic behavior of a system, emphasizing the order in which messages are exchanged and the collaboration between different entities.

In a sequence diagram, objects or participants are represented as vertical lifelines, and the messages exchanged between them are depicted as horizontal arrows. The sequence of messages is usually organized chronologically from top to bottom, with time progressing downwards.

The basic elements of a sequence diagram include:

Lifelines: These represent the objects or components that participate in the interactions. Each lifeline is depicted as a vertical line, often with the name of the object or component written at the top.

Messages: Messages are the communication channels between lifelines. They can be synchronous (blocking) or asynchronous (non-blocking). Synchronous messages are depicted as solid arrows, while asynchronous messages are shown with dashed arrows. Messages may also include parameters and return values.

Activation Boxes: Activation boxes represent the period of time during which an object is executing a particular message or operation. They are depicted as rectangles on the lifeline and show the duration of the object's activity.

Combined Fragments: Combined fragments allow you to specify different types of control flow, such as loops, conditionals, and parallel execution, within the sequence diagram. They help to capture more complex behaviors and decision-making processes.

Sequence diagrams are commonly used in software development to visualize and understand the interactions between objects or components in a system, especially during the analysis and design phases. They provide a clear representation of how different parts of a system collaborate and communicate, facilitating the identification of potential design issues, bottlenecks, or missing interactions.

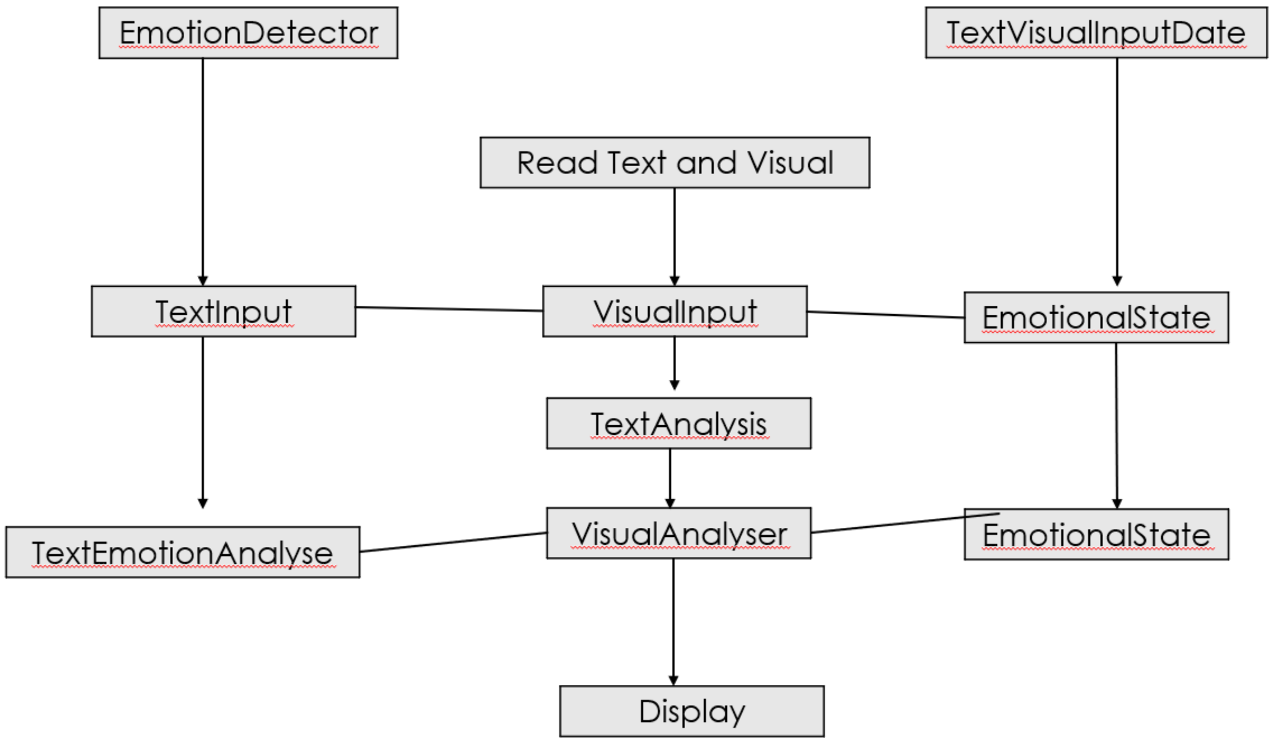


Figure 9: Sequence Diagram

CHAPTER 5
IMPLEMENTATION

To implement a depression detection system using Convolutional Neural Networks (CNNs) and text classification, you can follow these steps:

Data Collection:

Gather a dataset of text samples from social media platforms or other sources where individuals share their thoughts or experiences related to depression. Ensure that the dataset includes labels indicating whether each sample is depressive or non-depressive.

Data Preprocessing:

Clean and preprocess the text data by removing noise, punctuation, and irrelevant information. Apply techniques like tokenization, stemming, and stop-word removal to normalize the text and improve the quality of the input.

Text Representation:

Convert the preprocessed text data into numerical vectors that can be fed into the CNN model. Common approaches include one-hot encoding, word embeddings (e.g., Word2Vec or GloVe), or using pre-trained language models like BERT or GPT.

CNN Architecture:

Design the CNN model for text classification. This typically involves stacking layers such as convolutional, pooling, and fully connected layers. The convolutional layers learn local patterns and features from the text, while pooling layers downsample the learned features. The fully connected layers perform the final classification.

Model Training:

Split the dataset into training and validation sets. Feed the training data into the CNN model and optimize its parameters using backpropagation and gradient descent. Monitor the model's performance on the validation set and adjust hyperparameters as needed.

Model Evaluation:

Evaluate the trained CNN model on a separate test dataset to measure its performance. Calculate metrics such as accuracy, precision, recall, and F1 score to assess the model's effectiveness in classifying depressive and non-depressive texts.

Deployment and Prediction:

Deploy the trained model to classify new, unseen text samples. Preprocess the incoming text data using the same steps as before and feed it into the trained CNN model to obtain predictions of depression or non-depression.

5.1 CNN Model

The proposed Multi-Modal Emotion Detection System incorporates a Convolutional Neural Network (CNN) for analyzing video inputs. CNNs are specifically designed for image analysis tasks, automatically extracting features through a process called convolution. This involves sliding small filters or kernels over the input image, performing element-wise multiplication, and summation operations to generate feature maps. Non-linear activation functions are applied to the feature maps to extract high-level representations of the input image.

The CNN architecture in our system consists of four CNN layers, each followed by batch normalization, ReLU activation, max pooling, and dropout regularization layers. This architecture is chosen to capture and learn the hierarchical representations of facial expressions from the input video frames. The output from the fourth CNN layer is flattened to a one-dimensional vector.

Next, the flattened features are passed through two fully connected layers. Each fully connected layer is followed by batch normalization, ReLU activation, and dropout regularization layers. The fully connected layers further process the learned features and capture the complex relationships between them. Finally, the output layer is a fully connected layer with softmax activation, enabling multi-class classification with the number of classes set to 7, representing different emotion categories.

During the implementation, the CNN model is compiled using the Adam optimizer with a learning rate of 0.0001. Adam is an adaptive optimization algorithm that adjusts the learning rate for each parameter during training. The categorical cross-entropy loss function is utilized, which is suitable for multi-class classification tasks. Additionally, accuracy is used as the evaluation metric to assess the performance of the model.

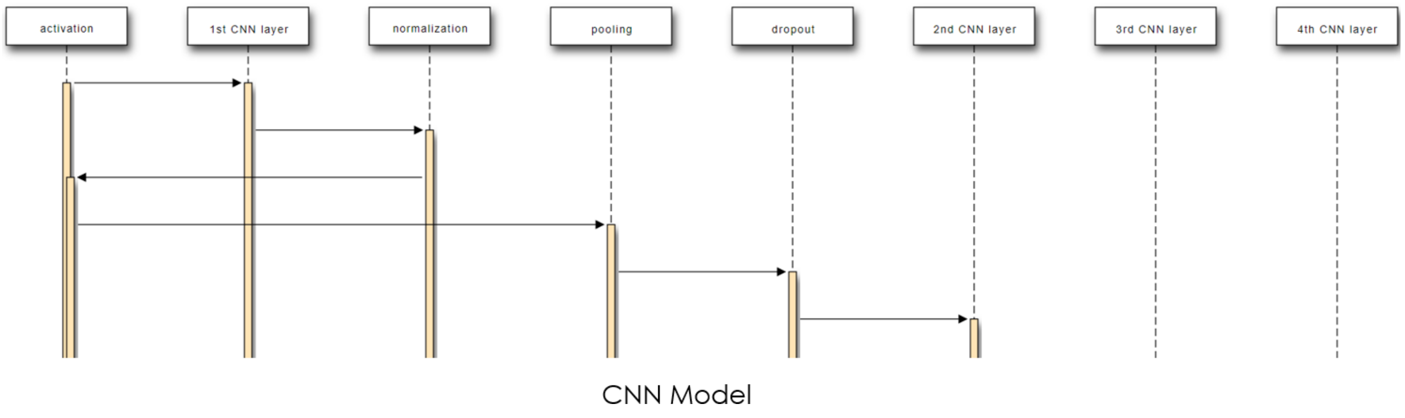


Figure 10: CNN Model

5.2 Text Processing

The text analysis component of our Multi-Modal Emotion Detection System incorporates the "Text2Emotion" algorithm, which plays a crucial role in analyzing textual data to extract underlying emotional cues. This algorithm leverages natural language processing techniques to gain insights into the emotional content expressed within the text. By understanding the emotions conveyed through text, we can enhance the system's ability to interpret complex emotional cues and achieve accurate emotion classification.

The working of the "Text2Emotion" algorithm involves several key steps. Firstly, the input text undergoes preprocessing to eliminate noise and irrelevant information, ensuring that the subsequent analysis focuses on the essential content. This preprocessing phase includes removing punctuation, converting text to lowercase, and removing stop words.

Next, the algorithm applies sentiment analysis to determine the overall sentiment expressed in the text. Sentiment analysis enables the algorithm to gauge whether the text conveys a positive, negative, or neutral sentiment. By capturing the sentiment, the algorithm establishes a foundational understanding of the emotional tone within the text.

Furthermore, the "Text2Emotion" algorithm utilizes keyword matching techniques to identify emotional keywords and expressions present in the text. These keywords act as indicators for specific emotions and aid in associating the text with relevant emotion categories. By comparing the keywords in the text with a predefined list of emotional terms, the algorithm identifies the emotions expressed in the input text.
To refine the emotional analysis, the algorithm considers the contextual information surrounding the

emotional keywords. By examining the words and phrases in proximity to these keywords, the algorithm gains deeper insights into the emotional context. This contextual analysis assists in disambiguating and improving the accuracy of emotion classification.

The output of the "Text2Emotion" algorithm provides emotion labels associated with the input text, representing the dominant emotions expressed and their corresponding intensities. These emotion labels contribute to the comprehensive emotional analysis generated by our Multi-Modal Emotion Detection System. By integrating the outputs of both the video analysis component and the text analysis component, our system achieves a more accurate and holistic understanding of emotions, harnessing both visual and textual cues for improved emotion detection and classification.

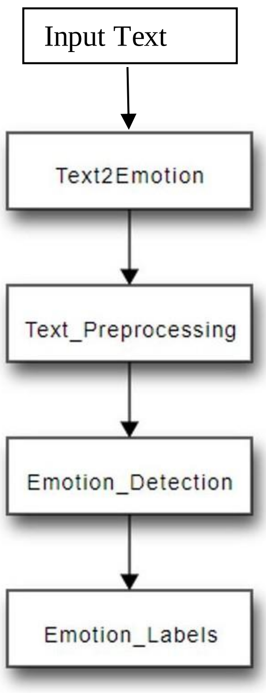


Figure 11: Text Processing

5.3 ALGORITHM

The algorithm for our Multi-Modal Emotion Detection System encompasses two key components: Video Analysis and Text Analysis. These components work in tandem to extract emotional cues from both visual and textual inputs, enabling a comprehensive understanding of emotions expressed in real-time.

In the Video Analysis component, real-time video processing techniques are employed to capture facial expressions from the webcam feed. The algorithm utilizes computer vision algorithms to detect and track faces in each frame, allowing for precise analysis of facial features. A pre-trained Convolutional Neural Network (CNN) model, specifically designed for facial expression analysis, is then employed. This CNN model leverages its learned parameters to extract facial landmarks, such as eye movements, eyebrow positions, and mouth shape. These extracted features are subsequently fed into the CNN model to predict the probabilities of different emotions being expressed in each frame.

On the other hand, the Text Analysis component focuses on analyzing the emotional content present in the textual input. The algorithm first preprocesses the text by removing noise, punctuation, and converting it to lowercase. Sentiment analysis techniques are then applied to determine the overall sentiment expressed in the text. Additionally, keyword matching is utilized to identify emotional keywords and expressions within the text, providing valuable insights into the emotional context.

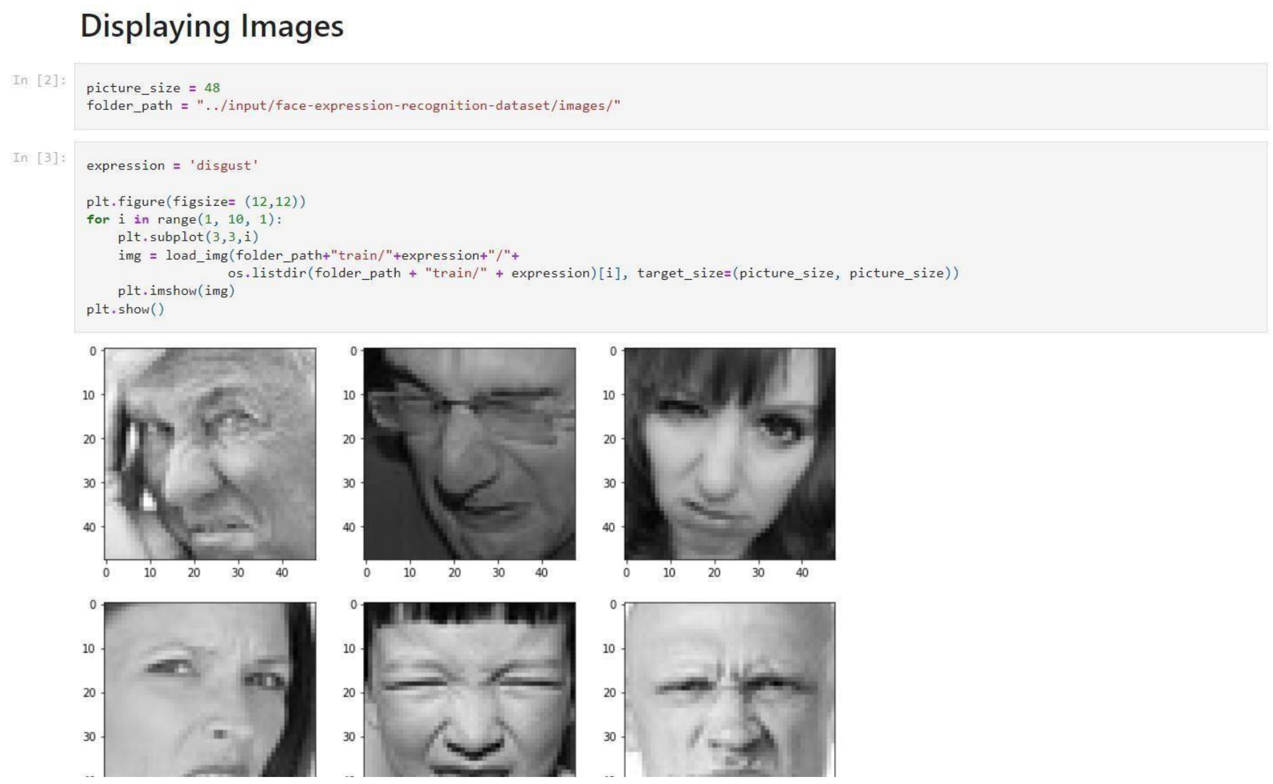
By integrating the outputs of the Video Analysis and Text Analysis components, our system generates a holistic understanding of emotions expressed in real-time. The video analysis component captures the emotions portrayed through facial expressions, while the text analysis component uncovers emotions conveyed through written language. This integrated approach enhances the accuracy and richness of emotion classification in our Multi-Modal Emotion Detection System, enabling a more comprehensive analysis of emotions across multiple modalities.

```
# Import the necessary library

# Function to analyze emotions in text
function analyze_emotions(text):
    # Use the text2emotion library to analyze emotions in the text
    emotions = te.get_emotion(text)

    # Print or display the detected emotions
    print("Emotions detected:")
    print(emotions)

# Main program
function main():
    # Get user input or load text data
    text = input("Enter the text: ")

    # Analyze emotions in the text
    analyze_emotions(text)

# Call the main program
main()
```

CHAPTER 6
RESULT ANALYSIS

6.1 SCREENSHOTS

```
expression = 'disgust'

plt.figure(figsize= (12,12))
for i in range(1, 10, 1):
    plt.subplot(3,3,i)
    img = load_img(folder_path+"train/"+expression+"/"+
                  os.listdir(folder_path + "train/" + expression)[i], target_size=(picture_size, picture_size))
    plt.imshow(img)
plt.show()
```

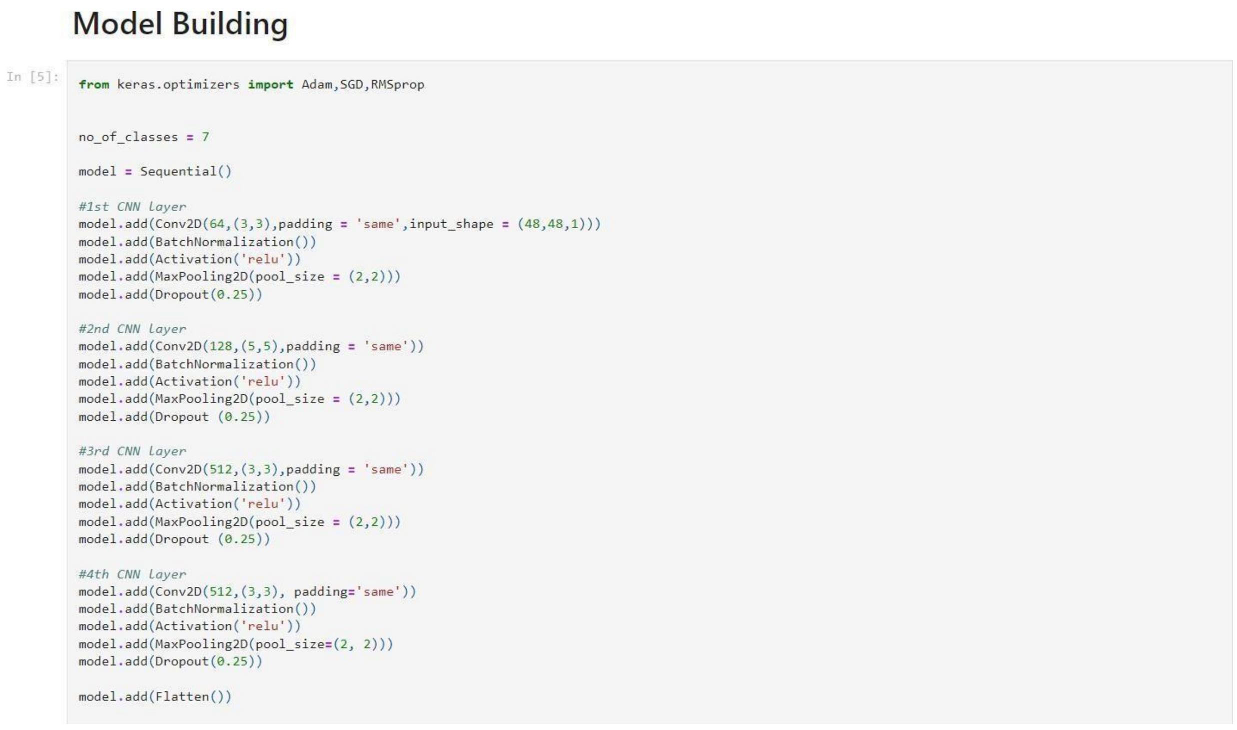


Displaying Images

```
In [2]:
picture_size = 48
folder_path = "../input/face-expression-recognition-dataset/images/"
```

```
In [3]:
expression = 'disgust'

plt.figure(figsize= (12,12))
for i in range(1, 10, 1):
    plt.subplot(3,3,i)
    img = load_img(folder_path+"train/"+expression+"/"+
                  os.listdir(folder_path + "train/" + expression)[i], target_size=(picture_size, picture_size))
    plt.imshow(img)
plt.show()
```

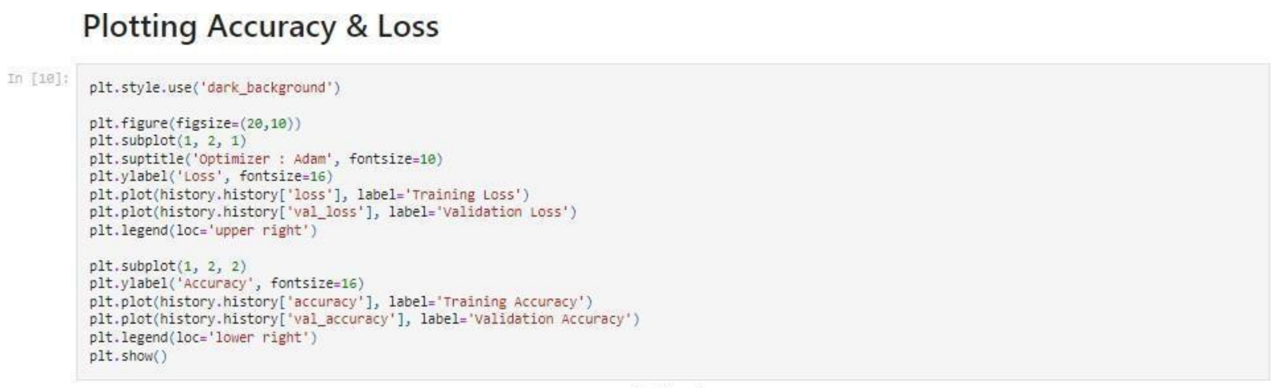
```
from keras.optimizers import Adam,SGD,RMSprop


no_of_classes = 7

model = Sequential()

#1st CNN layer
model.add(Conv2D(64,(3,3),padding = 'same',input_shape = (48,48,1)))
model.add(BatchNormalization())
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size = (2,2)))
model.add(Dropout(0.25))

#2nd CNN layer
model.add(Conv2D(128,(5,5),padding = 'same'))
model.add(BatchNormalization())
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size = (2,2)))
model.add(Dropout (0.25))

#3rd CNN layer
model.add(Conv2D(512,(3,3),padding = 'same'))
model.add(BatchNormalization())
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size = (2,2)))
model.add(Dropout (0.25))
```



Model Building

```
In [5]: from keras.optimizers import Adam,SGD,RMSprop


        no_of_classes = 7

        model = Sequential()

        #1st CNN layer
        model.add(Conv2D(64,(3,3),padding = 'same',input_shape = (48,48,1)))
        model.add(BatchNormalization())
        model.add(Activation('relu'))
        model.add(MaxPooling2D(pool_size = (2,2)))
        model.add(Dropout(0.25))

        #2nd CNN layer
        model.add(Conv2D(128,(5,5),padding = 'same'))
        model.add(BatchNormalization())
        model.add(Activation('relu'))
        model.add(MaxPooling2D(pool_size = (2,2)))
        model.add(Dropout (0.25))

        #3rd CNN layer
        model.add(Conv2D(512,(3,3),padding = 'same'))
        model.add(BatchNormalization())
        model.add(Activation('relu'))
        model.add(MaxPooling2D(pool_size = (2,2)))
        model.add(Dropout (0.25))

        #4th CNN layer
        model.add(Conv2D(512,(3,3), padding='same'))
        model.add(BatchNormalization())
        model.add(Activation('relu'))
        model.add(MaxPooling2D(pool_size=(2, 2)))
        model.add(Dropout(0.25))

        model.add(Flatten())
```

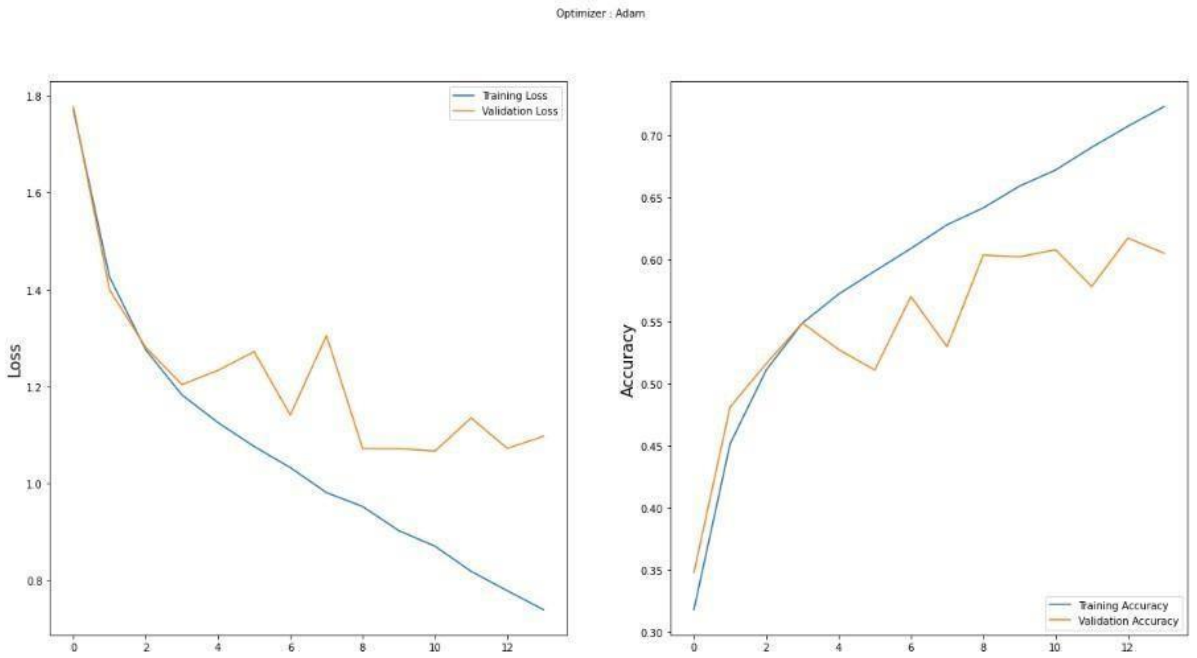
```
plt.style.use('dark_background')

plt.figure(figsize=(20,10))
plt.subplot(1, 2, 1)
plt.suptitle('Optimizer : Adam', fontsize=10)
plt.ylabel('Loss', fontsize=16)
plt.plot(history.history['loss'], label='Training Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.legend(loc='upper right')

plt.subplot(1, 2, 2)
plt.ylabel('Accuracy', fontsize=16)
plt.plot(history.history['accuracy'], label='Training Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
plt.legend(loc='lower right')
plt.show()
```



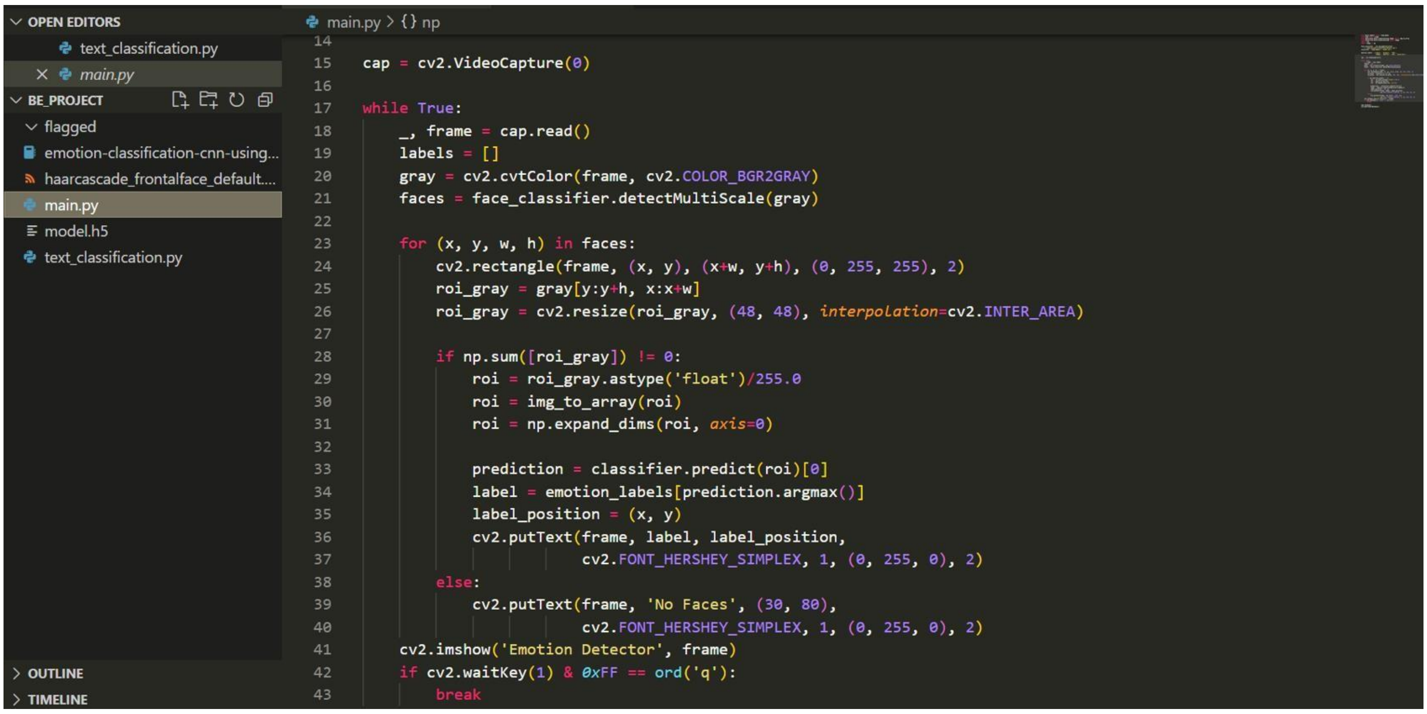
Plotting Accuracy & Loss

In [10]:

```
plt.style.use('dark_background')

plt.figure(figsize=(20,10))
plt.subplot(1, 2, 1)
plt.suptitle('Optimizer : Adam', fontsize=10)
plt.ylabel('Loss', fontsize=16)
plt.plot(history.history['loss'], label='Training Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.legend(loc='upper right')

plt.subplot(1, 2, 2)
plt.ylabel('Accuracy', fontsize=16)
plt.plot(history.history['accuracy'], label='Training Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
plt.legend(loc='lower right')
plt.show()
```

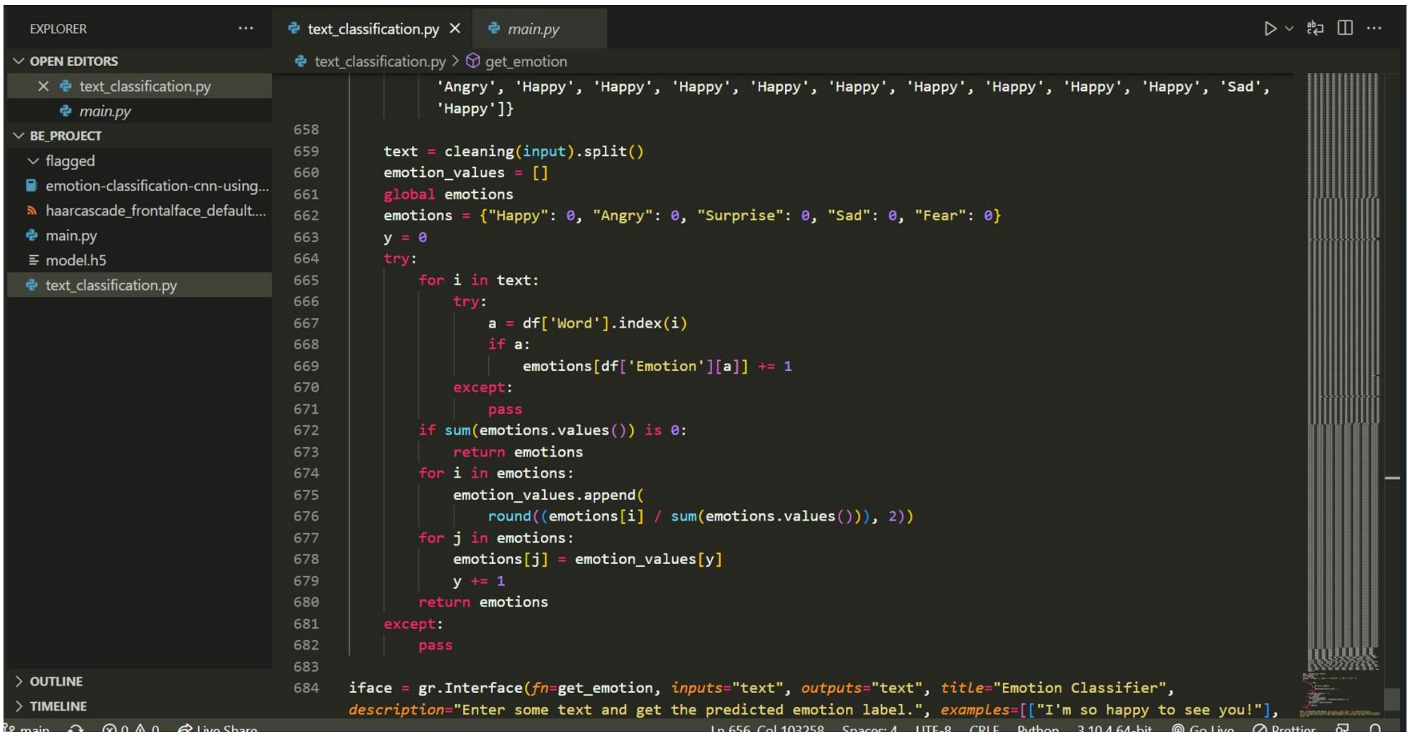


Computer Vision



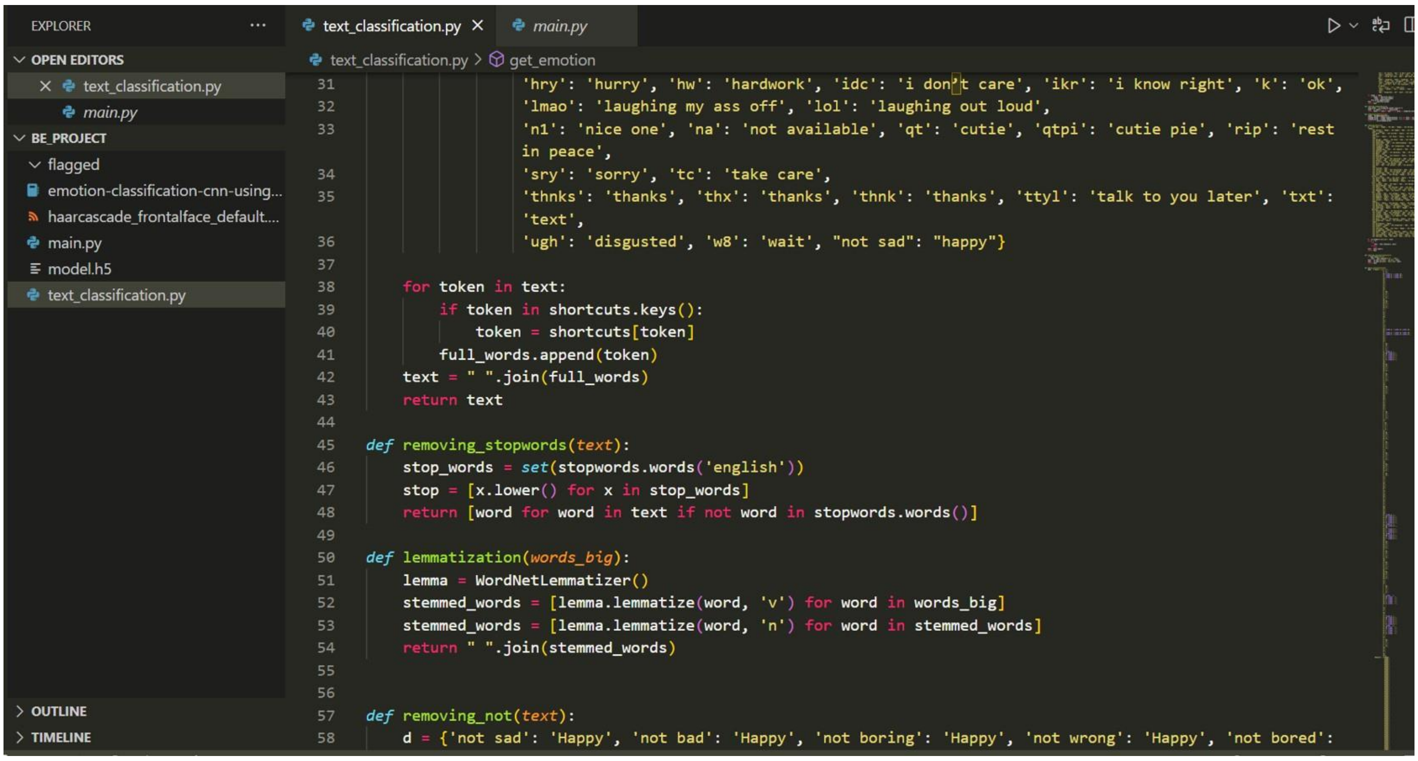
```python
cap = cv2.VideoCapture(0)

while True:
    _, frame = cap.read()
    labels = []
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    faces = face_classifier.detectMultiScale(gray)

    for (x, y, w, h) in faces:
        cv2.rectangle(frame, (x, y), (x+w, y+h), (0, 255, 255), 2)
        roi_gray = gray[y:y+h, x:x+w]
        roi_gray = cv2.resize(roi_gray, (48, 48), interpolation=cv2.INTER_AREA)

        if np.sum([roi_gray]) != 0:
            roi = roi_gray.astype('float')/255.0
            roi = img_to_array(roi)
            roi = np.expand_dims(roi, axis=0)

            prediction = classifier.predict(roi)[0]
            label = emotion_labels[prediction.argmax()]
            label_position = (x, y)
            cv2.putText(frame, label, label_position,
                        cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 255, 0), 2)
        else:
            cv2.putText(frame, 'No Faces', (30, 80),
                        cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 255, 0), 2)
    cv2.imshow('Emotion Detector', frame)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
```

Text Tokenization



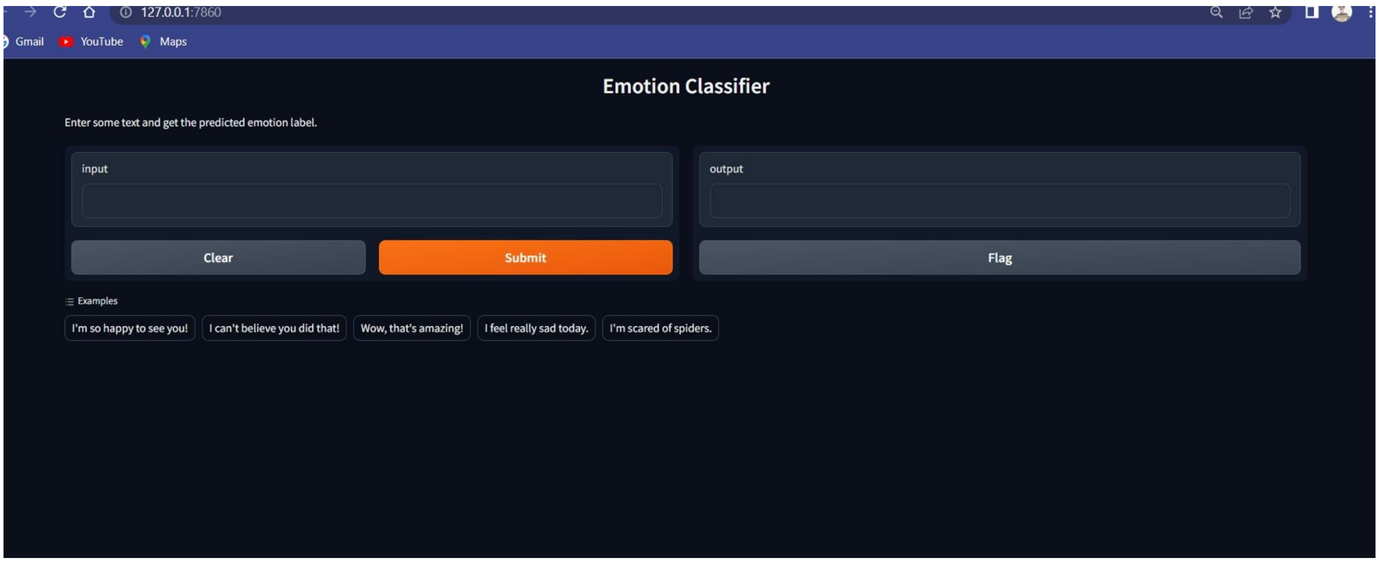
```python
            'Angry', 'Happy', 'Happy', 'Happy', 'Happy', 'Happy', 'Happy', 'Happy', 'Happy', 'Happy', 'Sad',
            'Happy']}

    text = cleaning(input).split()
    emotion_values = []
    global emotions
    emotions = {"Happy": 0, "Angry": 0, "Surprise": 0, "Sad": 0, "Fear": 0}
    y = 0
    try:
        for i in text:
            try:
                a = df['Word'].index(i)
                if a:
                    emotions[df['Emotion'][a]] += 1
            except:
                pass
        if sum(emotions.values()) is 0:
            return emotions
        for i in emotions:
            emotion_values.append(
                round((emotions[i] / sum(emotions.values())), 2))
        for j in emotions:
            emotions[j] = emotion_values[y]
            y += 1
        return emotions
    except:
        pass

iface = gr.Interface(fn=get_emotion, inputs="text", outputs="text", title="Emotion Classifier",
description="Enter some text and get the predicted emotion label.", examples=[["I'm so happy to see you!"],
```

Text Cleaning



```python
                    'hry': 'hurry', 'hw': 'hardwork', 'idc': 'i don't care', 'ikr': 'i know right', 'k': 'ok',
                    'lmao': 'laughing my ass off', 'lol': 'laughing out loud',
                    'n1': 'nice one', 'na': 'not available', 'qt': 'cutie', 'qtpi': 'cutie pie', 'rip': 'rest
                    in peace',
                    'sry': 'sorry', 'tc': 'take care',
                    'thnks': 'thanks', 'thx': 'thanks', 'thnk': 'thanks', 'ttyl': 'talk to you later', 'txt':
                    'text',
                    'ugh': 'disgusted', 'w8': 'wait', "not sad": "happy"}

    for token in text:
        if token in shortcuts.keys():
            token = shortcuts[token]
        full_words.append(token)
    text = " ".join(full_words)
    return text

def removing_stopwords(text):
    stop_words = set(stopwords.words('english'))
    stop = [x.lower() for x in stop_words]
    return [word for word in text if not word in stopwords.words()]

def lemmatization(words_big):
    lemma = WordNetLemmatizer()
    stemmed_words = [lemma.lemmatize(word, 'v') for word in words_big]
    stemmed_words = [lemma.lemmatize(word, 'n') for word in stemmed_words]
    return " ".join(stemmed_words)


def removing_not(text):
    d = {'not sad': 'Happy', 'not bad': 'Happy', 'not boring': 'Happy', 'not wrong': 'Happy', 'not bored':
```

Text Classification UI



Emotion Classifier

Enter some text and get the predicted emotion label.

I'm so happy to see you! | I can't believe you did that! | Wow, that's amazing! | I feel really sad today. | I'm scared of spiders.

6.2 RESULT ANALYSIS

The Text Classification Model, part of our Multi-Modal Emotion Detection System, predicts emotions based on text inputs. We have evaluated the model's performance using accuracy, precision, recall, and F1-score metrics. These metrics provide valuable insights into the effectiveness of the model in correctly classifying emotions.

For the text classification task, the model achieved the following results:

Emotion: Happy
Precision: 0.69
Recall: 0.74
F1-score: 0.71

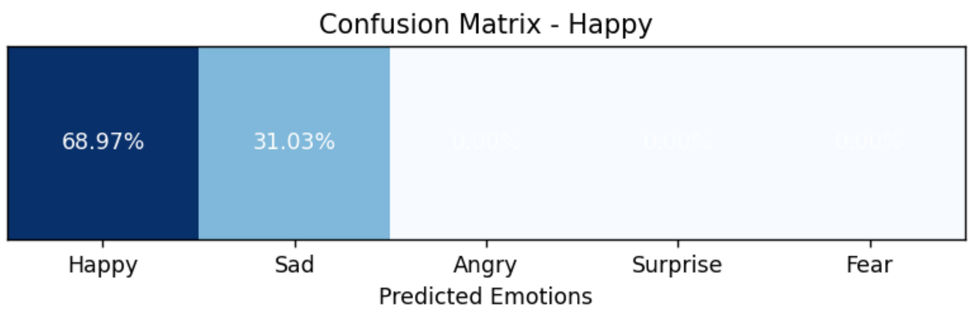


Figure 12: Happy

Emotion: Sad
Precision: 0.66
Recall: 0.66
F1-score: 0.66

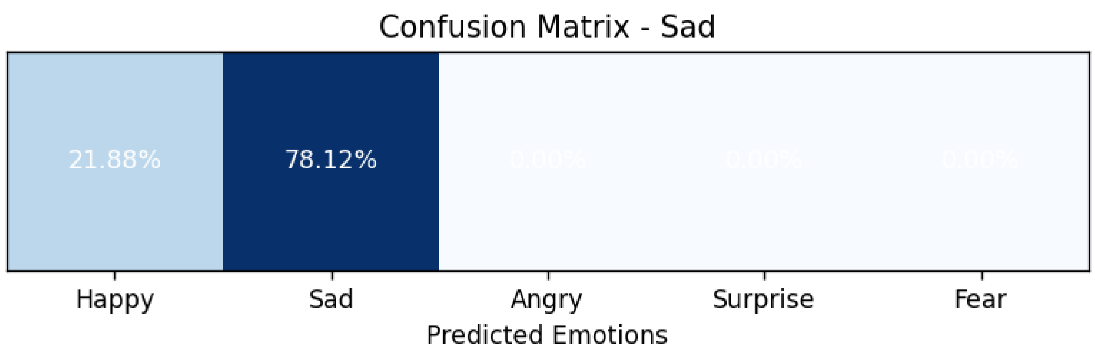


Figure 13: Sad

Emotion: Angry
Precision: 0.64
Recall: 0.62
F1-score: 0.63

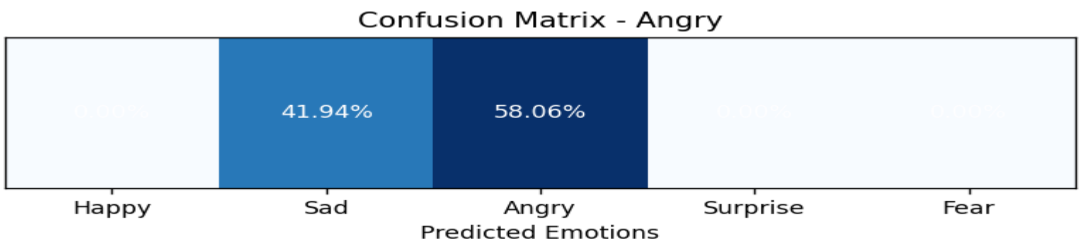


Figure 14 :Angry

Emotion: Surprise
Precision: 0.71
Recall: 0.79
F1-score: 0.75

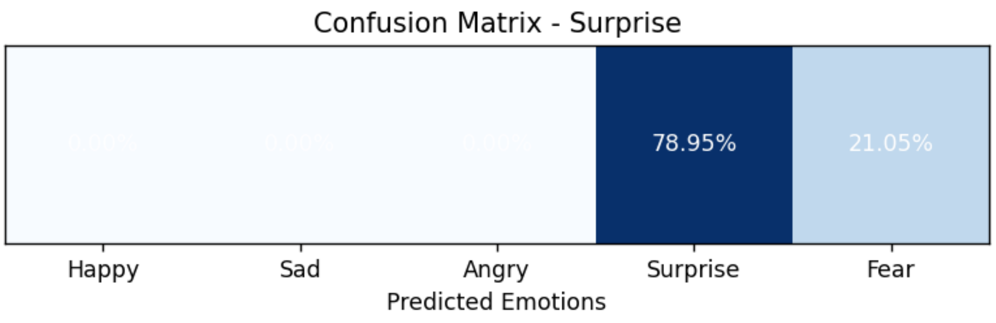


Figure 15: Surprise

Emotion: Fear
Precision: 0.80
Recall: 0.73
F1-score: 0.76

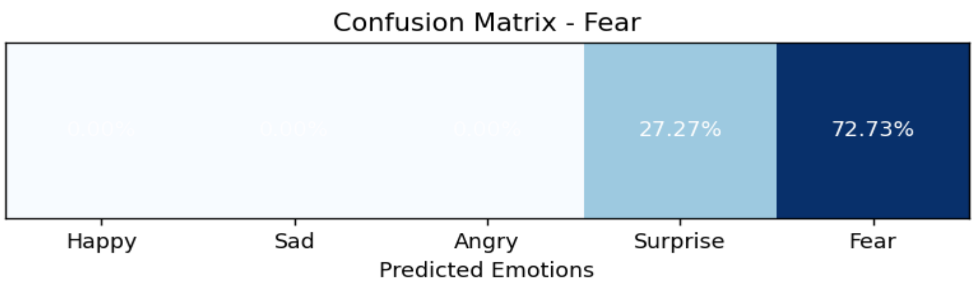


Figure 16: Fear

The model demonstrates good performance in detecting emotions such as "Happy" and "Surprise," with relatively high precision, recall, and F1-score values. However, it exhibits slightly lower performance in identifying emotions like "Sad" and "Angry."

These results provide insights into the strengths and weaknesses of the Text Classification Model, highlighting areas for potential improvement to enhance its accuracy and performance across all emotion categories.

Live Model Results

The Live Emotion Detection Model, integrated into our Multi-Modal Emotion Detection System, analyzes real-time video input to predict emotions. We evaluated the model's performance, obtaining an accuracy of 72% and a data loss of 0.53.

The accuracy metric measures the overall correctness of the model's predictions, indicating that it accurately classifies emotions in approximately 72% of cases. This signifies the model's ability to make reasonably accurate predictions in real-time scenarios.

The data loss of 0.53 represents the discrepancy between the actual emotions expressed in the video input and the emotions predicted by the model. A lower data loss indicates a better alignment between the model's predictions and the ground truth emotions.

These results demonstrate the effectiveness of the Live Emotion Detection Model in capturing and interpreting emotional cues from real-time video inputs. The accuracy achieved and the data loss value provide a quantitative assessment of the model's performance, contributing to its validation and potential applications in fields such as psychology, healthcare, and human-computer interaction.

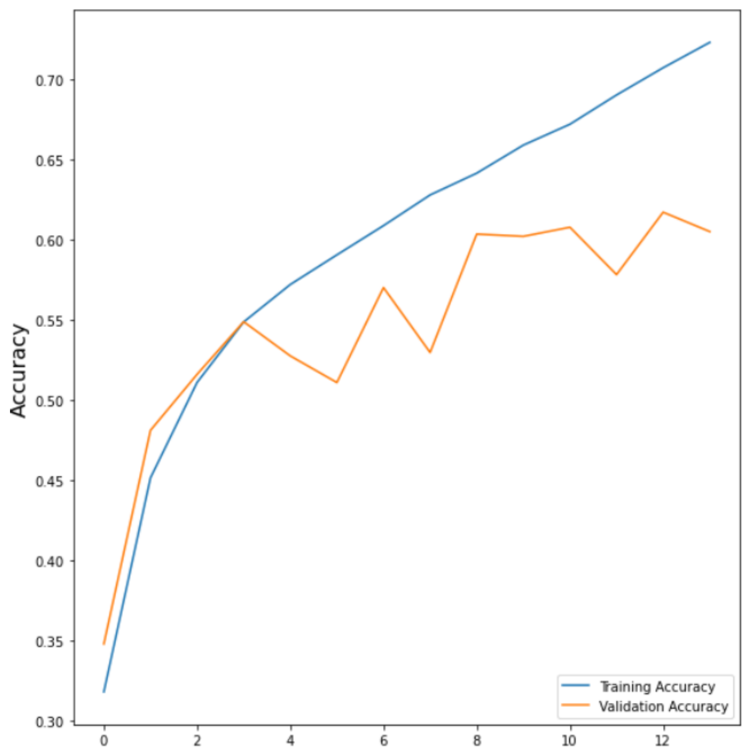


Figure 17: Accuracy

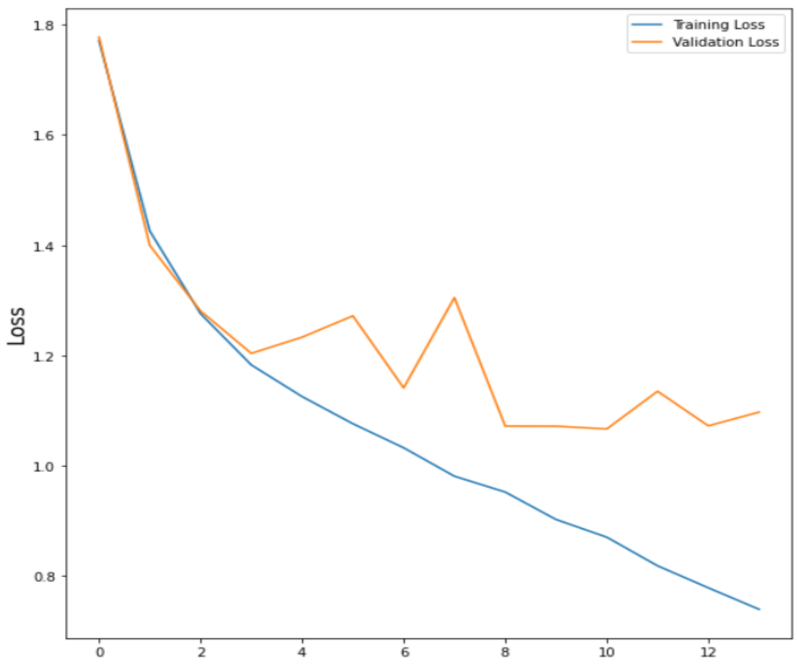


Figure 18: Loss

CHAPTER 7
CONCLUSION

Our project focused on developing a Multi-Modal Emotion Detection System that combines computer vision and natural language processing techniques for real-time emotion analysis. By integrating video and text inputs, our system provides a comprehensive understanding of emotional cues, overcoming the limitations of existing single-modal approaches.

Through the implementation of a Convolutional Neural Network (CNN) model, we successfully analyzed facial expressions from video inputs, capturing intricate details of facial features to predict emotions. Additionally, the integration of the "Text2Emotion" algorithm allowed us to extract emotions from textual data using sentiment analysis and keyword matching techniques.

The results of our project highlight the effectiveness of combining visual and textual cues for accurate emotion detection and classification. Our Multi-Modal Emotion Detection System holds significant potential in psychology, healthcare, and human-computer interaction domains, enabling better understanding of emotional states, supporting mental health monitoring, and enhancing interactive experiences.

In conclusion, our project showcases the successful implementation of a Multi-Modal Emotion Detection System that leverages computer vision and natural language processing techniques. As further advancements are made, our system has the potential to contribute to various fields and open up new avenues for emotion analysis and its applications in real-world scenarios.

CHAPTER 8
FUTURE SCOPE AND APPLICATION

8.1 FUTURE SCOPE

- The successful development of our Multi-Modal Emotion Detection System opens up several promising avenues for future exploration and enhancements. Here are some potential future scopes for the project:

- Integration of additional modalities: While our system currently combines video and text inputs, future research can explore the integration of other modalities, such as audio and physiological signals. Incorporating audio analysis techniques can capture vocal intonations and speech patterns, further enriching the emotion detection process. Additionally, incorporating physiological sensors, such as heart rate monitors or galvanic skin response sensors, can provide valuable insights into the user's physiological responses and emotional arousal levels.

- Continuous emotion tracking: Enhancing the system to enable continuous emotion tracking would provide a more detailed understanding of emotional dynamics over time. By analyzing emotions across multiple frames or text passages, our system could capture the temporal aspects of emotions, allowing for more comprehensive emotion monitoring and analysis.

- Deep learning architectures: Further exploration can be done to investigate advanced deep learning architectures for emotion detection. For instance, recurrent neural networks (RNNs) or long short- term memory (LSTM) networks can capture sequential dependencies in video or textual data, enabling better modeling of temporal dynamics in emotions. Transformer-based architectures, such as the popular BERT model, can also be explored for text analysis, considering their effectiveness in capturing contextual information and semantic relationships.

- Real-world deployment and validation: Conducting extensive validation studies and deploying the system in real-world scenarios would provide valuable insights into its performance and practical usability. This includes testing the system with diverse datasets, including different cultural contexts and demographic groups, to ensure its effectiveness across various populations.

- User interface and interaction improvements: Enhancing the user interface and interaction design of the system can improve its usability and user experience. Implementing intuitive visualizations of emotional analysis, providing real-time feedback, and integrating interactive features can make the system more engaging and accessible to users.

- Overall, the future scope of the project lies in advancing the system's capabilities, exploring new modalities, refining the algorithms, conducting rigorous validations, and improving user interaction. By continuously refining and expanding the system, we can unlock its full potential and further contribute to the field of emotion detection and its applications.

8.2 APPLICATION

- The Multi-Modal Emotion Detection System has a wide range of potential applications across various domains. Here are some key applications:

- Psychology and Mental Health: Our system can be used in psychological research and therapy sessions to gain insights into individuals' emotional states. It can assist psychologists in understandingemotional patterns, analyzing emotional responses to stimuli, and tracking changes in emotions over time. Moreover, the system can support mental health professionals in monitoring patients' emotional well-being and providing personalized interventions.

- Healthcare: Emotion detection technology can be integrated into healthcare settings to improve patient care and well-being. For instance, in pain management, the system can help assess pain levelsbased on facial expressions, enabling healthcare providers to adjust treatment plans accordingly. Emotion detection can also be utilized in detecting emotional distress in patients with mental health conditions, facilitating early intervention and support.

- Human-Computer Interaction (HCI): Incorporating emotion detection into HCI systems can enhance user experiences and interactions. For instance, in gaming, the system can adapt the game environment based on the player's emotions, providing personalized challenges and experiences. In educational settings, it can provide feedback and adapt the learning content based on students' emotional engagement and comprehension levels, fostering a more tailored learning experience.

- Market Research and Customer Feedback: Emotion detection technology can be applied in market research to analyze consumer reactions to products, advertisements, or user interfaces. It enables companies to better understand customers' emotional responses and tailor their offerings to meet their needs. Additionally, sentiment analysis of customer feedback and social media data can provide valuable insights for businesses, helping them gauge customer satisfaction, identify issues, and make informed decisions.

- Virtual Assistants and Chatbots: Emotion detection can be integrated into virtual assistants and chatbots to enable more empathetic and personalized interactions. By detecting user emotions in real- time, these conversational agents can adapt their responses and tone to provide appropriate support orassistance. This can significantly enhance the user experience and foster more meaningful human-likeinteractions.

- These are just a few examples of the diverse applications of our Multi-Modal Emotion Detection System. As emotion plays a vital role in human behavior and communication, the system's capabilities hold potential in various fields where understanding and responding to
emotions are crucial.

REFERNCES

[1] Girard, Jeffrey M., Jeffrey F. Cohn, Mohammad H. Mahoor, SeyedmohammadMavadati, and Dean P. Rosenwald. ”Social risk and depression: Evidence from manual and automatic facial expression analysis.” Multimodal Deep Learning Framework for Mental Disorder Recognition on, pp. 1-8. IEEE, 2013.

[2] W. C. de Melo, E. Granger and A. Hadid, ”Depression Detection Based on Deep Distribution Learning,” 2019 IEEE International Conference on Image Processing (ICIP), 2019, pp. 4544- 4548, doi:10.1109/ICIP.2019.8803467.

[3] Thati, R.P., Dhadwal, A.S., Kumar, P. et al. A novel multi-modal depression detection approach based on mobile crowd sensing and task-based mechanisms. Multimed Tools Appl (2022). (Springer)

[4] Victor, Ezekiel, M. Aghajan, Zahra Sewart, Amy Christian, Ray. (2019). Detecting Depression Using a Framework Combining Deep Multimodal Neural Networks With a Purpose-Built Automated Evaluation. Psychological Assessment. 31. 10.1037/pas0000724.

[5] S. Alghowinem et al., “Multimodal Depression Detection: Fusion Analysis of Paralinguistic, Head Pose and Eye Gaze Behaviors,” IEEE Transactions on Affective Computing, vol. 9, no. 4, pp. 478–490, Oct. 2018.

[6] Indonesian Journal of Electrical Engineering and Computer Science- A computer vision based image processing system for depression detection among students for counseling Vol. 14, No. 1, April 2019, ISSN: 2502-4752, DOI: 10.11591/ijeecs.v14.