Project Report on

EMOTION RECOGNITION USING DIFFERENT DEEP LEARNING TECHNIQUES

at

U.V Patel College of Engineering

Internal Guide:
Prof. Ketan Sarvakar

Prepared By:
Mr. Abhishek V Jani (19012011092)
Mr. Karrtik Baheti (19012531005)

B. Tech Semester VIII
(Computer Engineering)
May 2023

Submitted to,
Department of Computer Engineering
U.V. Patel College of Engineering
Ganpat University, Kherva - 384 012

U.V. PATEL COLLEGE OF ENGINEERING

Ganpat University
॥ विद्यया समाजोत्कर्षः ॥
U.V. Patel College of Engineering
CELEBRATING 25 YEARS OF EXCELLENCE 1997 - 2022
25 Years excellence in innovative technical education in shaping engineers

13/05/23

C E R T I F I C A T E

TO WHOM SO EVER IT MAY CONCERN

This is to certify that Mr. **Abhishek V Jani** student of **B.Tech. Semester VIII (Computer Engineering)** has completed his full semester on site project work titled "**Emotion Recognition using different Deep Learning techniques**" satisfactorily in partial fulfillment of the requirement of Bachelor of Technology degree of Computer Engineering of Ganpat University, Kherva, Mehsana in the year 2022-2023.

Prof. Ketan Sarvakar	**Dr. Paresh M. Solanki**
College Project Guide	**HOD, Computer Engineering**

U.V. PATEL COLLEGE OF ENGINEERING

Ganpat University
॥ विद्यया समाजोत्कर्षः ॥
U.V. Patel College of Engineering
CELEBRATING 25 YEARS OF EXCELLENCE 1997 - 2022
25 Years excellence in innovative technical education in shaping engineers

13/05/23

CERTIFICATE

TO WHOM SO EVER IT MAY CONCERN

This is to certify that Mr. **Karrtik Baheti** student of **B.Tech. Semester VIII (Computer Engineering A.I)** has completed his full semester on site project work titled "**Emotion Recognition using different Deep Learning techniques**" satisfactorily in partial fulfillment of the requirement of Bachelor of Technology degree of Computer Engineering of Ganpat University, Kherva, Mehsana in the year 2022-2023.

Prof. Ketan Sarvakar	**Dr. Paresh M. Solanki**
College Project Guide	**HOD, Computer Engineering**

Capgemini Technology Services India Limited
(Formerly known as IGATE Global Solutions Limited)
IT 1, IT 2, Airol MIDC, Thane - Belapur Road,
Navi Mumbai 400708, Maharashtra, India.
Tel: +91 22 7144 4283 | Fax: +91 22 7141 2121
www.capgemini.com/in-en

Superset ID: 3540664

Letter of Intent ("LOI")

December 17, 2022

Dear Abhishek V Jani,

We are pleased to inform that your candidature has been shortlisted for the position of **Analyst/A4** with **Capgemini Technology Services India Limited** (hereinafter referred to as "Capgemini" or Company).
You will be required to participate and complete the pre-onboarding training program assigned and applicable to you as may be communicated by the Company later.
Please note that it is essential for you to participate, effectively leverage and successfully complete this program as a prerequisite prior to being onboarded as an employee with Capgemini.

We request you to carefully read and understand the Terms and Conditions of this Letter of Intent with Annexures hereto (hereinafter referred to as LOI).

A Please note that your name mentioned in this LOI will be used to create your records in Capgemini & the same will be continued for all the communication & Company documentation purpose. In case you need a change in the name; please notify before commencement of training. Please note that no changes to the record can be made later in time. The name provided by you should match with the identification documents submitted to the Company, such as Aadhar Card, PAN card, Passport, etc.

B We are proposing compensation package and benefits post-onboarding, the details of which are set forth in **Annexure 1** to this LOI.

C Upon accepting this LOI, you will be required to submit a set of documents as mentioned in the **Annexure- 2** . Thereafter, you will be provided access to our pre-onboarding training program, as applicable. This will enable you to learn and master the concepts and skills required to be industry ready. The pre-onboarding training program can include physical classroom training/ self-paced e-learning/ hybrid model of training. The learning journey will be inclusive of assignments, assessments, hackathons/ competitions, and webinars as deemed appropriate by Capgemini.

D The progress made by you in this learning journey would not only help you in getting onboarded but also help you to be trained for advanced skills relevant to your career at Capgemini. We also encourage you to learn beyond the prescribed course curriculum and acquire industry recognized certifications to accelerate your career in this competitive industry.

E Pre-onboarding training Program and Terms & Conditions of the LOI

1. Pre-onboarding Document Verification: Capgemini adheres to a strong document verification process. As a part of this process all the personal, educational and professional (if

Capgemini Technology Services India Limited
(Formerly known as IGATE Global Solutions Limited)
IT 1, IT 2, Airol MIDC, Thane - Belapur Road,
Navi Mumbai 400708, Maharashtra, India.
Tel: +91 22 7144 4283 | Fax: +91 22 7141 2121
www.capgemini.com/in-en

Superset ID: 3485590

Letter of Intent ("LOI")

January 02, 2023

Dear Karrtik Baheti,

We are pleased to inform that your candidature has been shortlisted for the position of **Analyst/A4** with **Capgemini Technology Services India Limited** (hereinafter referred to as "Capgemini" or Company).
You will be required to participate and complete the pre-onboarding training program assigned and applicable to you as may be communicated by the Company later.
Please note that it is essential for you to participate, effectively leverage and successfully complete this program as a prerequisite prior to being onboarded as an employee with Capgemini.

We request you to carefully read and understand the Terms and Conditions of this Letter of Intent with Annexures hereto (hereinafter referred to as LOI).

A Please note that your name mentioned in this LOI will be used to create your records in Capgemini & the same will be continued for all the communication & Company documentation purpose. In case you need a change in the name; please notify before commencement of training. Please note that no changes to the record can be made later in time. The name provided by you should match with the identification documents submitted to the Company, such as Aadhar Card, PAN card, Passport, etc.

B We are proposing compensation package and benefits post-onboarding, the details of which are set forth in **Annexure 1** to this LOI.

C Upon accepting this LOI, you will be required to submit a set of documents as mentioned in the **Annexure- 2** . Thereafter, you will be provided access to our pre-onboarding training program, as applicable. This will enable you to learn and master the concepts and skills required to be industry ready. The pre-onboarding training program can include physical classroom training/ self-paced e-learning/ hybrid model of training. The learning journey will be inclusive of assignments, assessments, hackathons/ competitions, and webinars as deemed appropriate by Capgemini.

D The progress made by you in this learning journey would not only help you in getting onboarded but also help you to be trained for advanced skills relevant to your career at Capgemini. We also encourage you to learn beyond the prescribed course curriculum and acquire industry recognized certifications to accelerate your career in this competitive industry.

E Pre-onboarding training Program and Terms & Conditions of the LOI

1. Pre-onboarding Document Verification: Capgemini adheres to a strong document verification process. As a part of this process all the personal, educational and professional (if

Ganpat University | U.V. Patel College of Engineering
॥ विद्यया समाजोत्कर्षः ॥

Date:09/05/2023

Certificate of Completion of Final Semester Project

This is to certify that Mr. Abhishek V Jani(Enroll No.19012011092), student of B. Tech (Computer Engineering) from Ganpat University - U V Patel College of Engineering has successfully completed the final semester project titled Emotion Recognition using different Deep Learning techniques from 2'nd January 2023 to 2'nd May 2023. During the period of his internship program, we found him punctual, hardworking and inquisitive.

We wish him all the best in his future endeavours.

Chirag

Mr. Chirag Gami.
Assistant Professor at U.V.P.C.E,
Computer Engineering Department.
Contact No : +91 9824335549
Email : ccg01@ganpatuniversity.ac.in

U.V.PATEL COLLEGE OF ENGINEERING
GANPAT VIDYANAGAR
ESTD-1997

Ganpat University | U.V. Patel College of Engineering
॥ विद्यया समाजोत्कर्षः ॥

Date:09/05/2023

Certificate of Completion of Final Semester Project

This is to certify that Mr. Karrtik Baheti(Enroll No.19012531005), student of B. Tech (Computer Engineering-AI) from Ganpat University - U V Patel College of Engineering has successfully completed the final semester project titled Emotion Recognition using different Deep Learning techniques from 2'nd January 2023 to 2'nd May 2023. During the period of his internship program, we found him punctual, hardworking and inquisitive.

We wish him all the best in his future endeavours.

Mr. Chirag Gami.
Assistant Professor at U.V.P.C.E,
Computer Engineering Department.
Contact No : +91 9824335549
Email : ccg01@ganpatuniversity.ac.in

U.V.PATEL COLLEGE OF ENGINEERING
GANPAT VIDYANAGAR
ESTD-1997

ACKNOWLEDGEMENT

The successful completion of any task is a testament to the ceaseless cooperation and unwavering support of individuals whose guidance and encouragement elevate every effort towards triumph.

We extend our sincere thanks to **Prof. Ketan Sarvakar** for his dedicated guidance and expertise throughout the project. His unwavering support, timely feedback, and valuable suggestions have played a pivotal role in shaping the direction and ensuring the quality of this project.

We are also grateful to **Dr. Paresh Solanki**, Head of the Department of Computer Engineering at U V Patel College of Engineering, Mehsana, for his continual words of encouragement and motivation throughout this Major Project.

Additionally, we would like to express our sincere appreciation to **Mr. Ronak Kosti** for providing us with access to the dataset crucial for our research. His willingness to share the dataset and his assistance in understanding its intricacies have significantly contributed to the success of our project.

Furthermore, we would like to acknowledge the efforts of the entire faculty members of the Department of Computer Engineering for their contributions to our education and for fostering a collaborative learning environment.

The successful completion of this project would not have been possible without the unwavering support and guidance of these individuals, and we express our sincerest gratitude for their invaluable contributions.

ABSTRACT

Emotion detection is crucial for understanding human behavior and improving human-computer interaction. This project analyzes emotion detection using different pretrained models with the EMOTIC dataset, a benchmark dataset containing images depicting diverse emotional states.

The objective of this project is to compare and evaluate the performance of two pretrained models in accurately detecting emotions from images. VGG19 and AlexNet are the pretrained models utilized in this analysis. These models have been pre-trained on large-scale image datasets and exhibit strong feature extraction capabilities.

The results provide insights into the performance of pretrained models for emotion detection on the EMOTIC dataset, contributing to the field of emotion recognition and guiding the development of more accurate and efficient emotion detection systems.

This project presents a comprehensive analysis of emotion detection using various pretrained models, highlighting their capabilities and limitations. The findings advance emotion recognition technology and its applications in affective computing, human-computer interaction, and social robotics.

INDEX

Figures

Tables

1 INTRODUCTION

Emotion recognition is an essential component of human-machine interaction systems, which is widely used in various domains such as healthcare, entertainment, and security. Emotion recognition from facial expressions is a challenging task due to the complexity and variability of human facial expressions. Deep learning-based approaches have shown promising results in emotion recognition.

1.1 Purpose

The purpose of this project is to evaluate the performance of two pretrained models for emotion detection using the EMOTIC dataset. Emotion detection is crucial for understanding human behavior and enhancing human-computer interaction. By comparing pretrained models such as VGG19 and AlexNet, the project aims to determine their effectiveness in accurately classifying emotions depicted in images.

1.2 Scope of the Project

The scope of the project is to analyze pre-trained models on EMOTIC dataset and to predict emotions. The project involves the analysis of models such as Visual Geometry Group 19 (VGG19), AlexNet and their Transfer Learning Models, and their performance on the EMOTIC dataset.

The project aims to evaluate the performance of the models on the dataset, compare their results, and determine which model is best suited for emotion recognition tasks. The project also aims to identify the strengths and weaknesses of each model and provide insights for future research.

1.3 Objective

The objective of this project is to process the EMOTIC dataset and analyze pre-trained models on the Emotic dataset and to implement a model that can predict the emotion of a person based on the input image.

1.4 Emotion Detection

1.4.1 Overview

Emotion detection, also known as affective computing, is a field of study focused on recognizing and interpreting human emotions from various sources such as facial expressions, speech patterns, physiological signals, and textual data. It plays a crucial role in understanding human behavior, enhancing human-computer interaction, and enabling applications in diverse domains.

Emotion detection utilizes machine learning and artificial intelligence techniques to analyze and classify emotions based on input data. Facial expression analysis is a widely used method, leveraging computer vision algorithms to detect and interpret emotions from facial cues like eye movements, facial muscle contractions, and overall expression patterns. Speech analysis involves extracting emotional features from audio signals, including pitch, tone, and speech patterns. Textual analysis focuses on understanding emotions conveyed through written or spoken words.

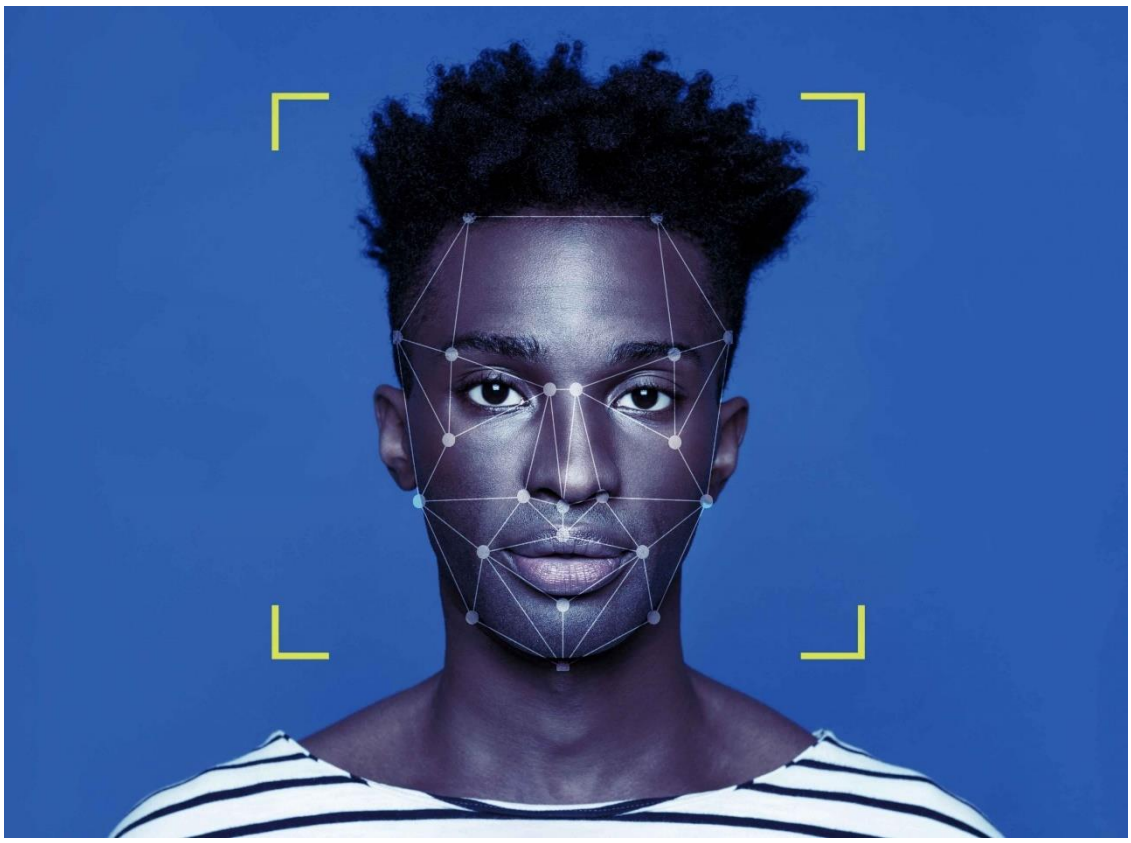

Figure 1.1 Emotion Detection

The advancement of deep learning algorithms and the availability of large-scale datasets have significantly improved emotion detection capabilities. Pretrained models, such as convolutional neural networks (CNNs) and Visual Geometry Groups (VGGs), have shown remarkable performance in recognizing emotions from different modalities.

Applications of emotion detection are diverse and range from affective computing in human-computer interaction to personalized marketing, mental health monitoring, and social robotics. In human-computer interaction, emotion detection enables systems to respond and adapt based on users' emotional states, enhancing user experience and engagement. In marketing, it helps gauge customer sentiment and tailor product offerings accordingly. In mental health monitoring, emotion detection aids in early detection of psychological disorders and provides personalized support. Social robotics leverages emotion detection to enable robots to perceive and respond to human emotions, facilitating more natural and meaningful interactions.

However, emotion detection still faces challenges, such as the subjective nature of emotions, cultural variations, and the need for diverse and balanced datasets. Ongoing research focuses on improving the accuracy and robustness of emotion detection algorithms and addressing these challenges to create more reliable and applicable emotion recognition systems.

Overall, emotion detection is a dynamic and interdisciplinary field that continues to evolve with advancements in technology and understanding of human emotions. It has the potential to revolutionize human-computer interaction and enable a wide range of applications that rely on understanding and responding to human emotions effectively.

1.4.2 Pre-Trained Models

1.4.2.1 AlexNet

- AlexNet is a deep convolutional neural network (CNN) architecture that made a significant impact on the field of computer vision.
- It was developed by Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton and won the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) in 2012.
- The architecture of AlexNet played a crucial role in advancing the field and paved the way for subsequent developments in deep learning.

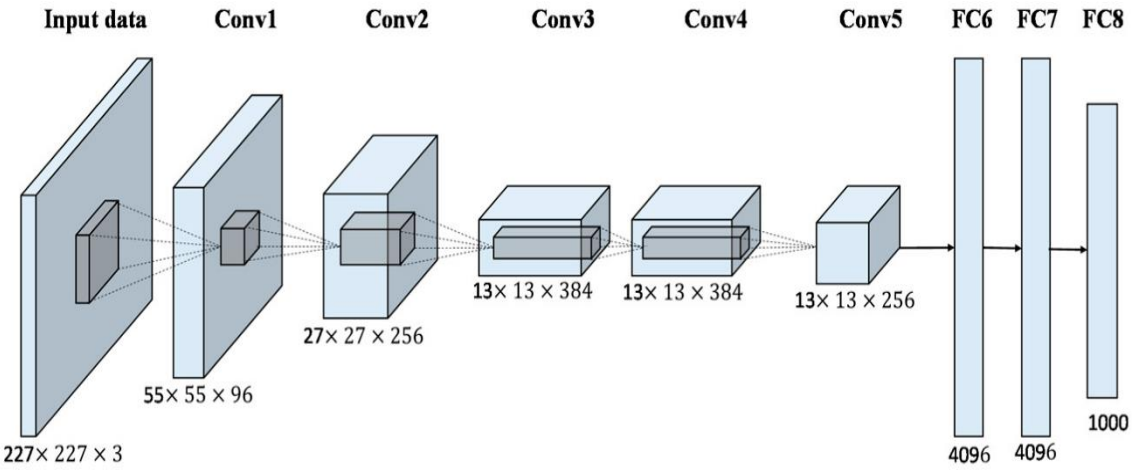

Figure 1.2 ALEXNET Architecture

- The detailed explanation of the AlexNet architecture:

    1. Convolutional Layers:

        - AlexNet starts with five convolutional layers.
        - The first layer performs convolutions on the input image. It consists of 96 filters with a size of 11x11 and a stride of 4 pixels.
        - This stride value helps in reducing the spatial dimensions of the image. The output is then passed through a rectified linear unit (ReLU) activation function, which introduces non-linearity to the model.
        - The subsequent convolutional layers are stacked on top of each other. The second layer has 256 filters of size 5x5 and a stride of 1, followed by a ReLU activation.
        - The third, fourth, and fifth convolutional layers have 384, 384, and 256 filters of size 3x3, respectively, with a stride of 1. ReLU activation is applied after each of these layers.

    2. Max Pooling:

        - After the first and second convolutional layers, max pooling is performed to reduce the spatial dimensions of the output.
        - A max pooling layer with a filter size of 3x3 and a stride of 2 is applied. Max pooling helps in capturing the most important features while reducing the computational complexity.

    3. Fully Connected Layers:

        - After the convolutional and pooling layers, the output is flattened and fed into fully connected layers.
        - The first fully connected layer consists of 4096 neurons, followed by a dropout layer that helps in regularizing the model and preventing overfitting.
        - The second fully connected layer also has 4096 neurons, followed by another dropout layer.
        - Finally, the output layer with the desired number of neurons (usually equal to the number of classes in a classification task) is added.

    4. Softmax Activation:

        - The output layer is activated using the softmax function, which normalizes the outputs into probabilities.

- In classification tasks, this helps in determining the class probabilities of the input image.

- AlexNet played a pivotal role in the resurgence of neural networks and deep learning.
- Its success demonstrated the effectiveness of deep CNNs in complex visual recognition tasks.
- Since then, researchers have developed more advanced CNN architectures, but AlexNet remains a landmark model that paved the way for the current advancements in computer vision and deep learning.

1.4.2.2 VGG19

- VGG19 is a deep convolutional neural network (CNN) architecture that was proposed by the Visual Geometry Group (VGG) at the University of Oxford.
- It is widely used in various computer vision tasks, including emotion detection, due to its simplicity and strong performance.

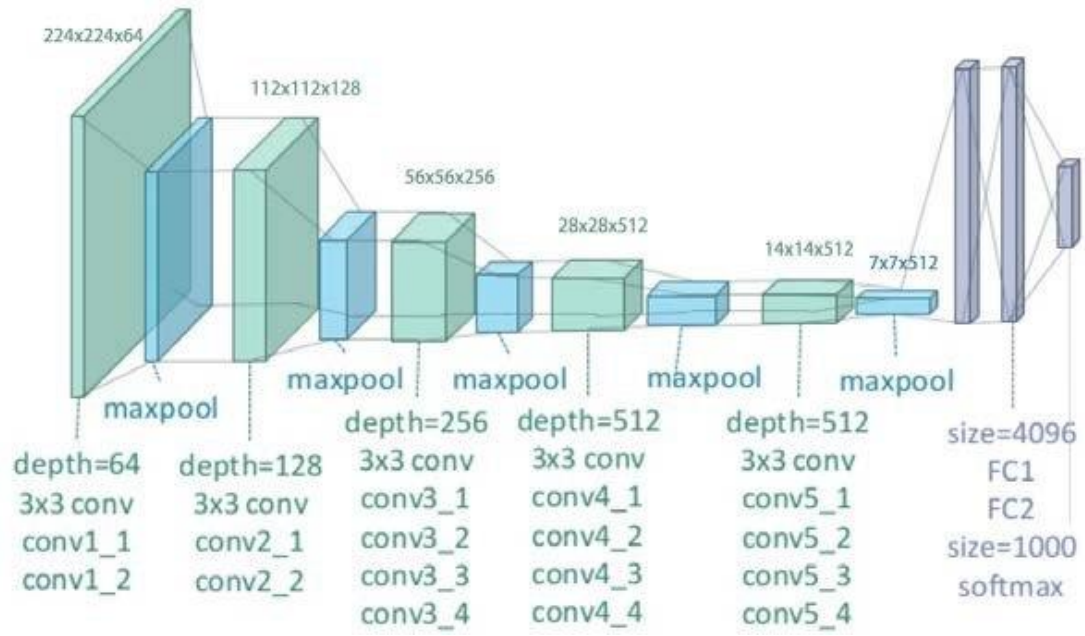

Figure 1.3 VGG19 Architecturre

- The architecture of VGG19 can be explained as follows:

    1. Input Layer: VGG19 takes an input image of size 224x224 pixels and processes it through the network.

    2. Convolutional Layers: VGG19 consists of 16 convolutional layers, each followed by a rectified linear unit (ReLU) activation function. These convolutional layers have a small receptive field of 3x3 pixels and use a stride of 1 for convolution. The use of multiple

convolutional layers helps capture different levels of abstraction in the input image.

3. Max Pooling Layers: After some of the convolutional layers, max pooling layers with a receptive field of 2x2 pixels and a stride of 2 are applied. Max pooling reduces the spatial dimensions of the feature maps while retaining the most important information.

4. Fully Connected Layers: After the convolutional and pooling layers, VGG19 has three fully connected layers. Each fully connected layer has 4,096 neurons. ReLU activation functions are applied to these layers as well. The fully connected layers integrate the learned features across the entire image for higher-level reasoning and classification.

5. Dropout Regularization: Dropout is applied after each fully connected layer to prevent overfitting. It randomly drops out a fraction of the neurons during training, forcing the network to learn more robust and generalizable features.

6. Output Layer: The output layer is a fully connected layer with the number of neurons corresponding to the number of emotion categories in the dataset. A softmax activation function is applied to produce the final classification probabilities for different emotions.

- The key characteristic of VGG19 is its deep architecture with a large number of layers, which allows it to learn intricate features and representations from the input image.
- However, this depth also leads to a higher number of parameters, making VGG19 more computationally expensive to train compared to some other architectures.
- VGG19 has shown impressive performance in various computer vision tasks, including emotion detection, when fine-tuned on emotion-specific datasets.
- Its ability to capture rich features and representations from images contributes to its effectiveness in accurately classifying emotions based on visual cues.

1.4.2.3 Transfer Learning

- Transfer learning is a technique in deep learning where a pre-trained model, which has been trained on a large-scale dataset, is used as a starting point for a new task or domain-specific problem.
- Instead of training a model from scratch, transfer learning leverages the learned knowledge and features from the pre-trained model to solve a related problem with limited labeled data.

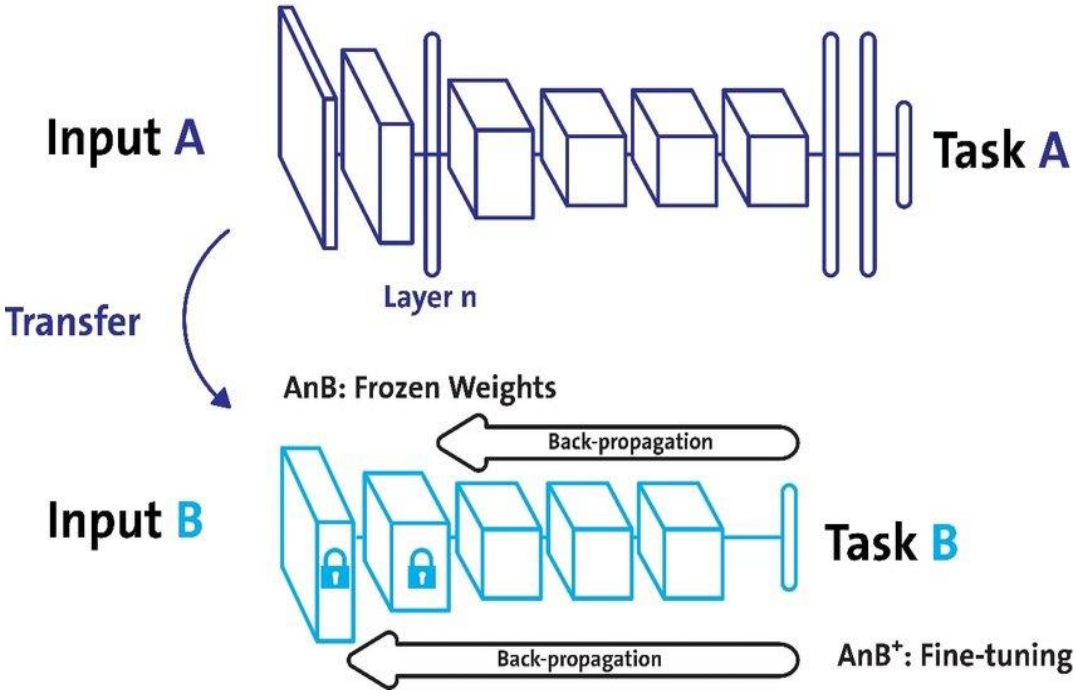

Figure 1.4 Transfer Learning

- In the context of emotion detection, transfer learning is beneficial for several reasons:

  1. Limited Labeled Data:

     - Emotion detection datasets are often relatively small compared to large-scale image datasets used for pre-training.
     - By using pre-trained models, which have learned general representations from extensive datasets, we can effectively leverage their knowledge to improve the performance on the emotion detection task even with limited labeled data.
     - It helps overcome the challenge of insufficient labeled data, as the pre-trained models have already learned generic features that are applicable across different tasks.

  2. Generalization and Feature Extraction:

     - Pre-trained models are trained on diverse and complex datasets, enabling them to learn rich and generalizable features.
     - These features capture high-level visual patterns and semantics, including shapes, textures, and contextual information.
     - By applying transfer learning, we can leverage these learned features to extract relevant information from emotion-related

images and improve the model's ability to recognize and classify different emotions accurately.

3. Time and Resource Efficiency:

    - Training deep neural networks from scratch can be computationally expensive and time-consuming, especially when dealing with large datasets.
    - By utilizing pre-trained models, we can significantly reduce the training time and computational resources required.
    - Transfer learning allows us to start with a well-initialized model, which has already learned low-level features, and fine-tune it on the emotion detection task.
    - This way, we can achieve good performance with less training time and computational cost.

- The Transfer learning is utilized in emotion detection to leverage the knowledge and features learned from pre-trained models on large-scale datasets.
- It helps overcome the limitations of limited labeled data, improves generalization, enhances feature extraction, and reduces training time and computational resources.
- Applying transfer learning with multiple pre-trained models allows for comparative analysis, ensemble learning, and capturing diverse representations, ultimately improving the accuracy and robustness of the emotion detection system.

2 FEASIBILITY STUDY

2.1 Study of the Current System

- Emotion detection has been studied extensively using various modalities, including images, text, vocal cues, videos, and more.
- Researchers have explored different approaches to recognize and classify emotions across multiple domains.
- Here is an overview of the existing approaches in emotion detection considering different modalities and the primary seven emotion classes:

1. Image-based Emotion Detection:

    - Image-based approaches analyze facial expressions to detect emotions.
    - They extract facial features, such as facial landmarks or local texture patterns, and utilize machine learning algorithms or deep learning models to classify emotions.
    - These approaches often focus on the primary seven emotion classes, including happiness, sadness, anger, fear, surprise, disgust, and neutral.

2. Text-based Emotion Detection:

    - Text-based approaches analyze textual data, such as social media posts, emails, or chat conversations, to detect emotions.
    - Natural Language Processing (NLP) techniques, including sentiment analysis and emotion classification algorithms, are used to extract emotions from the text.
    - Machine learning models, such as Support Vector Machines (SVM) or Recurrent Neural Networks (RNN), are commonly employed in text-based emotion detection.

3. Vocal-based Emotion Detection:

    - Vocal-based approaches analyze audio signals, such as speech or vocal intonations, to detect emotions.
    - Acoustic features, such as pitch, energy, or spectral properties, are extracted from the audio signals.
    - Machine learning or deep learning models, such as Gaussian Mixture Models (GMM) or Convolutional Neural Networks (CNN), are applied to classify emotions based on these features.

4. Multimodal Emotion Detection:

   - Multimodal approaches combine information from multiple modalities, such as images, text, and audio, to enhance emotion detection performance.
   - By integrating facial expressions, textual context, and vocal cues, these approaches capture a more comprehensive understanding of emotions.
   - Fusion techniques, including early fusion or late fusion, are used to combine features and make emotion predictions.

5. Video-based Emotion Detection:

   - Video-based approaches analyze temporal patterns in facial expressions and body movements to recognize emotions.
   - These approaches often involve tracking facial landmarks, capturing facial dynamics, and modeling the temporal evolution of emotions.
   - Machine learning models, such as Hidden Markov Models (HMM) or Recurrent Neural Networks (RNN), are employed to capture the temporal dynamics and classify emotions in videos.

6. Physiological-based Emotion Detection:

   - Physiological-based approaches analyze physiological signals, such as heart rate, skin conductance, or brain activity, to infer emotional states.
   - These approaches explore the relationship between physiological responses and emotions.
   - Techniques like Electrocardiography (ECG), Electrodermal Activity (EDA), or Electroencephalography (EEG) are utilized to collect and analyze physiological signals for emotion detection.

7. Deep Learning-based Emotion Detection:

   - Deep learning approaches have gained popularity in emotion detection tasks. Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), or Transformer models are used to automatically learn high-level representations from raw data, such as images, text, or audio.
   - Deep learning models excel in capturing intricate patterns and dependencies, leading to improved emotion detection performance.

- Existing approaches in emotion detection span multiple modalities, including images, text, vocal cues, videos, and physiological signals.
- Researchers have developed and applied various machine learning algorithms and deep learning models to recognize emotions across different domains.
- The primary seven emotion classes, namely happiness, sadness, anger, fear, surprise, disgust, and neutral, serve as the basis for emotion classification in these approaches.

- The existing commonly used datasets in existing approaches for emotion detection:

    1. CK+ (Cohn-Kanade) Database:

        - The CK+ database consists of facial images and videos capturing various facial expressions.
        - It provides detailed annotations of facial muscle movements and intensity for different emotions.
        - The dataset is widely used for studying facial expression analysis and emotion recognition.

    2. AffectNet:

        - AffectNet is a large-scale and diverse dataset containing over one million facial images annotated with seven basic emotions.
        - It covers a wide range of demographics, expressions, and variations in lighting conditions, making it suitable for training deep learning models.

    3. FER2013 (Facial Expression Recognition 2013):

        - The FER2013 dataset is a popular dataset for facial expression recognition.
        - It includes a large number of grayscale facial images labeled with seven emotions.
        - The dataset is widely used to evaluate the performance of emotion recognition algorithms.

    4. RAF-DB (Ryerson Audio-Visual Database of Emotional Speech and Song):

        - The RAF-DB dataset comprises audio-visual recordings of individuals expressing emotions through speech and song.
        - It provides a diverse range of emotional expressions in both visual and audio modalities, enabling researchers to explore multimodal emotion recognition.

    5. SEMAINE (Sustained Emotionally colored Machine-human Interaction using Nonverbal Expression):

        - The SEMAINE dataset offers multimodal recordings of human-human interactions involving verbal and nonverbal emotional expressions.
        - It includes audio, visual, and physiological signals, making it suitable for studying multimodal emotion recognition approaches.

6. DEAP (Database for Emotion Analysis using Physiological Signals):

    - The DEAP dataset focuses on emotion analysis using physiological signals, including electroencephalography (EEG), galvanic skin response (GSR), and electromyography (EMG).
    - It provides a rich resource for studying the relationship between physiological signals and emotions.

7. MMI (Multimedia Understanding Group - Maastricht University):

    - The MMI dataset consists of video recordings capturing spontaneous facial expressions in real-world scenarios.
    - It provides a diverse range of emotional expressions and is commonly used for studying video-based emotion recognition.

8. IEMOCAP (Interactive Emotional Dyadic Motion Capture):

    - The IEMOCAP dataset contains multimodal recordings of human-human interactions, including audio, video, and textual transcriptions.
    - It captures emotional expressions in natural and unscripted conversations, making it suitable for studying multimodal emotion recognition in social interactions.

2.2 Drawbacks of the Current System

The current system for emotion detection has a few drawbacks that can be addressed to improve its performance and effectiveness. Some of the common drawbacks include:

1. Limited Emotion Representation:

    - The current system often focuses on recognizing a limited set of basic emotions, such as happiness, sadness, anger, fear, surprise, disgust, and neutral.
    - However, human emotions are complex and diverse, and they cannot be fully captured by these predefined categories.
    - There is a need to incorporate a more comprehensive and nuanced representation of emotions to better reflect the subtleties and variations in human emotional experiences.

2. Context Dependency:

    - Emotions are influenced by various contextual factors, including culture, social norms, and individual differences.
    - The current system often lacks the ability to account for these contextual influences, resulting in potential biases or inaccuracies in emotion detection.

- Incorporating contextual information and considering individual differences can lead to more accurate and culturally sensitive emotion recognition.

3. Cross-Domain Generalization:

    - Emotion detection models trained on specific datasets or modalities may struggle to generalize well to new domains or modalities.
    - The current system often lacks robustness in handling cross-domain scenarios, where the model needs to recognize emotions in different contexts, modalities, or datasets.
    - Enhancing cross-domain generalization capabilities can improve the system's adaptability and performance in real-world applications.

4. Limited Training Data:

    - The availability of labeled training data for emotion detection is often limited, especially for certain domains or specific emotional states.
    - This scarcity of data can hinder the training of accurate and robust emotion detection models.
    - Collecting and annotating more diverse and representative datasets can help overcome this limitation and improve the system's performance.

5. Performance on Unseen or Challenging Data:

    - The current system may struggle to accurately detect emotions in challenging scenarios, such as low-quality images, noisy audio, or non-standardized expressions.
    - Robustness and performance on such unseen or challenging data remain important aspects to be addressed for real-world deployment of emotion detection systems.

2.3 Requirements of the better System

The requirements for a better emotion detection system include:

1. Comprehensive Emotion Representation:

    - To capture the complexity of human emotions, the system should incorporate a broader range of emotion classes.
    - The utilization of the EMOTIC dataset, which consists of 26 emotion classes, allows for a more comprehensive representation of emotions.

2. Utilization of AlexNet and VGG19 Models:

    - Incorporating pre-trained models like AlexNet and VGG19 can leverage their deep learning capabilities for effective feature extraction and emotion recognition.

- These models have demonstrated strong performance in various computer vision tasks, including emotion detection.

3. Transfer Learning using VGG16:

    - Applying transfer learning techniques with VGG16 can further enhance the system's performance.
    - By leveraging the pre-trained weights and learned representations from VGG16, the system can benefit from the knowledge gained from a large-scale dataset, improving its accuracy and generalization capabilities.

2.4 Technical Feasibility

- Technical feasibility is a crucial aspect of implementing an emotion detection system that relies on image processing.
- The following technical requirements need to be considered for a feasible implementation:

    1. Sufficient GPU Power:

        - Emotion detection, particularly when utilizing deep learning models, benefits significantly from GPU acceleration.
        - A system with a powerful GPU, such as NVIDIA GeForce RTX or AMD Radeon, is essential to efficiently process and analyze images in real-time.

    2. Adequate RAM:

        - Emotion detection systems often process large volumes of image data simultaneously.
        - Sufficient RAM, typically 16GB or higher, ensures efficient storage and retrieval of data during image processing tasks, preventing performance bottlenecks.

    3. Powerful Processor:

        - A robust processor, such as Intel Core i7 or AMD Ryzen, is essential for handling the computational requirements of image processing algorithms.
        - This ensures fast and efficient feature extraction and classification of emotions from images.

    4. Image Processing Libraries and Frameworks:

        - Utilizing popular image processing libraries and frameworks, such as OpenCV or TensorFlow, provides access to a rich set of tools and functions for image manipulation, feature extraction, and model training.

- Integration with these libraries ensures a streamlined and efficient workflow.

5. Storage Capacity:

   - Emotion detection systems often require ample storage space to store large datasets, pre-trained models, and processed images.
   - Sufficient storage capacity, either in the form of SSD or HDD, is essential to accommodate these requirements.

6. Bandwidth and Network Connectivity:

   - If the system involves real-time emotion detection or relies on cloud-based processing, a stable internet connection with sufficient bandwidth is necessary for efficient data transfer and communication with remote servers or APIs.

7. Software and Development Environment:

   - Implementing an emotion detection system involves using programming languages like Python, along with frameworks and libraries specific to image processing and deep learning.
   - Ensuring the availability of the necessary software and development environment is crucial for seamless development and deployment.

By considering these technical aspects, including GPU power, RAM capacity, processor performance, image processing libraries, storage capacity, network connectivity, and software environment, the implementation of an emotion detection system becomes technically feasible and efficient in analyzing and recognizing emotions from images.

2.5 Economic Feasibility

The economic feasibility of the project involves considering the costs and potential returns associated with implementing the emotion detection system.

Key aspects to evaluate include:

1. Hardware Costs: Assessing the cost of necessary hardware components, such as GPUs, RAM, and processors, to ensure they are within the project budget.
2. Software Expenses: Considering any expenses related to software licenses, tools, or frameworks required for image processing and deep learning.
3. Development and Integration Expenses: Evaluating the costs associated with hiring skilled professionals, training, or outsourcing development and integration tasks.
4. Maintenance and Support Costs: Factoring in ongoing expenses for system maintenance, updates, bug fixes, hardware upgrades, and technical support.

5. Scalability and Future Expansion: Considering the system's scalability and the potential for future expansion to ensure it can accommodate increased usage, larger datasets, and additional features without significant additional costs.

By analyzing these economic aspects, decision-makers can determine if the project is economically feasible and if the expected benefits and returns justify the investment.

2.6 Operational Feasibility

- The operational feasibility of the project involves training emotion detection models (AlexNet and VGG19) using the EMOTIC dataset consisting of 26 emotion classes.
- The models will be utilized in a flask web application that captures user emotions in real time using the system's camera or uploaded images.
- The application will have an intuitive interface and meet the necessary hardware and software requirements.
- Thorough testing, scalability considerations, and regular maintenance will ensure the operational success of the project.

2.7 Feature of New System

The new system incorporates several key features:

1. Comprehensive Emotion Dataset: The system utilizes the EMOTIC dataset, which encompasses 26 emotion classes. This broadens the range of emotions that can be accurately detected and classified by the models.

2. Training with AlexNet and VGG19 Models: The system employs state-of-the-art deep learning models, namely AlexNet and VGG19, for emotion detection.

3. Transfer Learning using VGG16: To further enhance performance, transfer learning techniques are applied using the pre-trained VGG16 model. This approach allows the system to leverage the knowledge and learned representations of VGG16, resulting in improved accuracy and efficiency during model training.

4. Flask Web Interface: The system incorporates a user-friendly web interface created using Flask, a Python web framework. This interface allows users to interact with the system in real time, either by capturing emotions using the device's camera or by uploading images from the device.

5. Real-time Emotion Prediction: Leveraging the camera or uploaded images, the system utilizes the trained models to predict emotions in real time. The models analyze the input data and provide instant feedback on the detected emotions, enabling quick and efficient emotion recognition.

3 HARDWARE AND SOFTWARE REQUIREMENTS:

3.1 User Side Software Requirements

Table 1 User Side Software Requirements

Component	Minimum	Recommended
PROCESSOR	Intel Core i3 or Equivalent to AMD	Intel Core i5 or Equivalent to AMD
RAM	4 GB	8 GB
DISPLAY	800 x 600	1920 x 1080
HDD/SSD	512 GB	1 TB/ 512 GB
CAMERA	Webcam (minimum 720p)	Webcam (1080p or above)
OS	Windows 7 or 8	Windows 10 or 11

3.2 Developer Side Software Requirements

Table 2 Developer Side Software Requirements

Component	Minimum	Recommended
PROCESSOR	Intel Core i5 or Equivalent to AMD	Intel Core i7 or Equivalent to AMD
RAM	8 GB	16 GB
DISPLAY	800 x 600	1920 x 1080
HDD/SSD	1 TB/ 512 GB	1 TB/ 1 TB
GRAPHICS CARD	NVIDEA GTX 1650 or Equivalent	NVIDEA RTX 3060 or Equivalent
CAMERA	Webcam (minimum 720p)	Webcam (1080p or above)
OS	Windows 7 or 8	Windows 10 or 11

4 EMOTIC DATASET

4.1 About the Dataset

- The EMOTIC dataset is a comprehensive collection of images of people in various real-world settings, annotated according to their apparent emotional states.
- The dataset consists of a total of more than 23,571 images and 34,320 annotated people.
- Some of the images were manually collected from the internet using Google search engine, employing different queries related to places, social environments, activities, and emotional states.
- Additionally, a portion of the images is sourced from two public benchmark datasets, COCO and Ade20k, adding diversity to the dataset.
- The dataset exhibits a wide range of contexts, featuring people in different locations, social settings, and engaged in various activities.
- The annotations of the images were performed using Amazon Mechanical Turk (AMT), where annotators were asked to label the emotional states, they perceived the people in the images to be experiencing.
- The annotations capture a range of emotions, and the dataset also includes continuous dimensions such as Valence, Arousal, and Dominance.
- To ensure a reliable and consistent annotation process, the dataset underwent multiple rounds of annotations.
- The images were divided into three sets: Training (70%), Validation (15%), and Testing (15%).
- Additional annotators were involved in annotating the Validation and Testing sets, resulting in a total of 5 annotators for the Validation set and 3 annotators for the Testing set. This approach helps evaluate the consistency of annotations and ensures the quality of the dataset.
- The EMOTIC dataset provides researchers with a rich resource for studying human emotions in real-world scenarios.
- Its annotations allow for the exploration of various emotional states and the examination of the inter-annotator consistency.
- The dataset statistics and algorithmic analysis conducted on the EMOTIC dataset further contribute to its comprehensiveness and usefulness for emotion detection research.

4.2 Annotations in Emotic Dataset

The annotations of the EMOTIC dataset combine two types of emotion representation systems: Discrete Categories and Continuous Dimensions.

4.2.1 Discrete Categories

List of the 26 Categories with corresponding definitions:

1. Peace: wellbeing and relaxed; no worry; having positive thoughts or sensations; satisfied.

2. Affection: fond feelings; love; tenderness

3. Esteem: feelings of favorable opinion or judgment; respect; admiration; gratefulness

4. Anticipation: state of looking forward; hoping on or getting prepared for possible future events

5. Engagement: paying attention to something; absorbed into something; curious; interested

6. Confidence: feeling of being certain; conviction that an outcome will be favorable; encouraged; proud

7. Happiness: feeling delighted; feeling enjoyment or amusement

8. Pleasure: feeling of delight in the senses

9. Excitement: feeling enthusiasm; stimulated; energetic

10. Surprise: sudden discovery of something unexpected

11. Sympathy: state of sharing others' emotions, goals or troubles; supportive; compassionate

12. Doubt/Confusion: difficulty to understand or decide; thinking about different options

13. Disconnection: feeling not interested in the main event of the surrounding; indifferent; bored; distracted

14. Fatigue: weariness; tiredness; sleepy

15. Embarrassment: feeling ashamed or guilty

16. Yearning: strong desire to have something; jealous; envious; lust

17. Disapproval: feeling that something is wrong or reprehensible; contempt; hostile

18. Aversion: feeling disgust, dislike, repulsion; feeling hate

19. Annoyance: bothered by something or someone; irritated; impatient; frustrated

20. Anger: intense displeasure or rage; furious; resentful

21. Sensitivity: feeling of being physically or emotionally wounded; feeling delicate or vulnerable

22. Sadness: feeling unhappy, sorrow, disappointed, or discouraged

23. Disquietment: nervous; worried; upset; anxious; tense; pressured; alarmed

24. Fear: feeling suspicious or afraid of danger, threat, evil or pain; horror

25. Pain: physical suffering

26. Suffering: psychological or emotional pain; distressed; anguished

4.2.2 Continuous Dimensions

Continuous dimensions with their descriptions:

Valence: this dimension measures how positive or pleasant an emotion is. It ranges from *negative* to *positive*.

Arousal: this dimension measures the agitation level of the person. It ranges from *non-active / in calm* to *agitated / ready to act*.

Dominance: this dimension measures the control level of the situation by the person. It ranges from *submissive / non-control* to *dominant / in-control*.

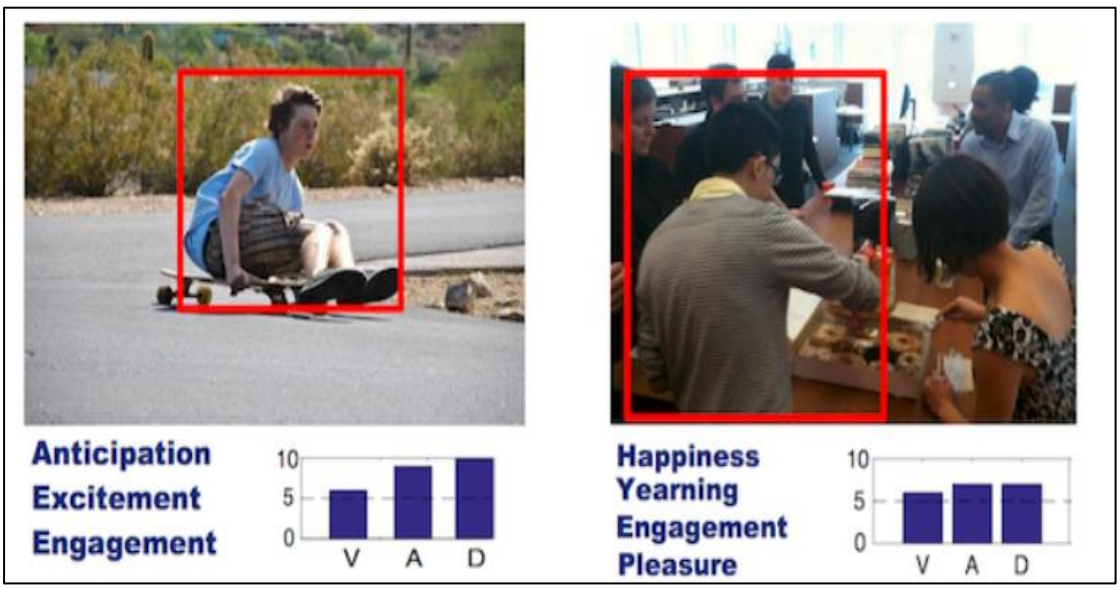

Figure 4.1 EMOTIC Dataset

5 IMPLEMENTATIONS

5.1 Introduction

The EMOTIC dataset is a large-scale benchmark for emotion recognition in context. It contains more than 24,000 images annotated with 26 fine-grained emotion categories and 2 additional neutral categories, as well as contextual information such as gender, age, and activity. The dataset was developed to provide a more realistic and challenging environment for emotion recognition than previous datasets, which often focused on posed expressions in controlled environments.

The significance of the EMOTIC dataset lies in its ability to address the limitations of previous emotion recognition datasets and provide a more comprehensive and accurate representation of real-world emotional experiences. The inclusion of contextual information allows for a more nuanced understanding of the emotional states of individuals in different situations, which can have important implications for various fields such as psychology, neuroscience, and human-computer interaction. Additionally, the size and diversity of the dataset make it an ideal benchmark for evaluating and comparing different emotion recognition models and techniques.

5.2 Data Preprocessing

After the downloading EMOTIC dataset, Annotation of dataset is derived by. mat to .csv code converter. Using this file, the annotation file of the dataset was derived based on which separation of images according to the emotion label. We have derived a code which can segregate the Images according to their respective emotion mention in annotation file. 26 Folders of Emotions are created and images are separated according to it.

The EMOTIC dataset was collected by combining multiple existing image datasets and annotating them with emotion labels. The annotations were obtained from the authors and were processed to generate separate folders for each emotion label. Each folder contains the images that were annotated with the corresponding emotion label. The dataset was then divided into three subsets: train, validation, and test, which are used for the model training and testing phases.

It first loops through each emotion folder in the original dataset and shuffles the images in each folder. Then, it calculates the number of images to put in each set based on the splits defined (70% for training, 15% for validation, and 15% for test), and copies the images to the corresponding directories.

The train_dir, val_dir, and test_dir variables specify the paths to the directories where the split dataset will be stored. The os.makedirs() function is used to create the directory paths if they don't already exist, and the shutil.copyfile() function is used to

copy the image files from the original dataset to the corresponding directories in the split dataset.

5.2.1 Steps for Pre-processing

The Following Data Pre-processing Steps were performed before model training:

1. Resizing the images to a fixed size: AlexNet and VGG19 were both trained on 224x224 pixel images, so it's common to resize the Emotic dataset images to this size before training. This can be done using an image processing library like Pillow or OpenCV.
2. Applying data augmentation: Data augmentation techniques, like the ones shown in the code snippet you provided, can be used to artificially increase the size of the training dataset and improve the generalization of the model. Common data augmentation techniques include random rotations, flips, zooms, and crops.
3. Normalizing the pixel values: Before training, it's common to normalize the pixel values of the images to have zero mean and unit variance. This can be done using the mean and standard deviation of the pixel values of the training dataset.
4. Splitting the data into training, validation, and testing sets: It's important to split the data into separate sets for training, validation, and testing to evaluate the performance of the model on unseen data. This can be done using the `train_test_split()` function from scikit-learn or a similar library.
5. Using Image Data Generator:
   Using data generators is useful for working with large datasets that cannot be loaded entirely into memory. The generators generate batches of data on the fly, which can be fed into the model for training or validation.

5.2.2 Data Generators

Using data generators is useful for working with large datasets that cannot be loaded entirely into memory. The generators generate batches of data on the fly, which can be fed into the model for training or validation. Applying data augmentation can help to improve the generalization of the model, while preprocessing can help to normalize the pixel values of the images.

The train and validation data generators play an important role in data preprocessing before training on AlexNet and VGG19.

The train generator is used to generate batches of training data from the train directory. It applies various data augmentation techniques such as flipping, shifting, shearing, zooming, and rotating the images. This helps to artificially increase the size of the training dataset and improve the generalization of the model. By randomly

applying these transformations, it also ensures that the model is exposed to a diverse range of training examples and helps to prevent overfitting.

The validation generator is used to generate batches of validation data from the validation directory. It applies only the preprocessing function, without any data augmentation, to ensure that the validation data is not artificially modified in any way. This ensures that the model is evaluated on data that is representative of the real-world distribution and helps to estimate the model's generalization performance.

Using data generators in this way also has the advantage of allowing the model to be trained on large datasets that may not fit entirely into memory. By generating batches of data on-the-fly, it avoids the need to load the entire dataset into memory at once.

Overall, the train and validation data generators are important for ensuring that the model is exposed to a diverse range of training examples, while also being evaluated on data that is representative of the real-world distribution. This can help to improve the model's accuracy and generalization performance.

5.2.3 Pros and Cons of Pre-Processing

5.2.3.1 Pros:

1. Resizing the images to a fixed size: This can help to ensure that all images have the same dimensions and can be fed into the model. It can also help to reduce the amount of memory required to store the images.
2. Applying data augmentation: This can help to artificially increase the size of the training dataset and improve the generalization of the model. It can also help to reduce overfitting by introducing randomness and diversity in the training data.
3. Normalizing the pixel values: This can help to ensure that the pixel values have a similar scale and distribution, which can improve the stability of the training process and make it easier for the model to learn.
4. Splitting the data into training, validation, and testing sets: This can help to ensure that the model is evaluated on unseen data and can generalize well. It can also help to prevent overfitting by monitoring the performance of the model on a separate validation set.

5.2.3.2 Cons:

1. Resizing the images to a fixed size: This can lead to some loss of information or distortion in the images if the aspect ratio is not preserved. It can also result in a loss of detail if the original images have a higher resolution than the fixed size.
2. Applying data augmentation: This can introduce randomness and diversity in the training data, but it can also result in some unrealistic or irrelevant images. It can also increase the training time and computational resources required.
3. Normalizing the pixel values: This can make the data more suitable for the model to learn from, but it can also introduce some bias if the normalization is not done properly. It can also increase the computational resources required.

4. Splitting the data into training, validation, and testing sets: This can help to prevent overfitting and ensure that the model generalizes well, but it can also reduce the amount of data available for training and increase the variability in the performance metrics due to the random split. It can also lead to some bias if the split is not representative of the overall dataset.

5.3 Model

The function **load_pretrained_model** is used to load different pre-trained models based on the provided model name. Here's an explanation of how each model is defined:

1. **AlexNet**: This model is defined as a sequential model. It consists of 8 layers:
    - Layer 1: Convolutional layer with 96 filters, kernel size of (11,11), and stride of (4,4). ReLU activation is applied, and the input shape is (224, 224, 3).
    - Batch normalization layer is added.
    - Max pooling layer with pool size (3,3) and stride (2,2) is added.
    - Layer 2: Convolutional layer with 256 filters, kernel size of (5,5), and stride of (1,1). ReLU activation is applied, and padding is set to "same".
    - Batch normalization layer is added.
    - Max pooling layer with pool size (3,3) and stride (2,2) is added.
    - Layer 3: Convolutional layer with 384 filters, kernel size of (3,3), and stride of (1,1). ReLU activation is applied, and padding is set to "same".
    - Batch normalization layer is added.
    - Layer 4: Convolutional layer with 384 filters, kernel size of (3,3), and stride of (1,1). ReLU activation is applied, and padding is set to "same".
    - Batch normalization layer is added.
    - Layer 5: Convolutional layer with 256 filters, kernel size of (3,3), and stride of (1,1). ReLU activation is applied, and padding is set to "same".
    - Batch normalization layer is added.

- Max pooling layer with pool size (3,3) and stride (2,2) is added.
- Layer 6: Flatten layer to convert the output into a 1D tensor.
- Fully connected layer with 4096 units and ReLU activation is added.
- Dropout layer with a dropout rate of 0.5 is added.
- Layer 7: Fully connected layer with 4096 units and ReLU activation is added.
- Dropout layer with a dropout rate of 0.5 is added.
- Layer 8: Fully connected layer with 26 units (output classes) and softmax activation is added.

2. **VGG19**: This model uses the VGG19 architecture with pre-trained weights from the ImageNet dataset. The pre-trained layers are frozen, and two additional layers are added:
    - Flatten layer to convert the output into a 1D tensor.
    - Fully connected layer with 64 units and ReLU activation is added.
    - Output layer with 26 units (output classes) and no activation function is added.
3. **AlexNet_TL**: This model combines the AlexNet architecture with transfer learning using the VGG16 base model. The AlexNet layers are defined similarly to the first model, and the VGG16 layers are frozen to retain the pre-trained weights. The model consists of the following layers:
    - AlexNet layers (similar to the first model).
    - VGG16 base model with frozen weights.
    - Flatten layer to convert the output into a 1D tensor.
    - Fully connected layer with 256 units and ReLU activation is added.
    - Dropout layer with a dropout rate of 0.5 is added.
    - Output layer with 26 units (output classes) and softmax activation is added.
4. **VGG19_TL**: This model utilizes the VGG19 architecture with transfer learning. The pre-trained VGG19 layers are frozen, and two additional layers are added for fine-tuning the model for a specific task. The model consists of the following layers:
    - VGG19 base model with frozen weights.

- Flatten layer to convert the output into a 1D tensor.
- Fully connected layer with 1024 units and ReLU activation is added.
- Output layer with 26 units (output classes) and softmax activation is added.

In both transfer learning models (AlexNet_TL and VGG19_TL), the pre-trained layers are frozen to retain their learned features from the ImageNet dataset. Only the additional layers introduced after the pre-trained base model are trained from scratch to adapt to a specific task or dataset.

The function returns the constructed model based on the provided model name. If an invalid model name is passed, a **ValueError** is raised.

5.3.1 Training Procedure

5.3.1.1 Train all models together

The train_all_models() function trains several pre-trained deep learning models on the EMOTIC dataset. Specifically, the function trains four models: ALEXNET, ALEXNET_TL, VGG19, VGG19_TL.

The function uses the load_pretrained_model(model_name) function to load the pre-trained model specified by the model_name parameter. The pre-trained models are loaded without the last classification layer. The last layer is replaced with a new classification layer that outputs a probability distribution over the six emotion classes.

For the ALEXNET and VGG19 models, the function uses the tf.keras.applications module to load the pre-trained models. The load_pretrained_model(model_name) function loads the pre-trained models with their respective weights, excluding the last classification layer. Then, the last layer is added back and retrained on the EMOTIC dataset.

The function trains each model for no. of epochs mentioned by user. Using the different optimizer with a different learning rate. It compiles each model with categorical cross-entropy loss and accuracy metrics. It then fits the training and validation data generators to the model.

Finally, the function evaluates the performance of the model on the validation set by calculating its loss and accuracy. It also creates a confusion matrix to visualize the model's predictions on the validation set. The results for each model are stored in a dictionary and saved as CSV files.

5.3.1.2 Training Individual Model

Giving Reference for one code file, we are loading a pre-trained AlexNet model and compiling it with the Adam optimizer, a categorical cross-entropy loss function, and an accuracy metric. We are then training the model on the EMOTIC dataset using the fit() method with a batch size of 56, 25 epochs, and a learning rate of 0.001.

The EMOTIC dataset is likely a dataset of images and labels that are used for training the AlexNet model. The train_generator and validation_generator are likely data generators that are used to feed batches of data to the model during training and validation, respectively. The steps_per_epoch and validation_steps arguments specify the number of steps (batches) to yield from the generators during each epoch of training and validation.

The class_weightss argument specifies the weights to be assigned to each class during training. This can be useful if the dataset is imbalanced and some classes have significantly fewer examples than others.

Finally, by varying the batch size, epoch, and learning rate parameters, we are experimenting with different hyperparameter configurations to see if we can improve the efficiency of the model training. These hyperparameters can significantly affect the performance of the model, so finding the optimal values can lead to better accuracy and faster convergence during training.

5.4 Evaluation Metrics

We have the following evaluation metrics:

Accuracy: This is the ratio of the number of correctly classified samples to the total number of samples in the dataset.

Loss: This is the error between the predicted output and the actual output.

Validation accuracy: This is the accuracy of the model on a validation set.

Validation loss: This is the loss of the model on a validation set.

Precision: Precision is the ratio of true positives to the sum of true positives and false positives.

Recall: Recall is the ratio of true positives to the sum of true positives and false negatives.

F1 score: The F1 score is the harmonic mean of precision and recall.

Confusion matrix: The confusion matrix is a tool used to visualize the performance of a classification model. It shows the number of correct and incorrect predictions made by the model compared to the actual outcomes.

Co-occurrence matrix: The Co-occurrence Matrix is used to find how the images in the test folder are co-related to emotion. Model predicts every image in the test set and Co-occurrence matrix is formed.

5.5 Result Analysis

Result Analysis for the ALEXNET, VGG19 Models train on 10 Epoch with learning rate of 0.1 and batch size of 56 using 3 Optimizer.

5.5.1 Accuracy

Table 3 Accuracy of Models

Model Name	Adam	SGD	RMSprop
ALEXNET	19.82902288	24.55987781	18.30399185
VGG19	1.745997183	25.51981509	0.4564225208
ALEXNET_TL	22.46975303	19.54647601	24.65768307
VGG19_TL	21.28522843	19.87973601	22.31036723

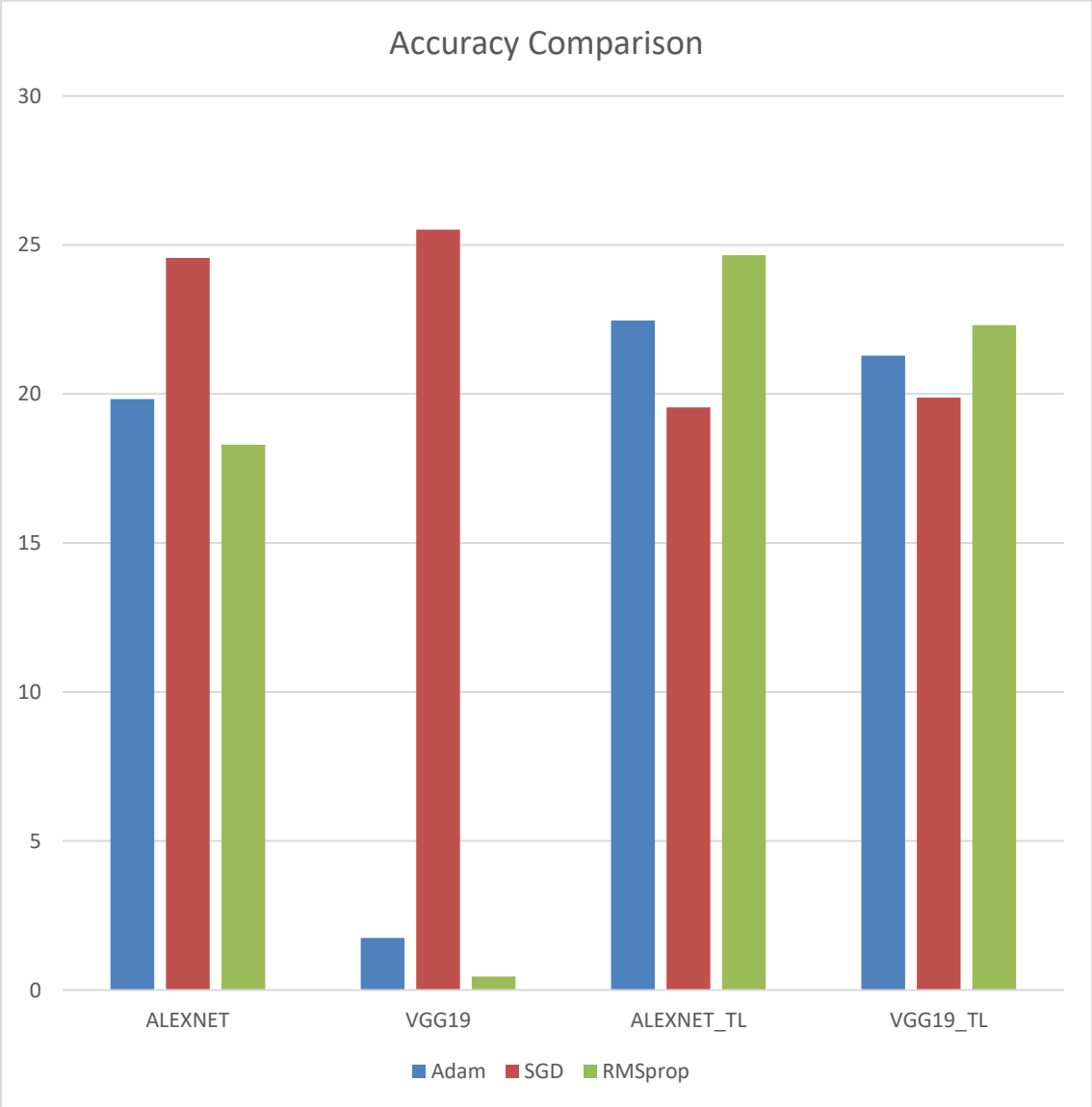

Figure 5.1 Accuracy Comparison

5.5.2 Valid Accuracy

Table 4 Valid Accuracy of Models

Model Name	Adam	SGD	RMSprop
ALEXNET	25.35714209	23.63945544	25.47619045
VGG19	1.683673449	25.5952388	0.4081632476
ALEXNET_TL	25.54421723	25.6462574	25.3231287
VGG19_TL	22.26190418	25.5952388	24.0476191

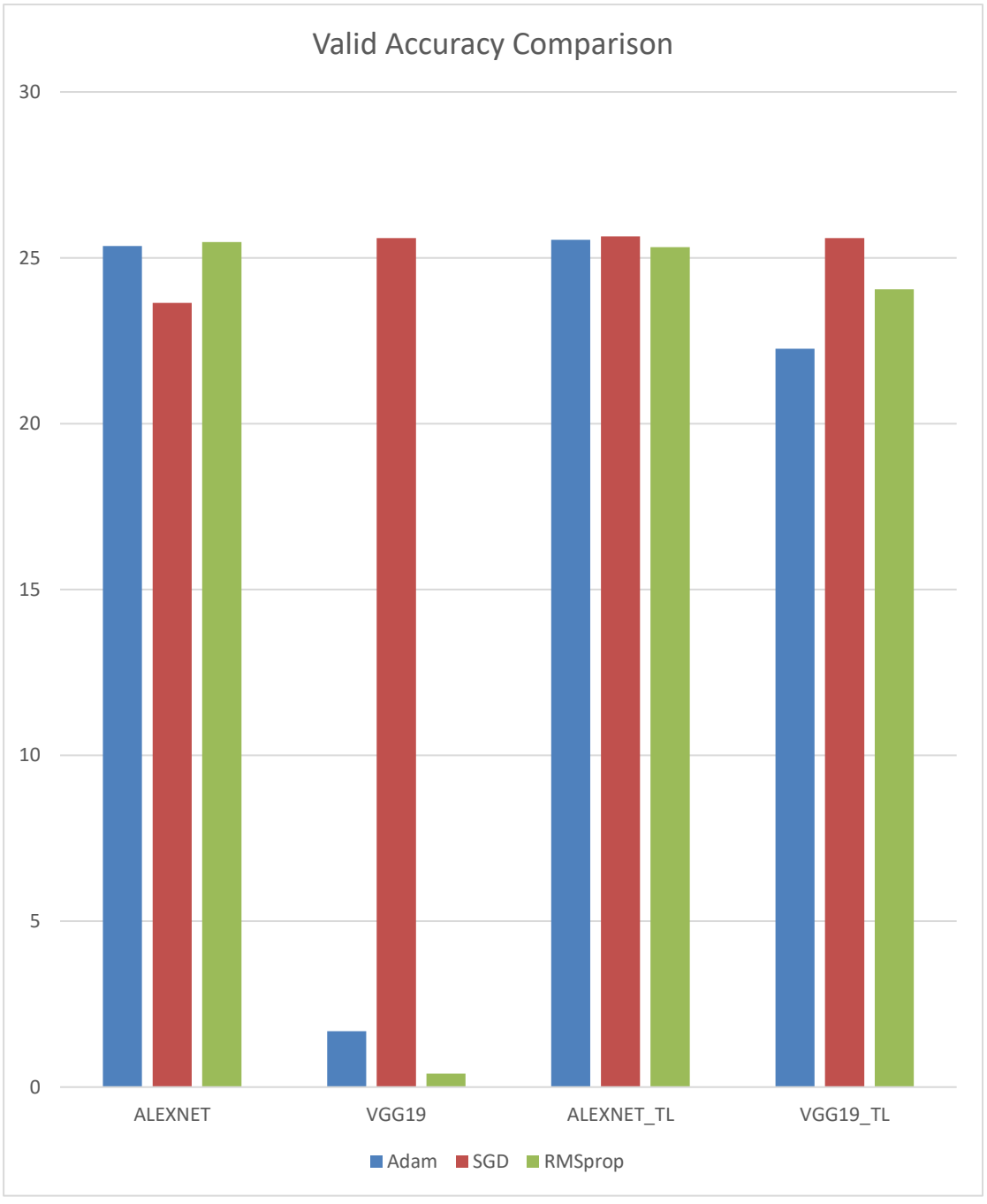

Figure 5.2 Valid Accuracy Comparison

5.5.3 Loss

Table 5 Loss of Models

Model Name	Adam	SGD	RMSprop
ALEXNET	20663.06763	NAN	20486.06262
VGG19	43480.21851	60975.27466	117584.0088
ALEXNET_TL	19799.45526	20046.98792	20111.94916
VGG19_TL	19221.86279	20049.83215	19573.79761

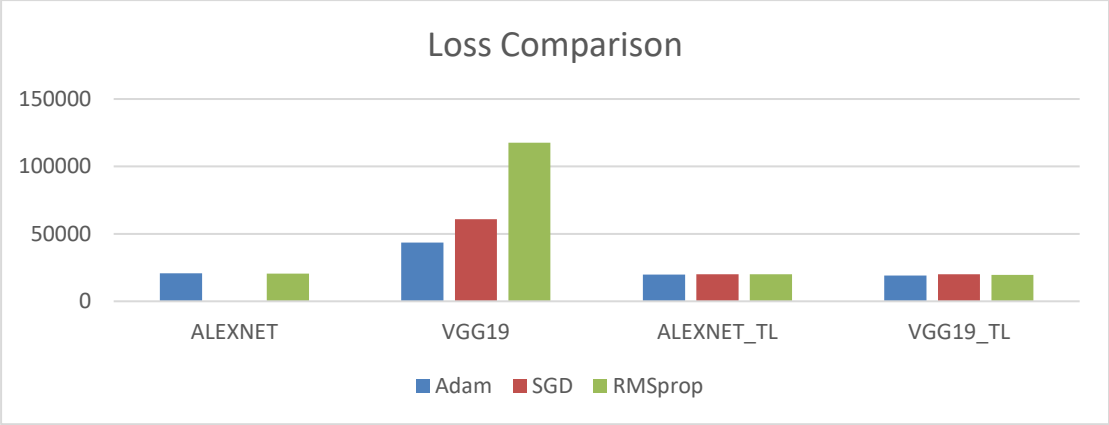

Figure 5.3 Loss Comparison

5.5.4 Valid Loss

Table 6 Valid Loss of Models

Model Name	Adam	SGD	RMSprop
ALEXNET	293.5677528	NAN	298.9732504
VGG19	626.0838509	567.1485901	1103.322124
ALEXNET_TL	299.1464615	308.0487251	304.6018124
VGG19_TL	295.1425552	310.0443125	294.5991993

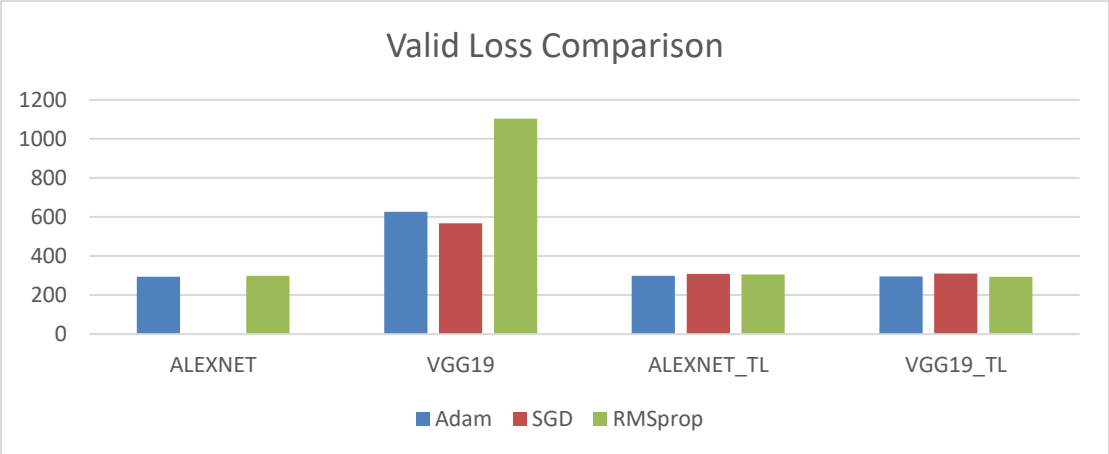

Figure 5.4 Valid Loss Comparison

5.5.5 Confusion Matrix and Co-Occurrence Matrix

5.5.5.1 Adam Optimizer

1. ALEXNET

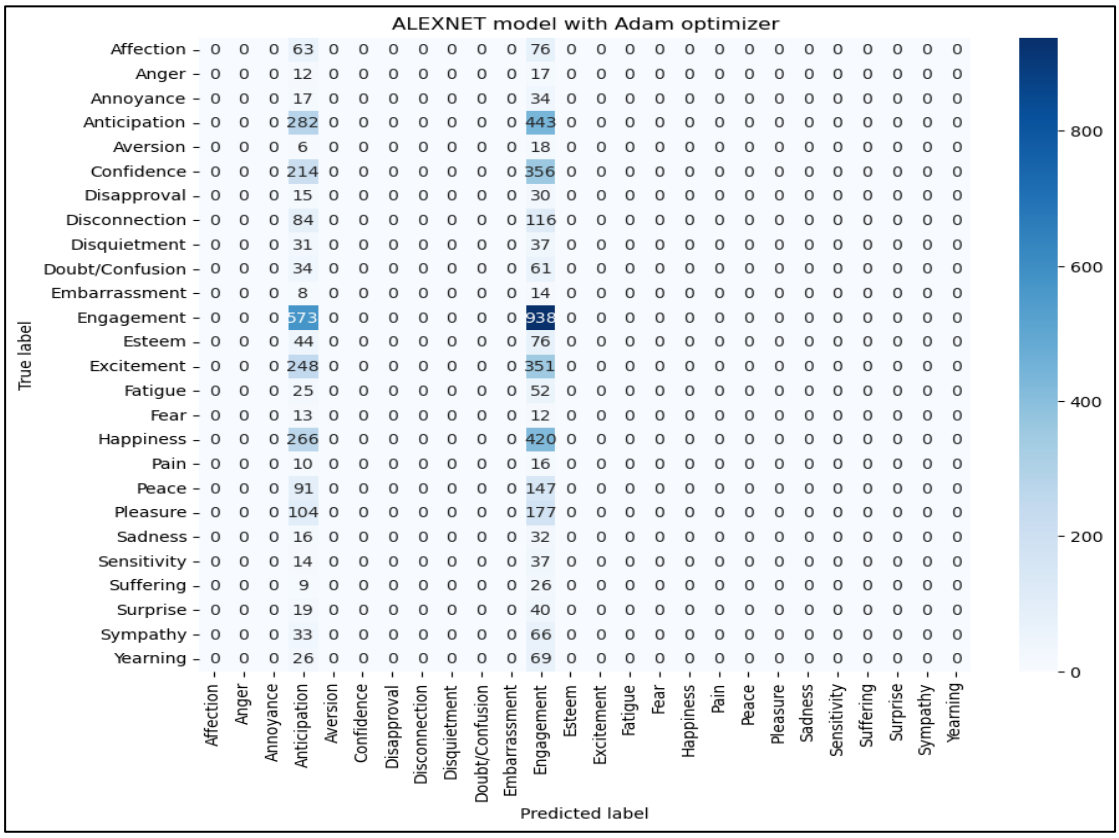

Figure 5.5 ALEXNET's Confusion Matrix for Adam Optimiser

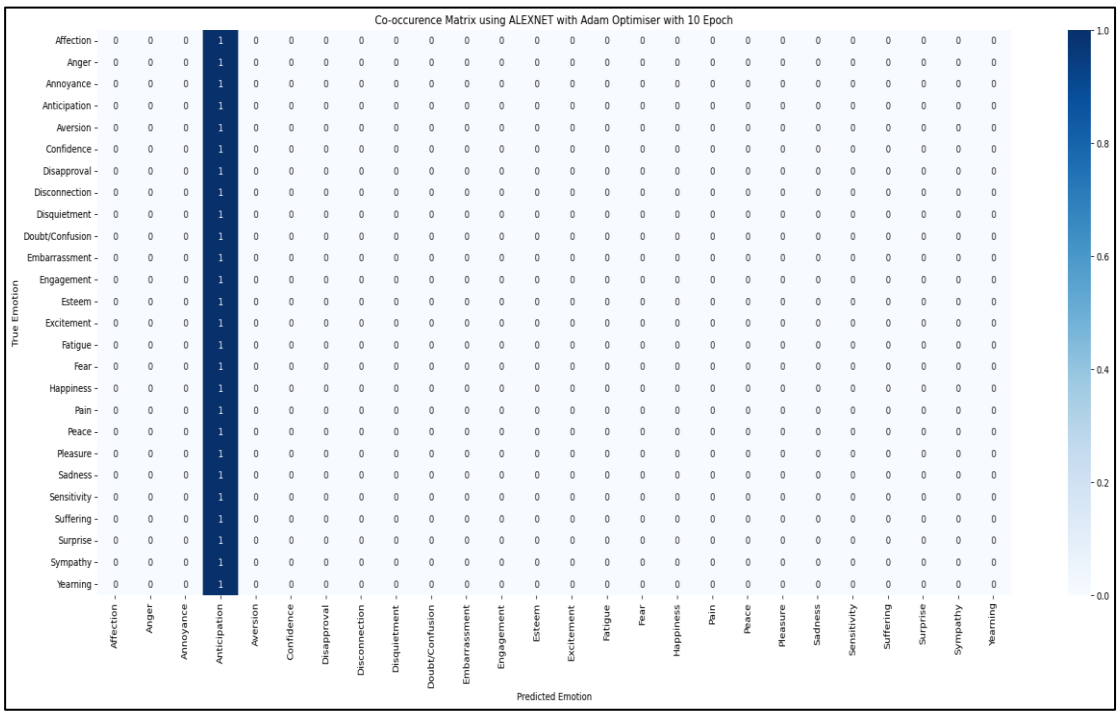

Figure 5.6 ALEXNET's Co-occurrence Matrix for Adam Optimiser

2. VGG19

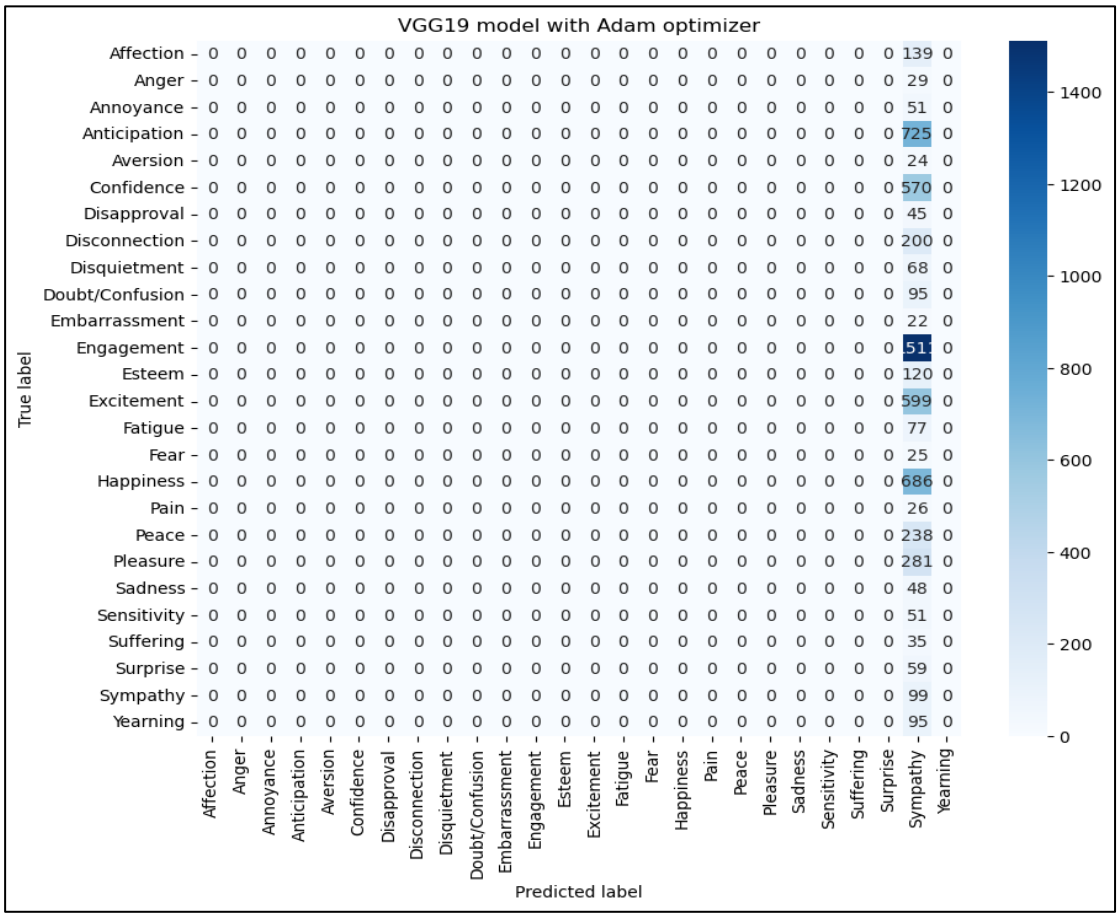

Figure 5.7 VGG19's Confusion Matrix for Adam Optimiser

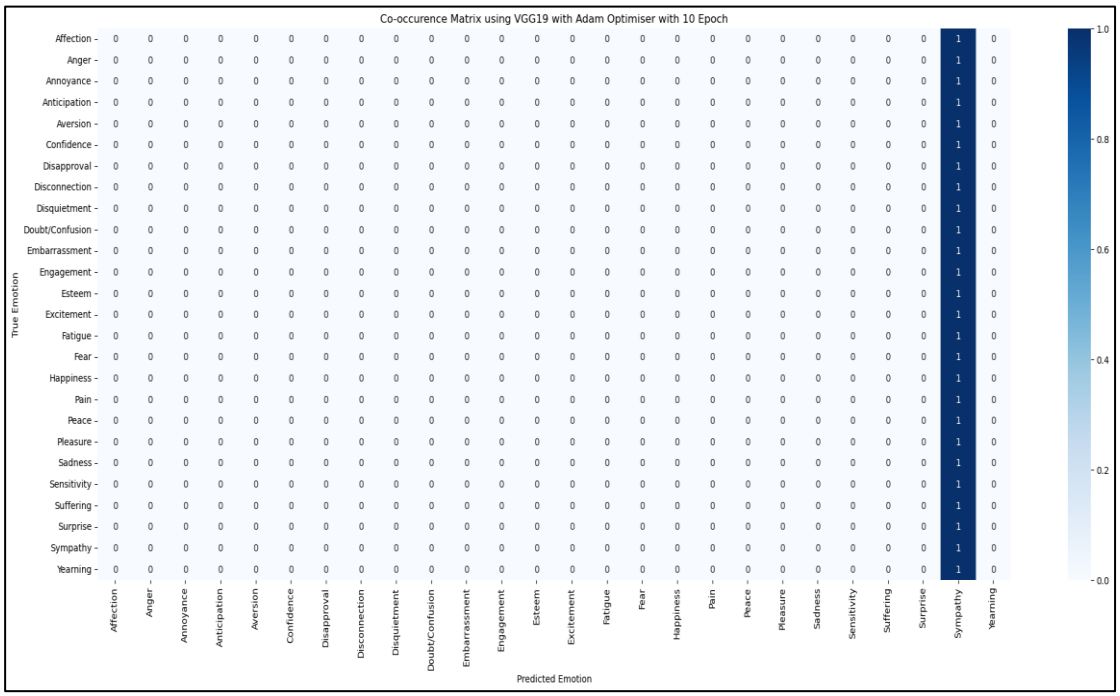

Figure 5.8 VGG19's Co-occurence Matrix for Adam Optimiser

3. ALEXNET_TL

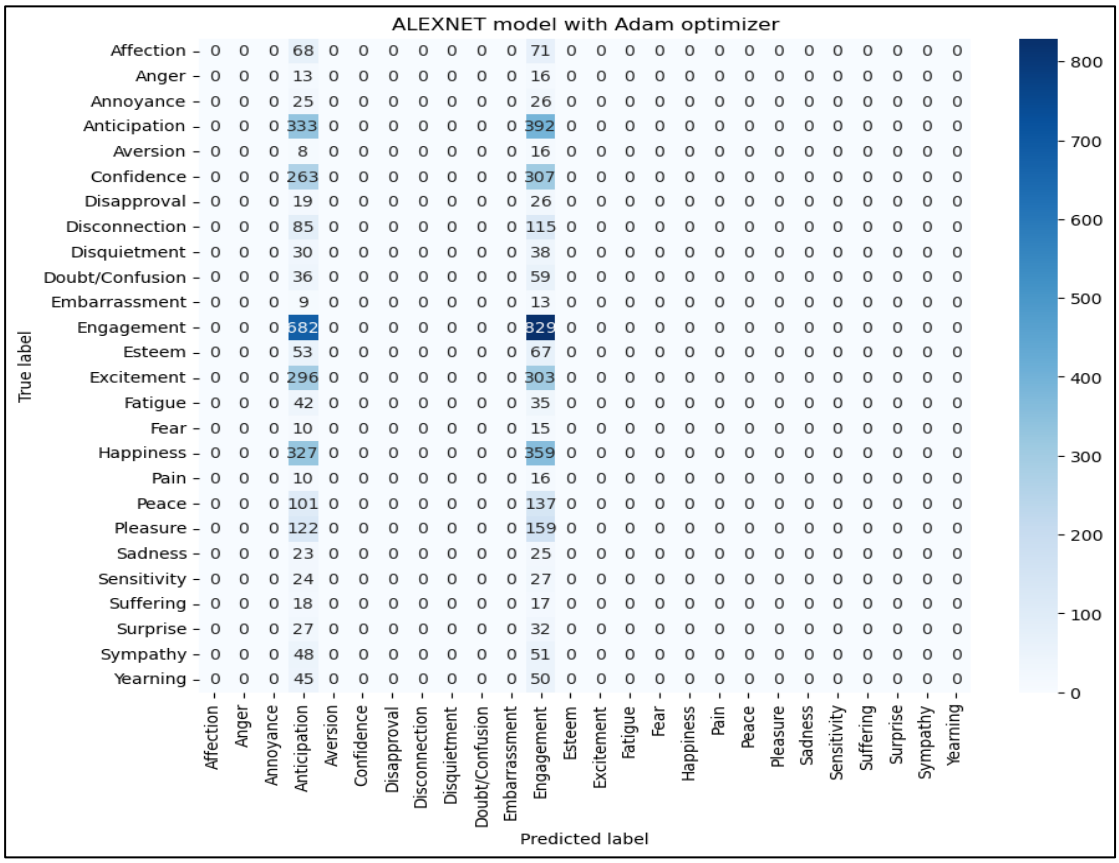

Figure 5.9 ALEXNET_TL's Confusion Matrix for Adam Optimiser

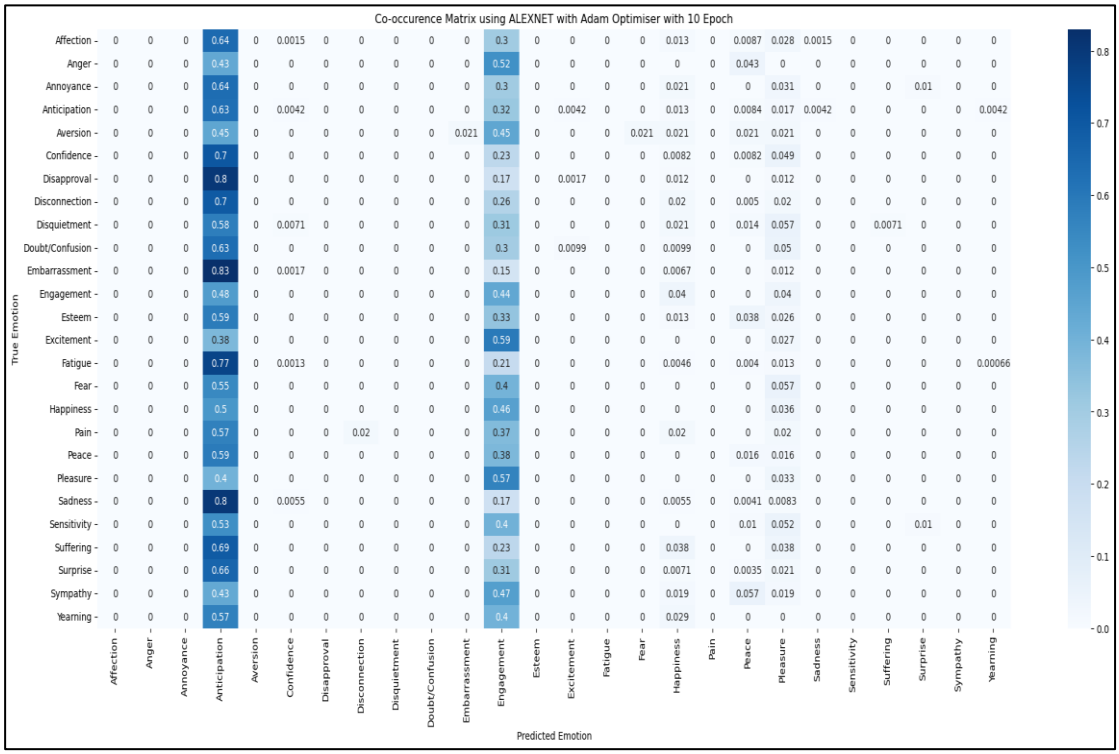

Figure 5.10 ALEXNET_TL's Co-occurence Matrix for Adam Optimiser

4. VGG19_TL

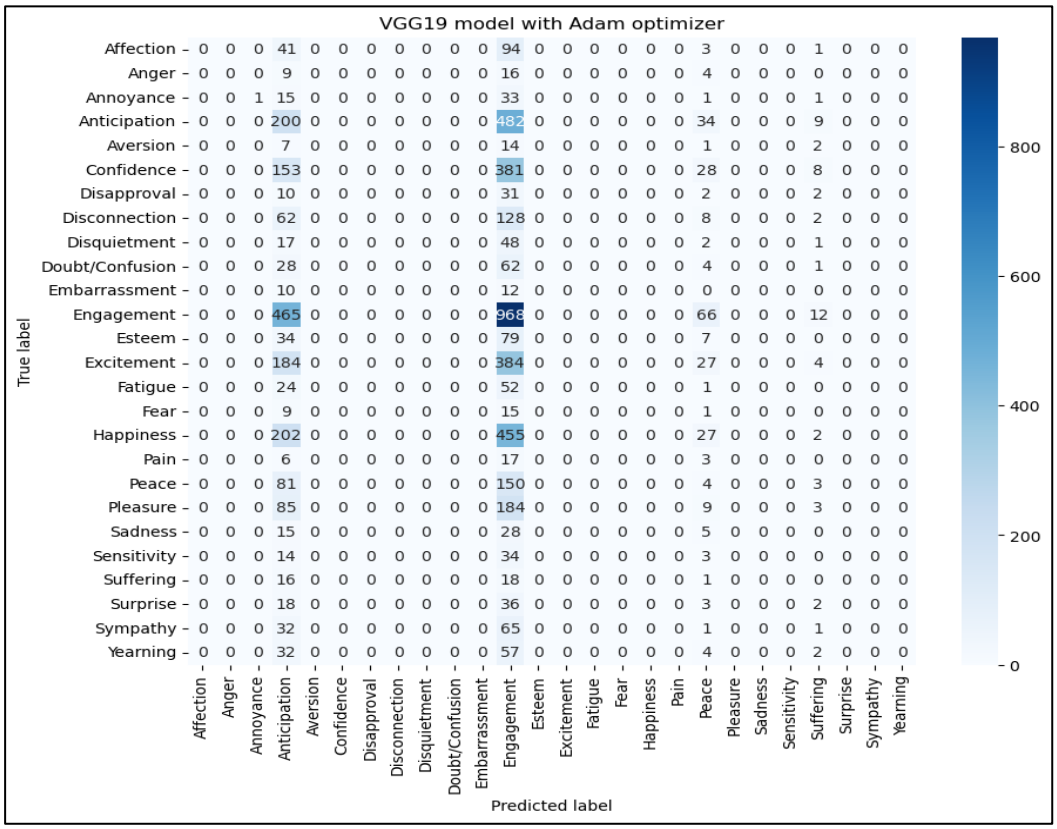

Figure 5.11 VGG19_TL's Confusion Matrix for Adam Optimiser

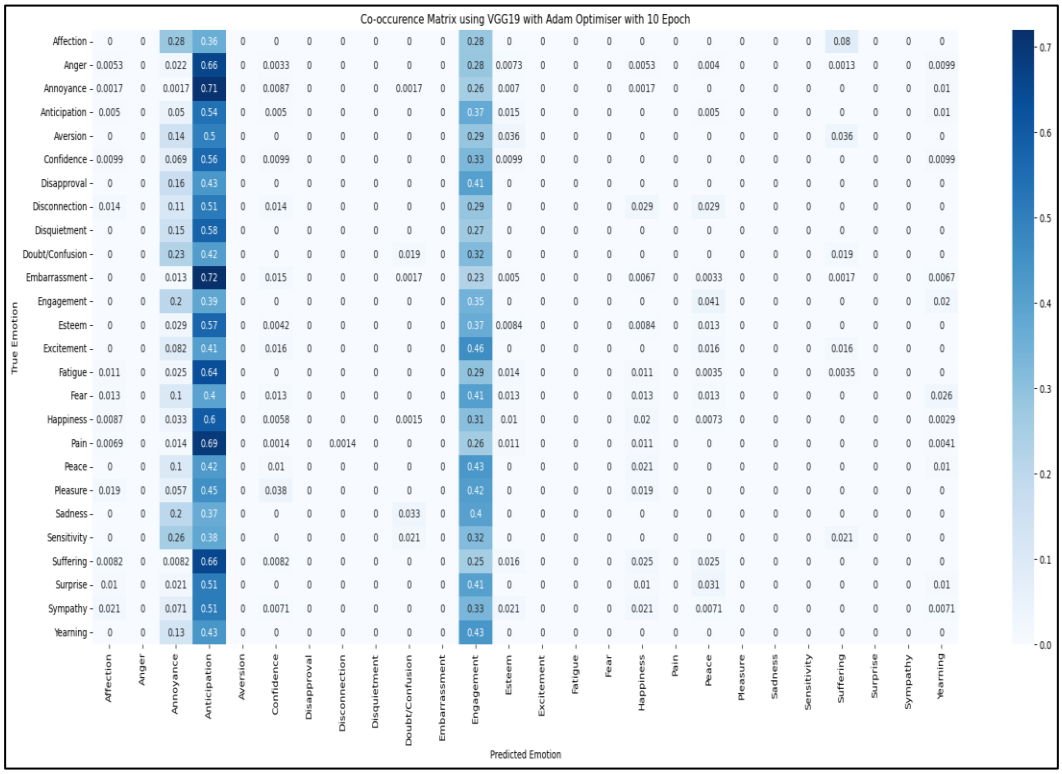

Figure 5.12 VGG19_TL's Co-occurrence Matrix for Adam Optimiser

5.5.5.2 SGD

1. ALEXNET

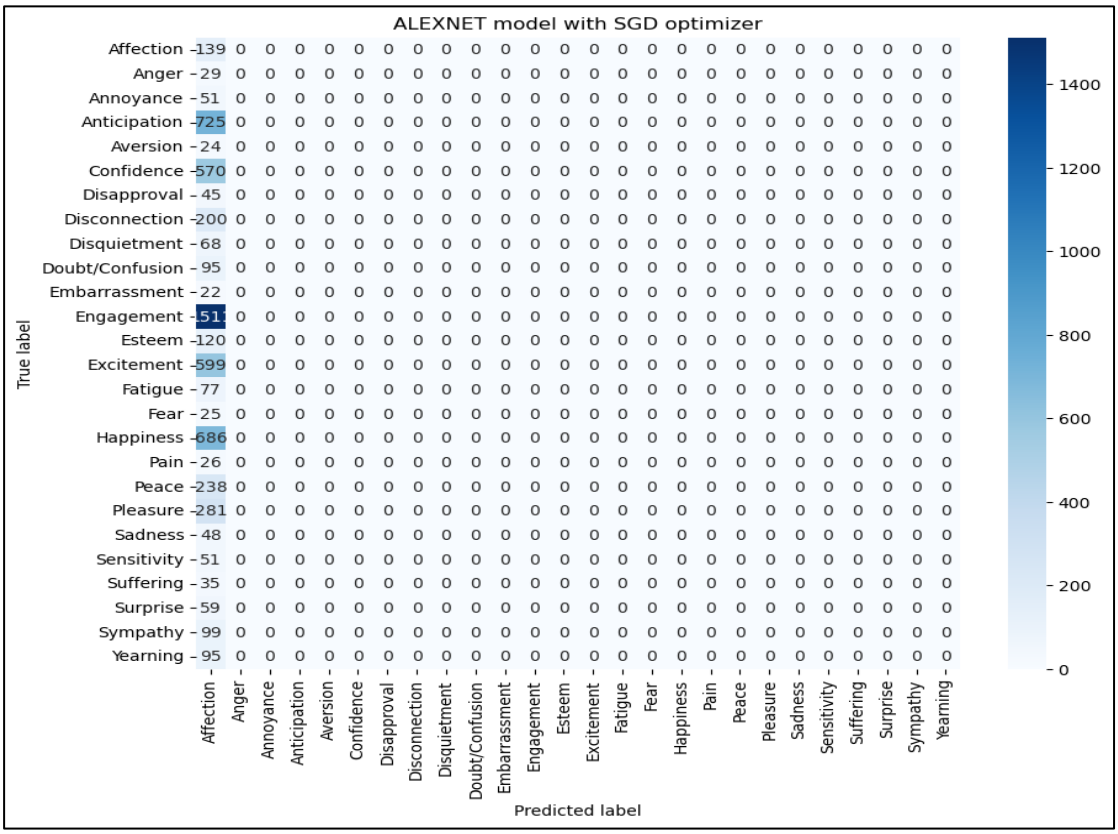

Figure 5.13 ALEXNET's Confusion Matrix for SGD Optimiser

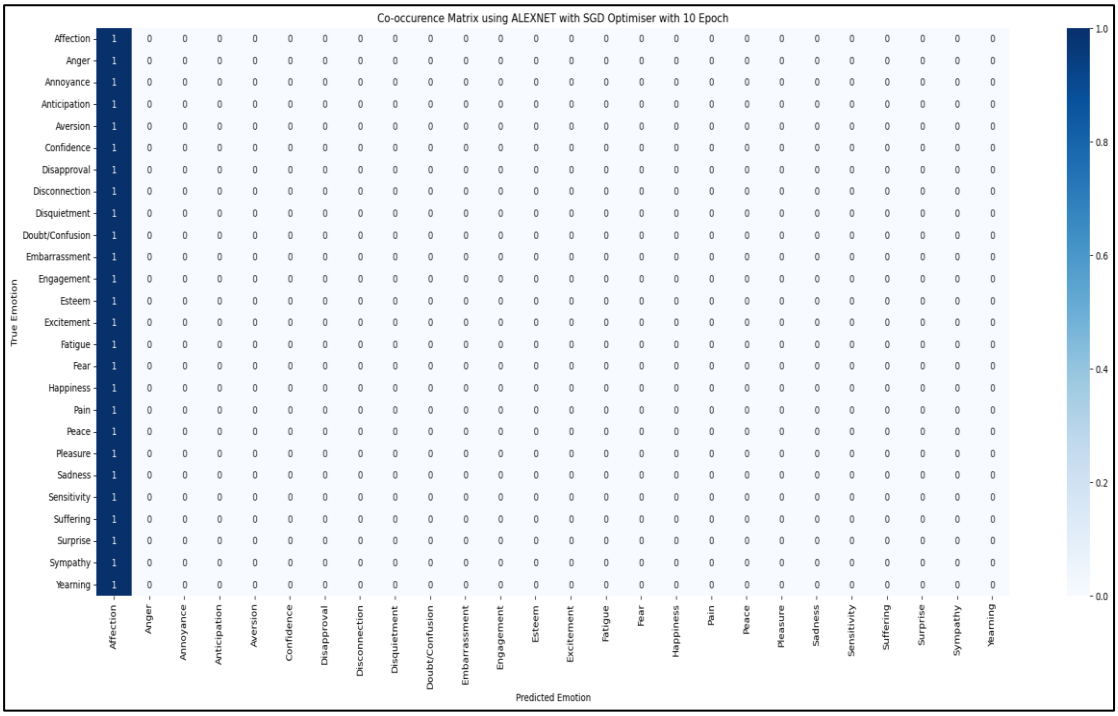

Figure 5.14 ALEXNET's Co-occurence Matrix for SGD Optimiser

2. VGG19

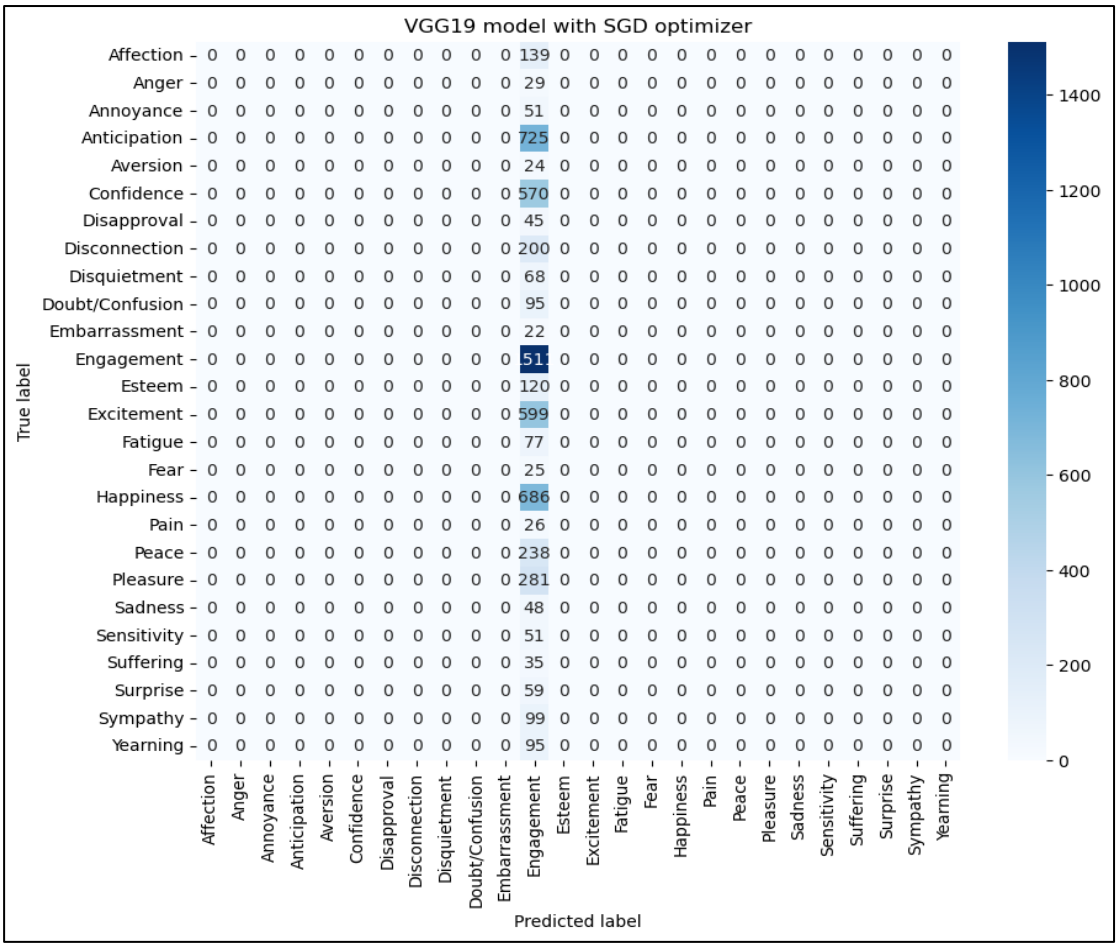

Figure 5.15 VGG19's Confusion Matrix for SGD Optimiser

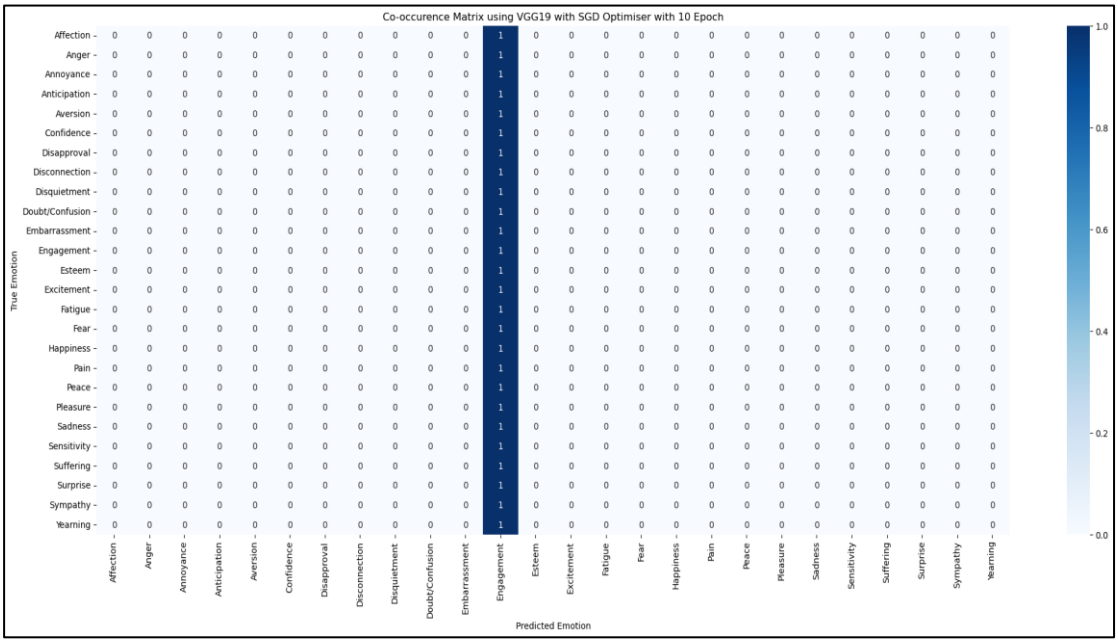

Figure 5.16 VGG19's Co-occurence Matrix for SGD Optimiser

3. ALEXNET_TL

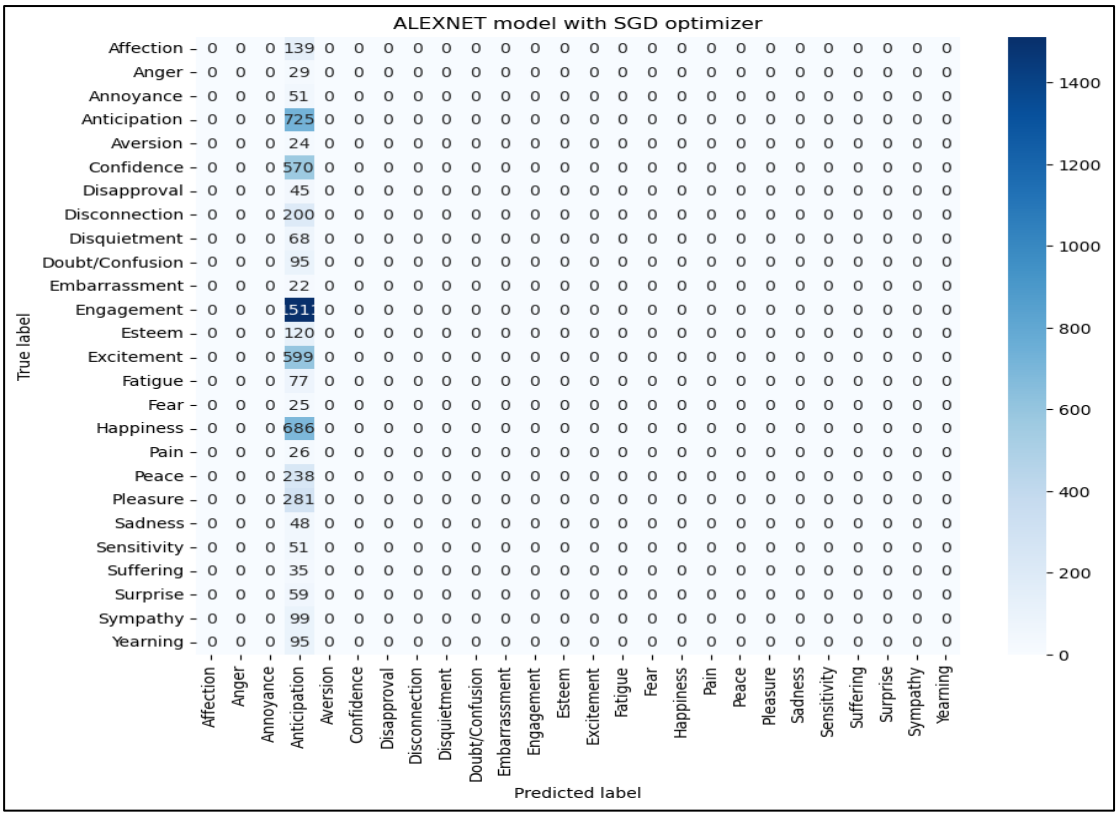

Figure 5.17 ALEXNET_TL's Confusion Matrix for SGD Optimiser

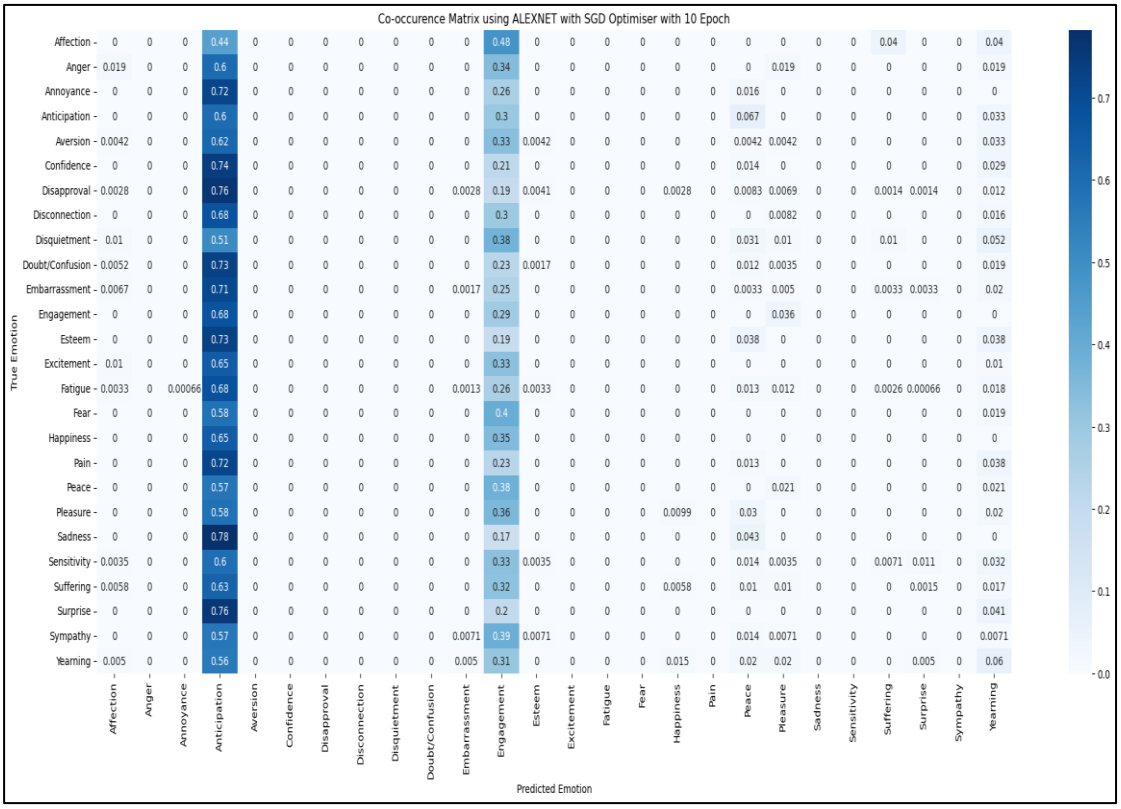

Figure 5.18 ALEXNET_TL's Co-occurence Matrix for SGD Optimiser

4. VGG19_TL

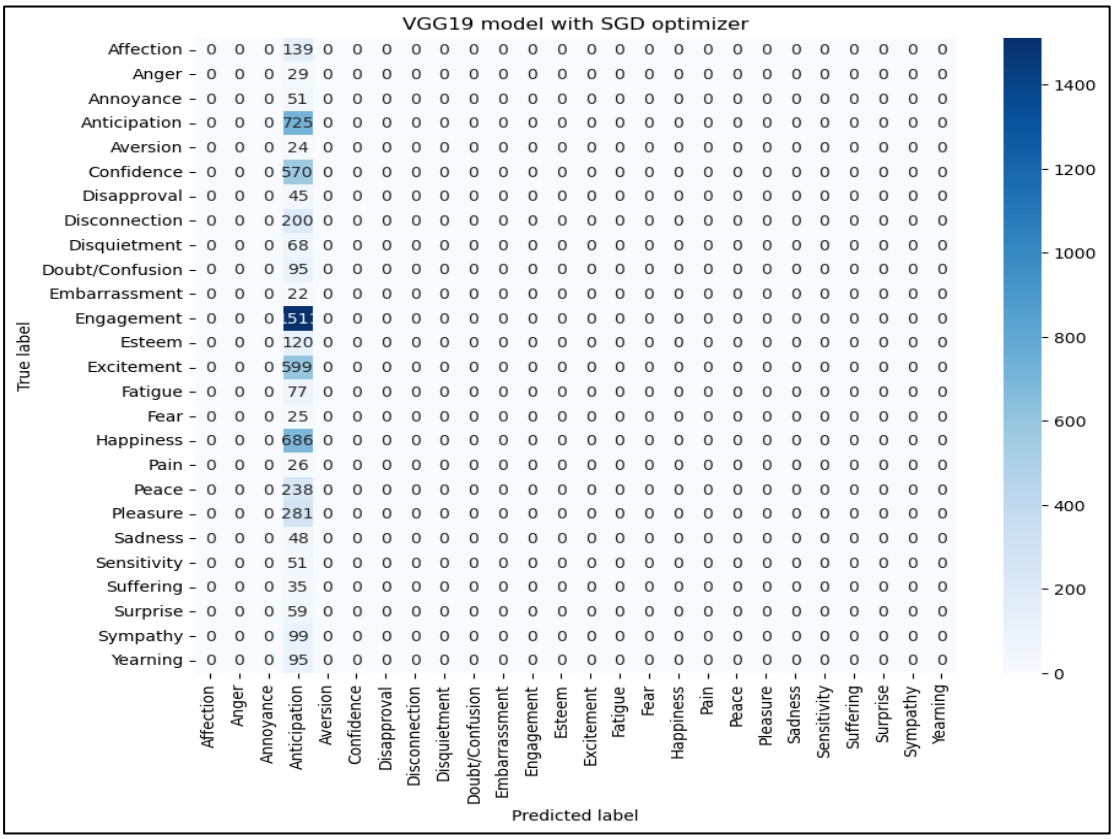

Figure 5.19 VGG19_TL's Confusion Matrix for SGD Optimiser

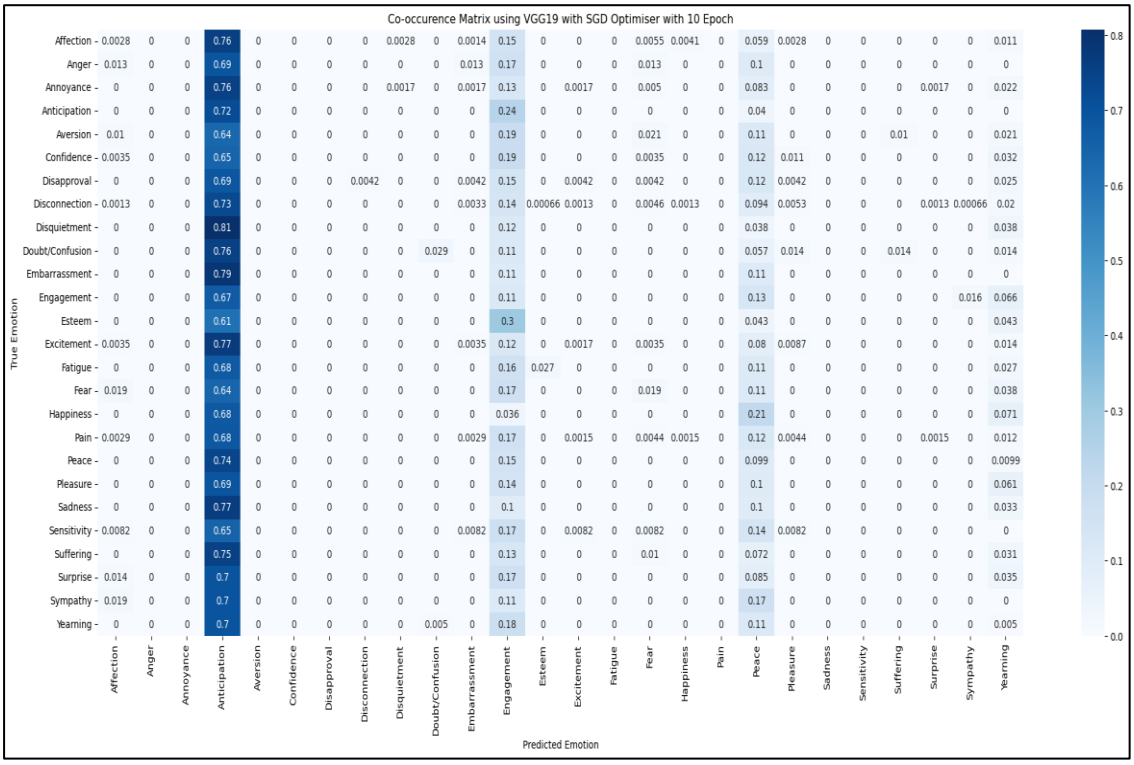

Figure 5.20 VGG19_TL's Co-occurence Matrix for SGD Optimiser

5.5.5.3 RMSprop

1. ALEXNET

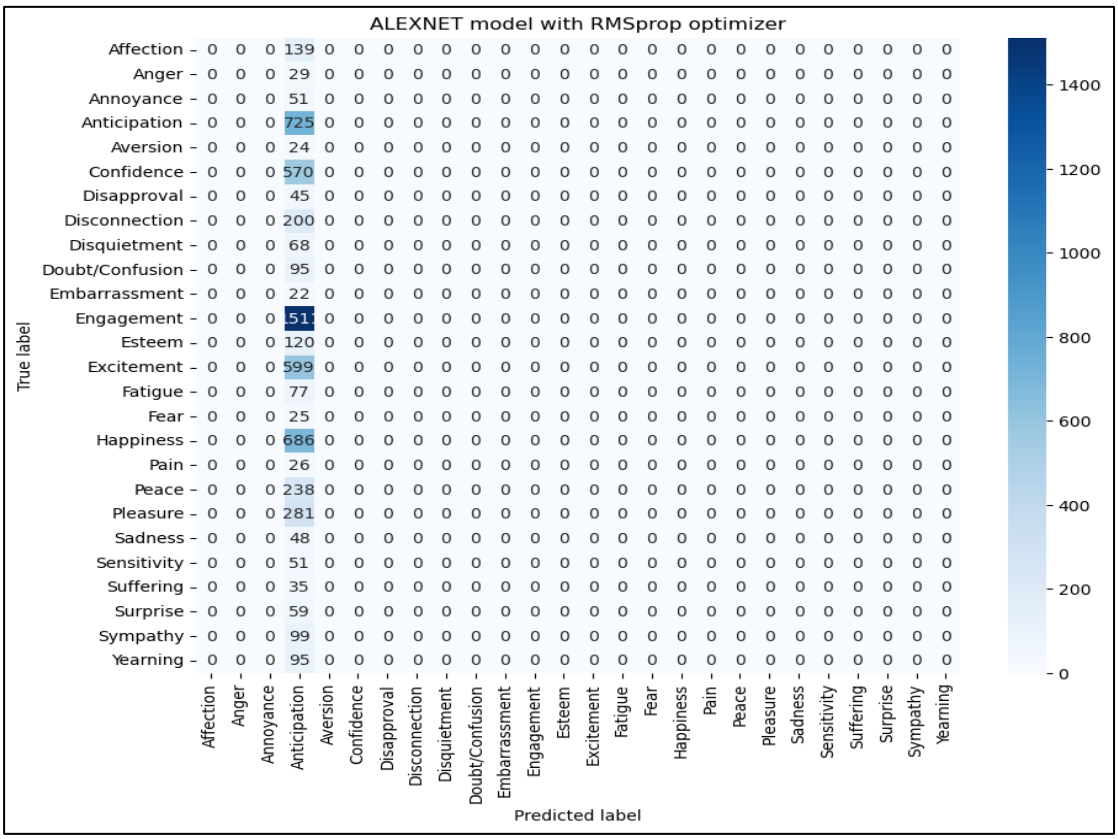

Figure 5.21 ALEXNET's Confusion Matrix for RMSprop Optimiser

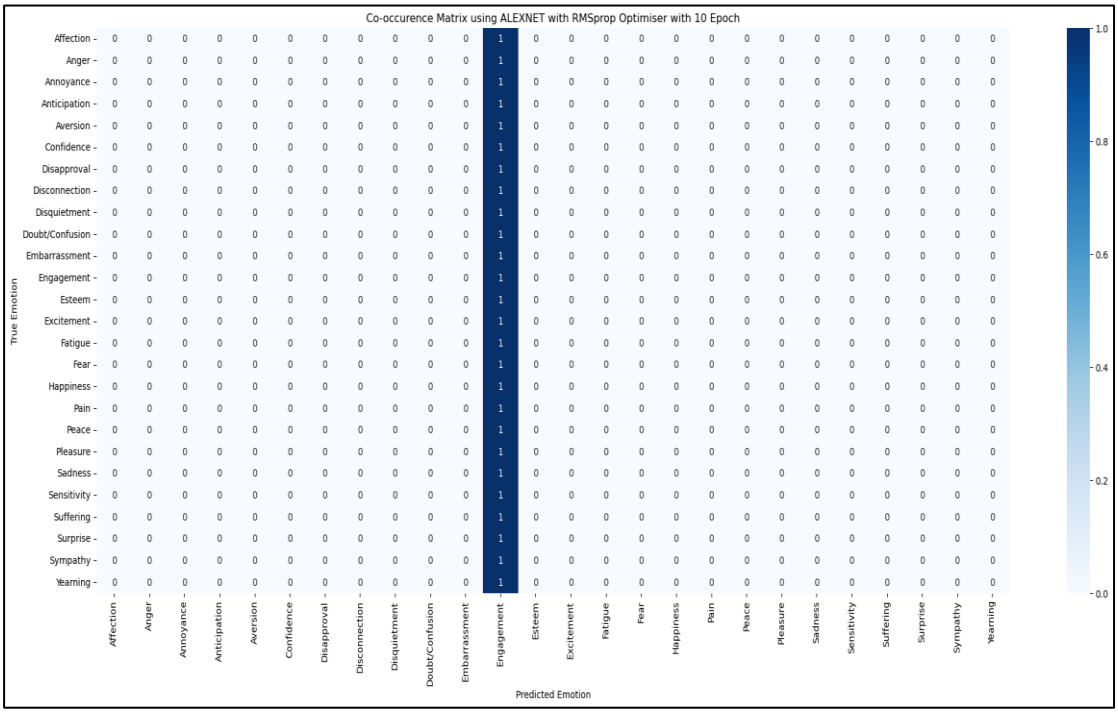

Figure 5.22 ALEXNET's Co-occurence Matrix for RMSprop Optimiser

2. VGG19

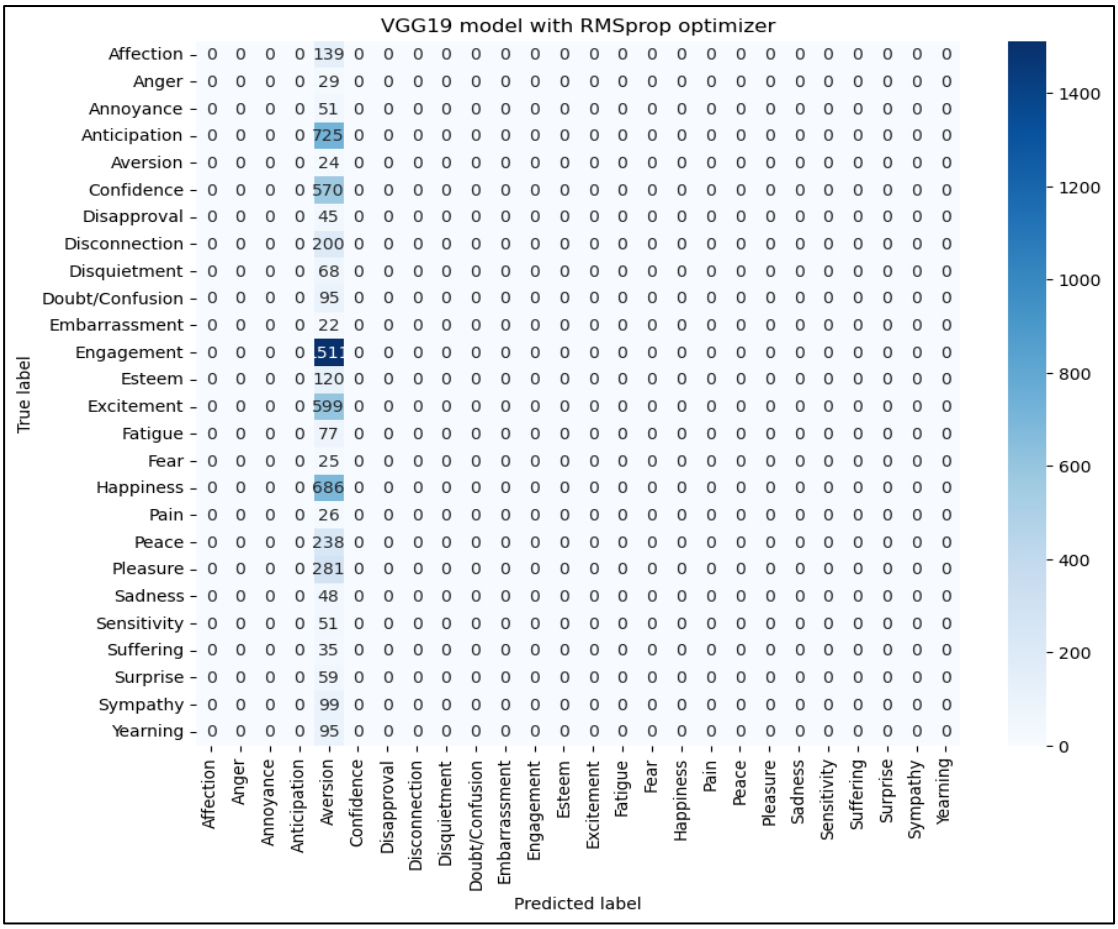

Figure 5.23 VGG19's Confusion Matrix for RMSprop Optimiser

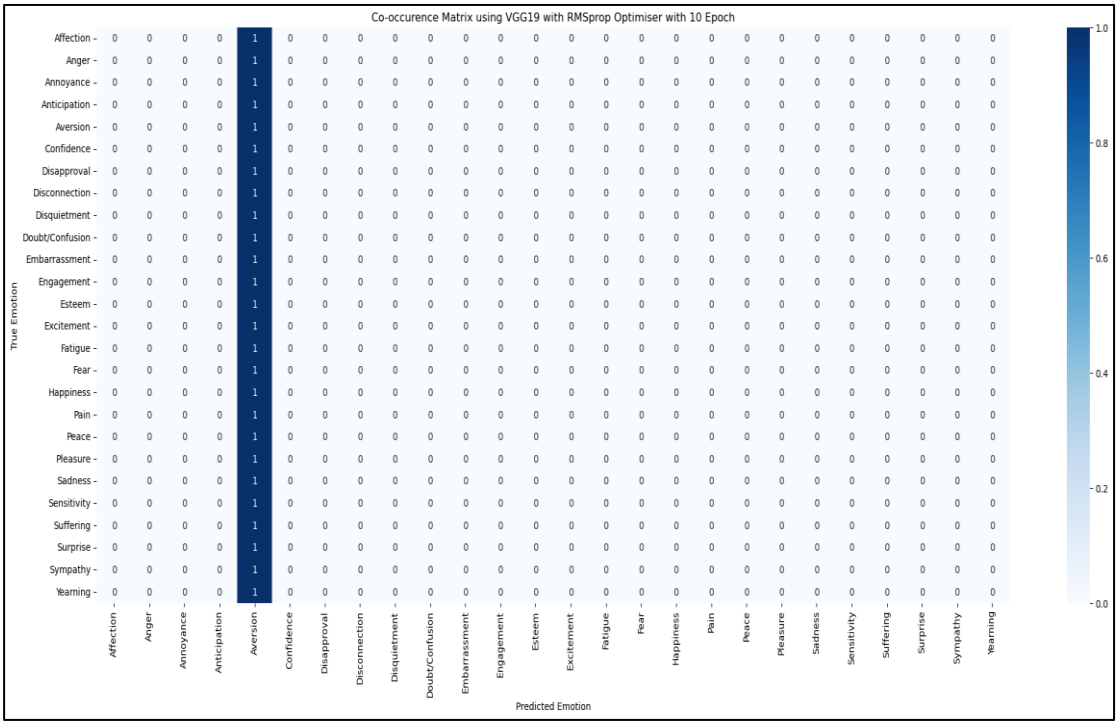

Figure 5.24 VGG19's Co-occurence Matrix for RMSprop Optimiser

3. ALEXNET_TL

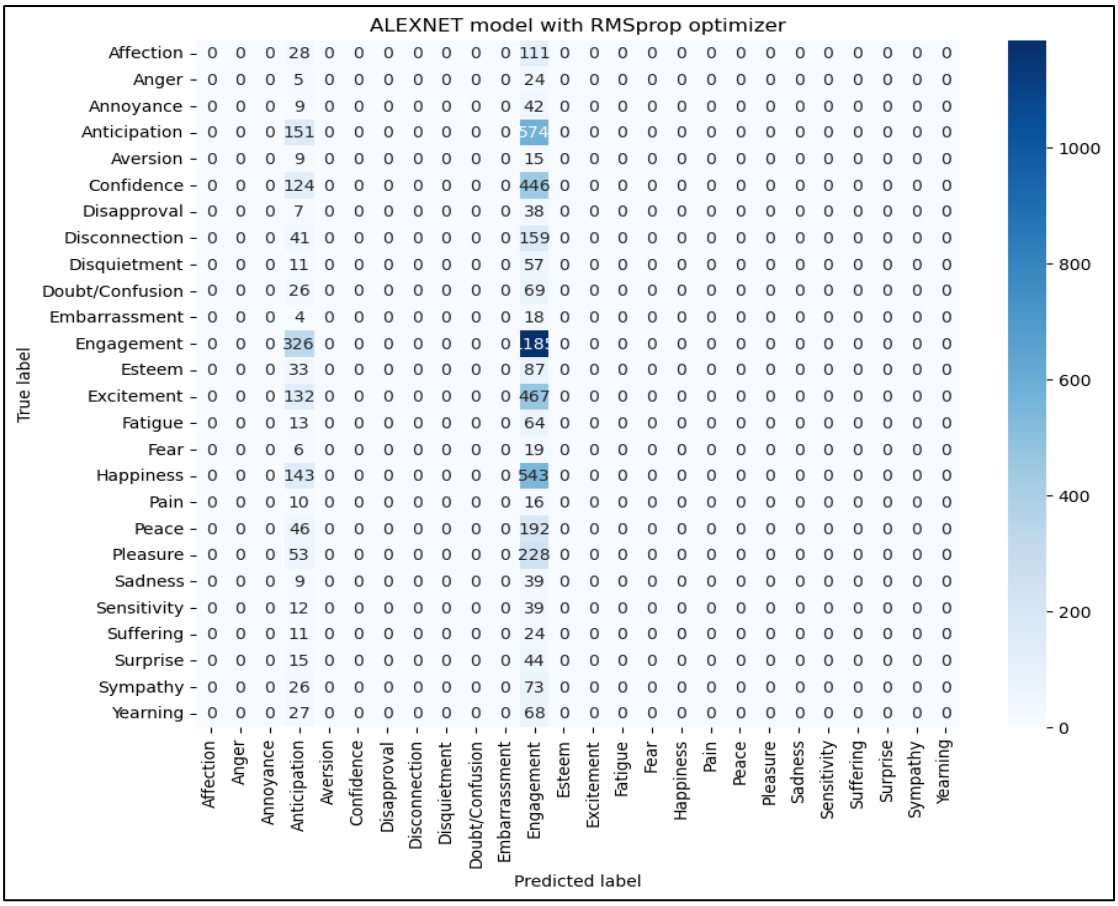

Figure 5.25 ALEXNET_TL's Confusion Matrix for RMSprop Optimiser

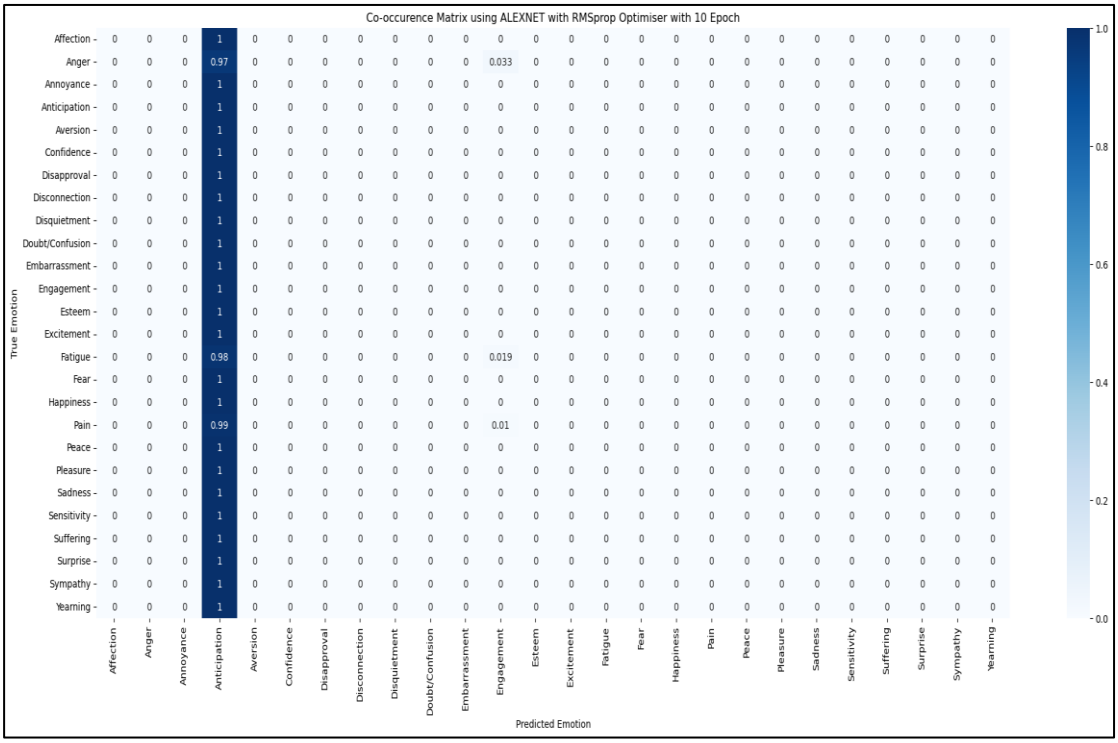

Figure 5.26 ALEXNET_TL's Co-occurence Matrix for RMSprop Optimiser

4. VGG19_TL

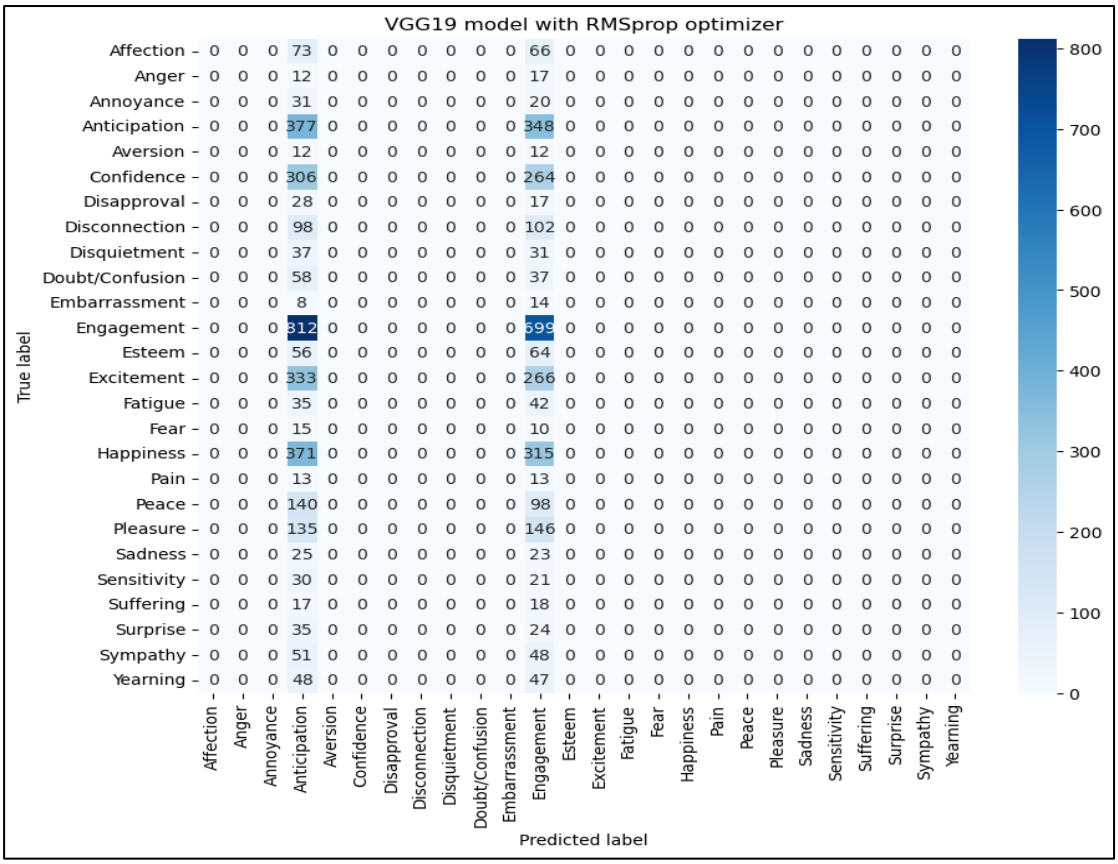

Figure 5.27 VGG19_TL's Confusion Matrix for RMSprop Optimiser

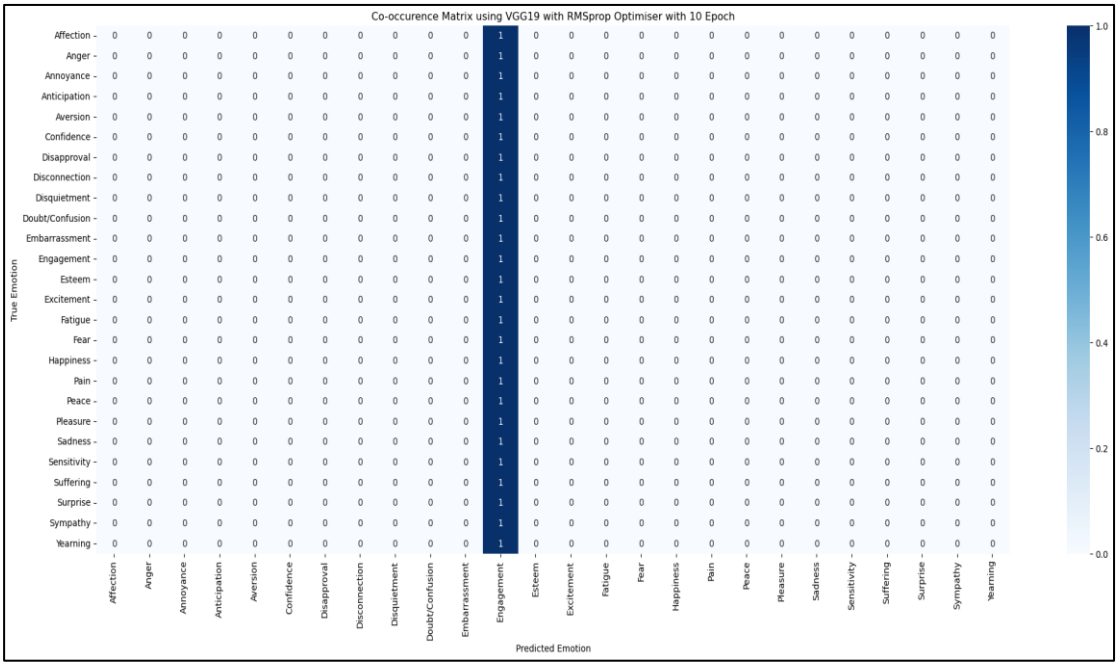

Figure 5.28 VGG19_TL's Co-occurence Matrix for RMSprop Optimiser

5.6 Flask Website Implementation

We have created Flask web application that uses OpenCV for face and eye detection, and Keras for emotion recognition. The main functionality of the application is to predict the emotions of a person from a camera feed, where the faces are detected and the emotions are recognized.

The application starts with the required imports, and Flask is initialized with the appropriate directories for templates and static files. Then, it loads the Haarcascade files for face and eye detection.

There is a function called draw_box which draws a bounding box around a detected face.

There is also a function called plot which takes emotion_label, confidence_score, and pred as input arguments, and returns a plot image with predicted emotions and their corresponding confidence scores.

There is a function called sort_index which returns the index for sorted elements.

Finally, there is a function called predict_emotion which takes an image path and a trained model as input arguments, and returns the predicted emotion label, confidence score, and prediction array.

5.6.1 Application Routes

The application has two routes:

1. The / route is the home page. It returns the Home_index.html template.
2. The /camera_feed route returns the result.html template. This template includes a JavaScript function that continuously captures images from the camera feed and sends them to the Flask server for processing.

5.6.2 Various Functions of the Application

5.6.2.1 predict()

The predict() function is responsible for predicting the emotion of a face image or a real-time video capture from the camera. It loads the trained model based on the selected option by the user, applies the prediction on the image, and displays the results on the prediction.html template. The function takes two HTTP methods, GET and POST. If the method is POST, the function handles two possible forms of user input; either an uploaded image or real-time video capture. If the method is GET, it returns the prediction.html template with the form to select the model for prediction.

The function first checks for the existence of an uploaded file and loads the selected model from the static/Models directory. If the selected option is AlexNet, it loads the AlexNet model, and if VGG19 is selected, it loads the corresponding model.

5.6.2.2 predict_emotion()

The function then applies prediction on the image using the predict_emotion() function that returns the predicted emotion label, confidence score, and the predicted class probabilities.

5.6.2.3 plot()

The plot() function then visualizes the results on a graph that shows the confidence score for each emotion. The function then draws a bounding box around the detected face region and encodes the image in base64 format to be displayed on the prediction.html template.

If the method is GET, the function simply returns the prediction.html template with the form to select the model for prediction.

5.6.2.4 presentation()

Finally, the presentation() function returns the presentation.html template to display the presentation of the project.

5.6.2.5 Camera Feed functions:

It is a Python code implementing a BaseCamera class and a CameraEvent class. The BaseCamera class is a template for camera objects that can be used to capture frames and stream them to clients. The CameraEvent class is used to signal clients when new frames are available.

The BaseCamera class has the following attributes:

1. thread: a background thread that reads frames from the camera.
2. frame: the current frame is stored here by the background thread.
3. last_access: the time of the last client access to the camera.
4. event: an instance of the CameraEvent class.

The BaseCamera class has the following methods:

1. init: initializes the object and starts the background thread if it is not running yet.
2. get_frame: returns the current camera frame.
3. frames: a generator that returns frames from the camera.
4. _thread: the camera background thread.

The CameraEvent class has the following attributes:

1. events: a dictionary that contains the events and timestamps for all active clients.

The CameraEvent class has the following methods:

1. init: initializes the object and creates an empty events dictionary.
2. wait: invoked from each client's thread to wait for the next frame.
3. set: invoked by the camera thread when a new frame is available.

4. clear: invoked from each client's thread after a frame was processed.

Overall, this code provides a basic framework for implementing a camera streaming system in Python. However, it is missing the implementation of the frames generator method in the BaseCamera class, which must be implemented by any subclass. Additionally, the code could be improved by adding error handling and improving the thread synchronization mechanism.

5.6.2.6 Camera_OpenCV

This file defines a Camera class that extends the BaseCamera class defined in the base_camera.py file. The Camera class uses OpenCV to capture video frames from a camera specified by the video_source variable.

The frames () method returns a generator that yields JPEG-encoded images, with each image containing the original video frame with a rectangle drawn around each detected face and eyes.

The haarcascade_frontalface_default.xml and haarcascade_eye.xml files are used for face and eye detection, respectively.

The set_video_source() method can be used to change the camera source at runtime by setting the video_source variable.

When the Camera object is created, it starts a background thread that reads frames from the camera and signals when a new frame is available.

The get_frame() method returns the current camera frame when called by a client. If no client accesses the camera for more than 5 seconds, the background thread stops.

5.7 Implementation Snippets

5.7.1 Base Layout

This is used for generalizing website pages and it is created in jinja template which is dynamic and compatible with python.

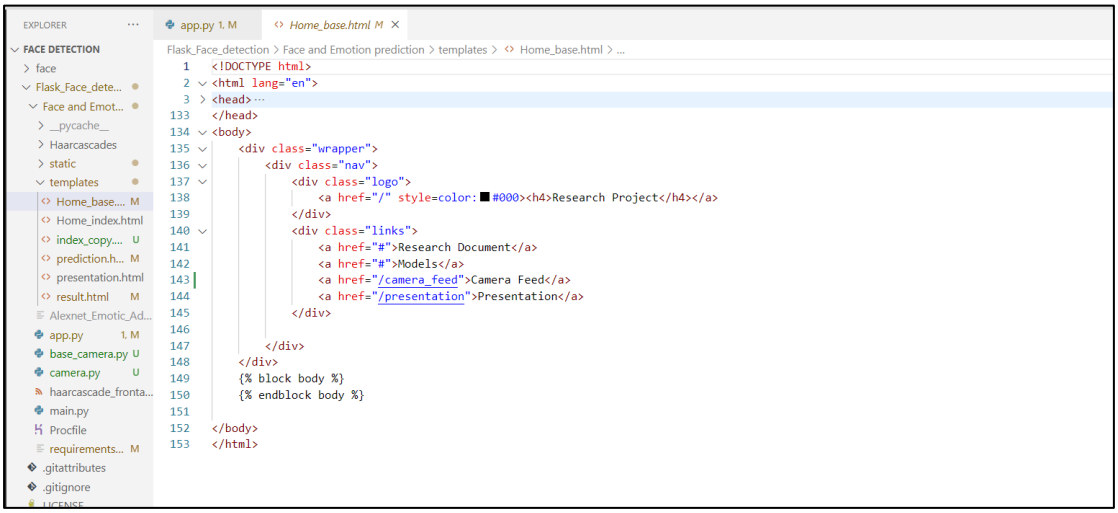

Figure 5.29 Base Layout Code

5.7.2 Home Page

```
Flask_Face_detection > Face and Emotion prediction > app.py > predict
107
108
109     ## Home Route
110     @app.route('/')
111     def index():
112         cv2.destroyAllWindows()
113         return render_template('Home_index.html')
114
```

Figure 5.30 Home Page code in app.py

```
Home_index.html | app.py 1, M | prediction.html M | Home_base.html M
Flask_Face_detection > Face and Emotion prediction > templates > Home_index.html > ...
1   {% extends 'Home_base.html' %}
2
3   {% block title %}
4   Home
5   {% endblock title %}
6
7   {% block css %}
8   <style type="text/css">···
334 </style>
335 {% endblock css %}
336
337 {% block body %}
338 <!-- LANDING PAGE -->
339
340 <div class="landing">
341     <div class="landingText" data-aos="fade-up" data-aos-duration="1000">
342         <h1>Exploring Emotion Recognition Techniques using<span style="color:#1b7cd6;font-size: 4vw"> Emotic Dataset</span> </h1>
343         <div class="btn">
344             <a href="/predict">Proceed</a>
345         </div>
346     </div>
347     <div class="landingImage" data-aos="fade-down" data-aos-duration="2000">
348         <img src="static/bg.png" alt="">
349     </div>
350 </div>
351
352 <!-- ABOUT SECTION -->
353
354 <div class="about">
355     <div class="aboutText" data-aos="fade-up" data-aos-duration="1000">
```

Figure 5.31 Home_index.html code

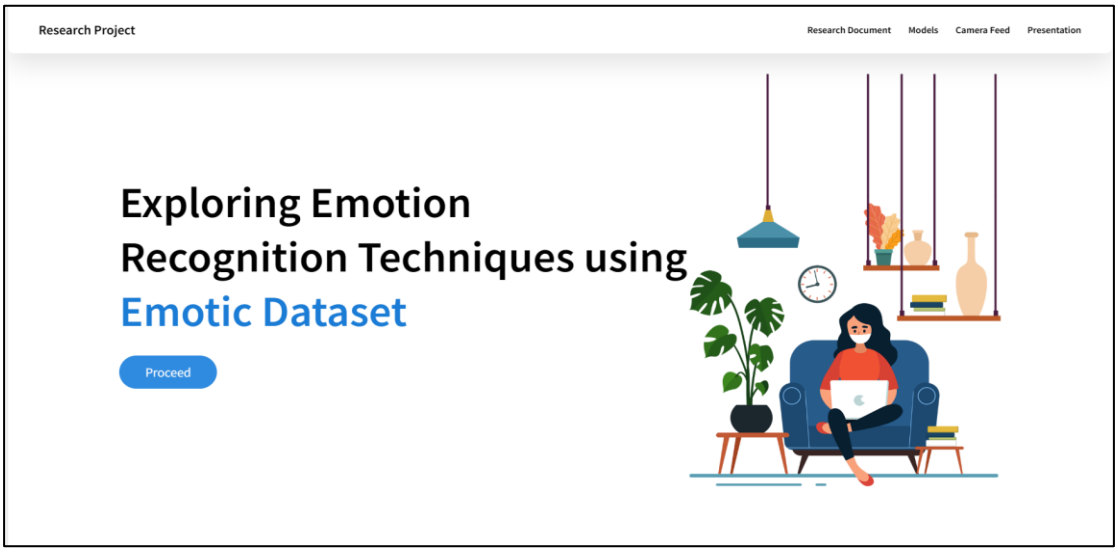

Figure 5.32 Home Page

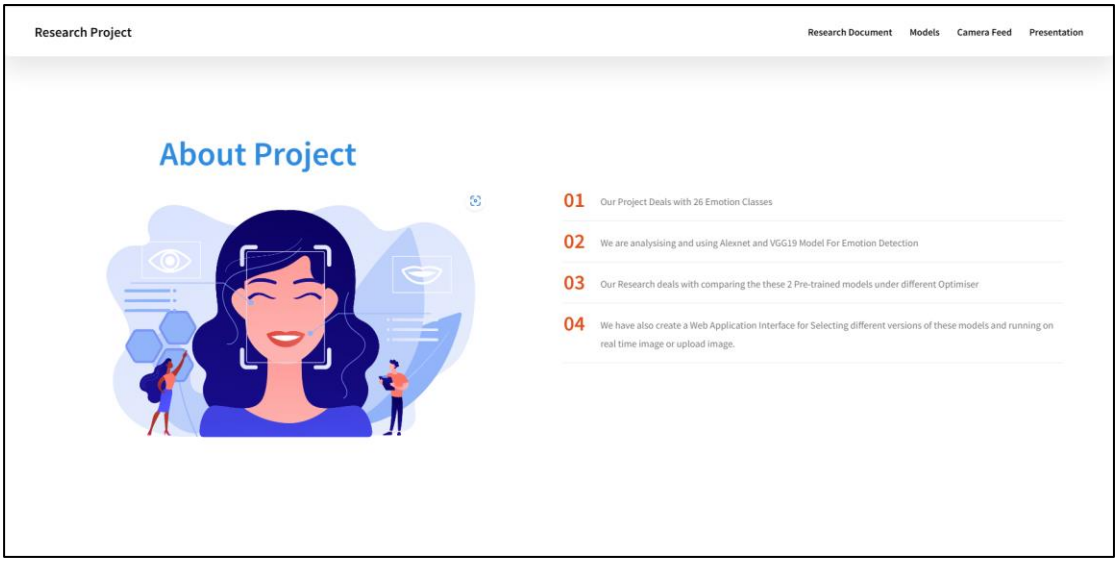

Figure 5.33 About Project Part in Home Page

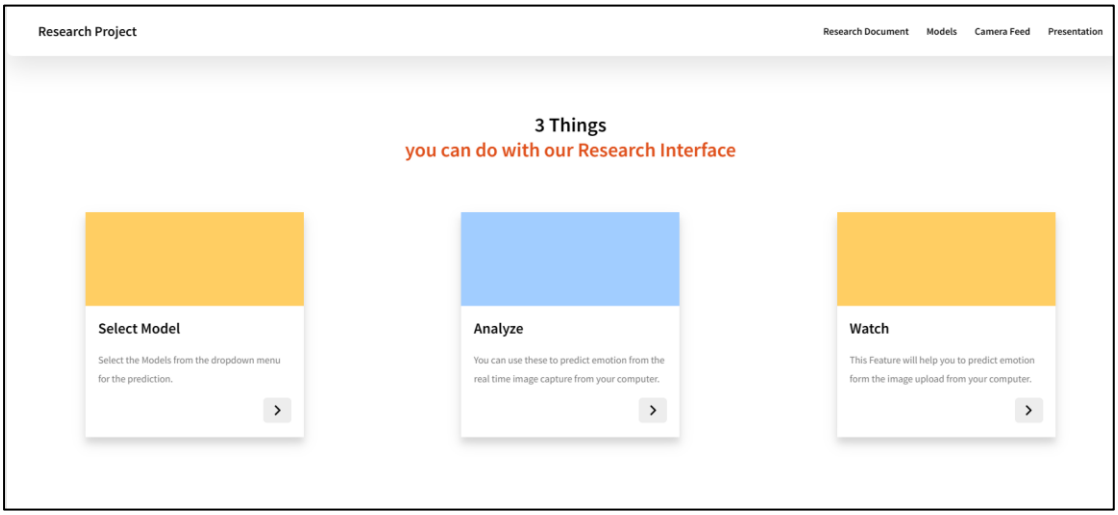

Figure 5.34 3 Functionalities of the Project

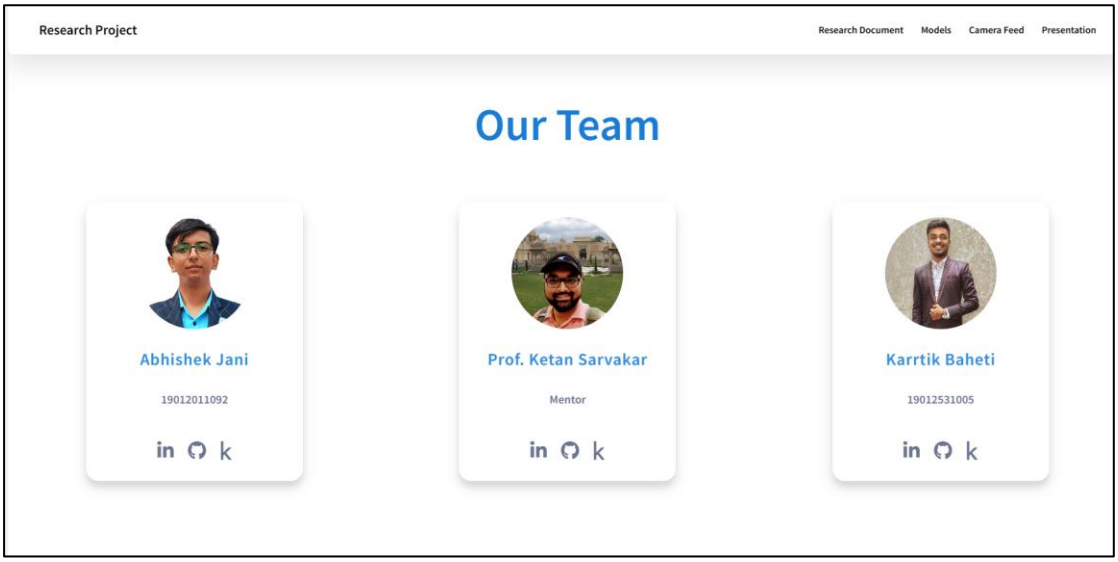

Figure 5.35 Team Members Section in Home Page

5.7.3 Camera Feed

Flask_Face_detection > Face and Emotion prediction > base_camera.py > ...

```
81
82      @staticmethod
83      def frames():
84          """Generator that returns frames from the camera."""
85          raise RuntimeError('Must be implemented by subclasses.')
86
87      @classmethod
88      def _thread(cls):
89          """Camera background thread."""
90          print('Starting camera thread.')
91          frames_iterator = cls.frames()
92          for frame in frames_iterator:
93              BaseCamera.frame = frame
94              BaseCamera.event.set()  # send signal to clients
95              time.sleep(0)
96
97              # if there hasn't been any clients asking for frames in
98              # the last 10 seconds then stop the thread
99              if time.time() - BaseCamera.last_access > 5:
100                 frames_iterator.close()
101                 print('Stopping camera thread due to inactivity.')
102                 break
103         BaseCamera.thread = None
104
```

Figure 5.36 Camera feed code -1

Flask_Face_detection > Face and Emotion prediction > app.py > predict

```
115 ## Camera Feed Route
116 @app.route('/camera_feed')
117 def camera_feed():
118     """Video streaming home page."""
119     return render_template('result.html')
120
121 def gen(camera):
122     """Video streaming generator function."""
123     yield b'--frame\r\n'
124     while True:
125         frame = camera.get_frame()
126         yield b'Content-Type: image/jpeg\r\n\r\n' + frame + b'\r\n--frame\r\n'
127
128 @app.route('/video_feed')
129 def video_feed():
130     """Video streaming route. Put this in the src attribute of an img tag."""
131     return Response(gen(Camera()),
132                     mimetype='multipart/x-mixed-replace; boundary=frame')
133
```

Figure 5.37 Camera feed code-2

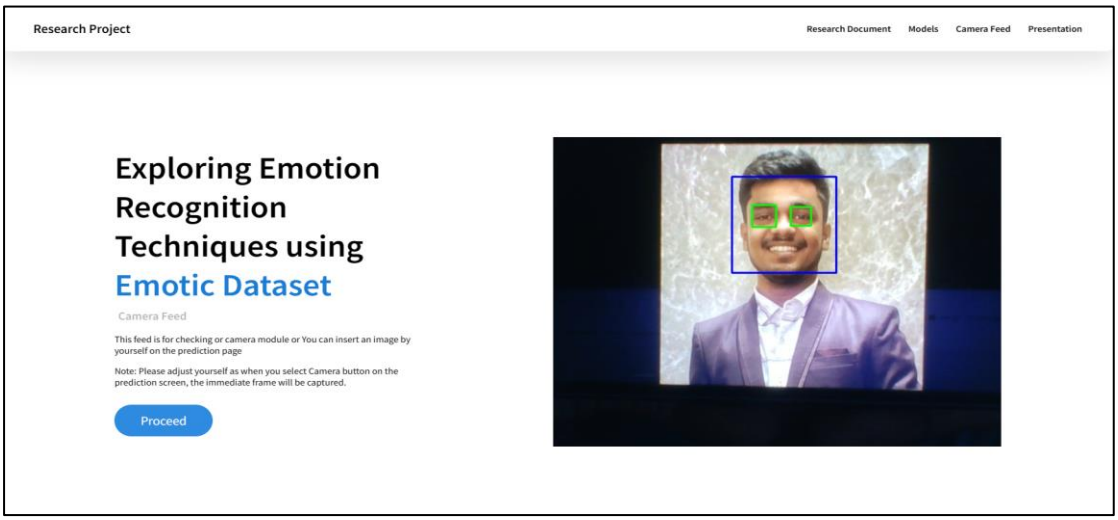

Figure 5.38 Camera Feed

5.7.4 Prediction Page

```python
## For Prediction page
@app.route("/predict", methods=['GET', 'POST'])
def predict():
    cv2.destroyAllWindows()
    model = load_model("Alexnet_Emotic_Adam.h5")
    if request.method == 'POST':
            if request.form.get('Analyze') == 'Analyze':···

            elif request.form.get('Upload') == 'Upload':···

            else:
                # pass # unknown
                return render_template("prediction.html",Models=[{'Model':'Alexnet'},{'Model':'VGG19'},{'Model':'DenseNet'}])
    elif request.method == 'GET':
            print("successful Get")

            return render_template('prediction.html',Models=[{'Model':'Alexnet'},{'Model':'VGG19'},{'Model':'DenseNet'}])
```

Figure 5.39 Prediction Page code in app.py - 1

```python
Flask_Face_detection > Face and Emotion prediction > app.py > predict_emotion
    return graph

## Returns the index for sorted elements
def sort_index(lst, rev=True):
    index = range(len(lst))
    s = sorted(index, reverse=rev, key=lambda i: lst[i])
    return s


##For Predicting Images and showing the Emotion Label
def predict_emotion(image_path,model):
    # Load the image and preprocess it
    # model = load_model("Alexnet_Emotic_Adam.h5")
    top3=[]
    emotion_label=[]
    # img = load_img(image_path, target_size=(224, 224))

    img = img_to_array(image_path)
    img = np.expand_dims(img, axis=0)
    img = tf.keras.applications.resnet.preprocess_input(img)

    # Make a prediction using the loaded model
    pred = model.predict(img)

    # Get the predicted emotion label and confidence score
    Pred = pred[0]
    top3.extend(sort_index(pred[0])[:3])
    # print(top3)
    for i in range(3):
        emotion_label.append(emotion_labels[top3[i]])
    confidence_score = pred[0][top3[0]]
    return emotion_label, confidence_score, Pred
```

Figure 5.40 Prediction Page code in app.py - 2

```python
Flask_Face_detection > Face and Emotion prediction > app.py > plot
            'Sympathy', 'Yearning']

# Plot Function which returns the Plot Image
def plot(emotion_label, confidence_score , pred):
    # Define a color map for the emotions
    cmap = plt.get_cmap('tab20b')

    # Predict the emotion label of the image
    # emotion_label, confidence_score , pred = predict_emotion(image_path)

    # Plot a bar graph with the predicted emotions and their corresponding confidence scores
    plt.tight_layout()
    fig, ax = plt.subplots(figsize=(28, 8))
    bars = ax.bar(emotion_labels, pred, color=cmap(range(len(emotion_labels))))
    plt.xticks(rotation=90)
    plt.xlabel('Emotion')
    plt.ylabel('Confidence Score')
    plt.title(f'Predicted Emotion: {emotion_label} (Confidence: {confidence_score:.2f})')

    # Add a legend to the bar graph
    handles = [mpatches.Patch(color=cmap(i), label=emotion_labels[i]) for i in range(len(emotion_labels))]
    ax.legend(handles=handles, bbox_to_anchor=(1.05, 1), loc='upper left')
    buf=io.BytesIO()
    plt.savefig(buf,format='png')
    buf.seek(0)
    graph = base64.b64encode(buf.read()).decode('utf-8')

    return graph
```

Figure 5.41 Prediction Page code in app.py

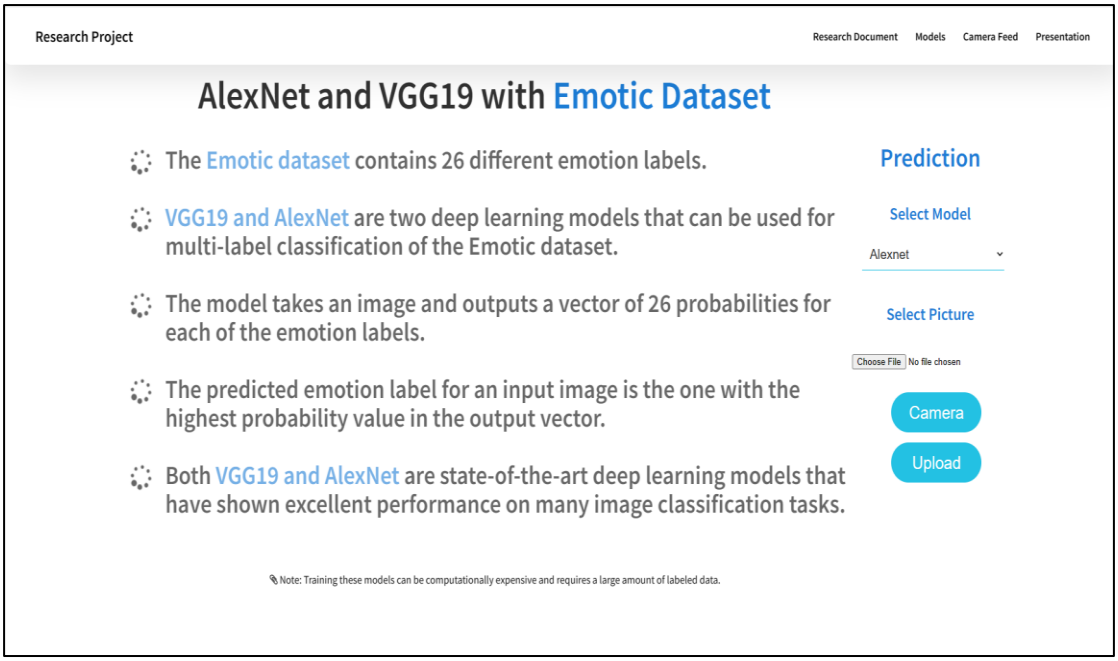

Figure 5.42 Prediction Page

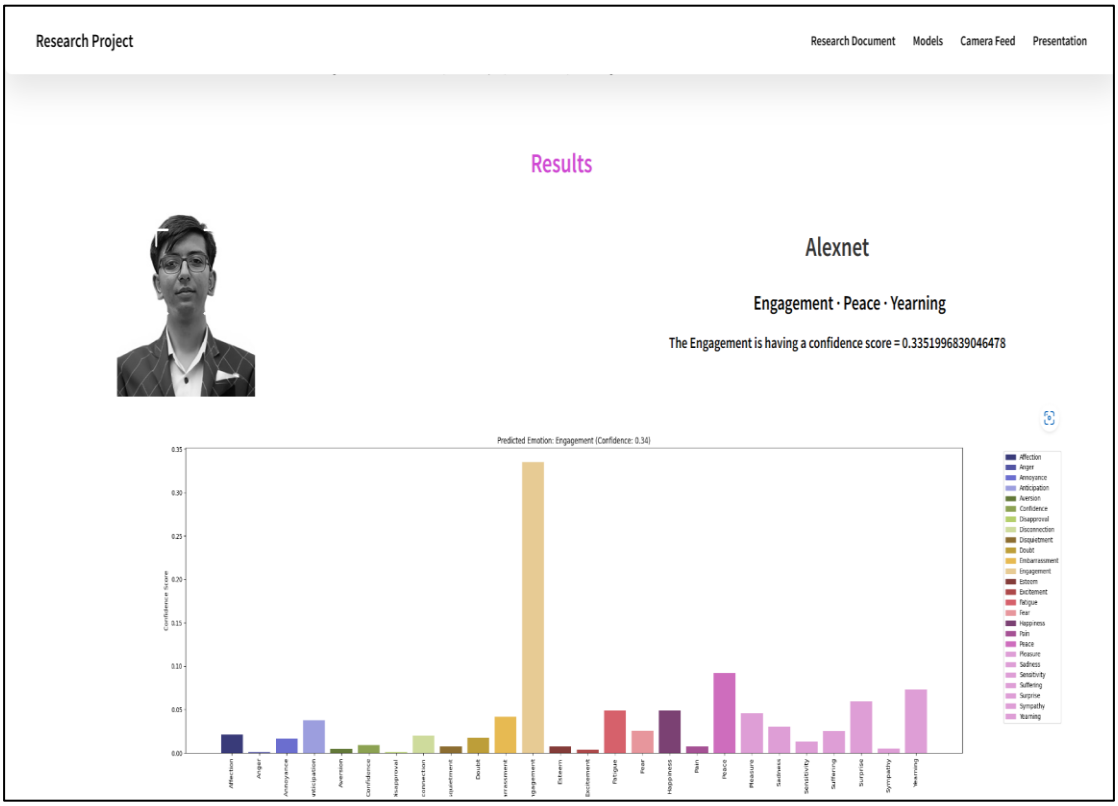

Figure 5.43 Prediction Page Result

5.7.5 Presentation Page

Flask_Face_detection > Face and Emotion prediction > templates > presentation.html > center

```
{% extends 'Home_base.html' %}

{% block title %}
Presentation
{% endblock title %}
{% block css %}
{% endblock css%}

{% block body %}
<br>
<br>
<br>
<br>
<br>
<center>
    <iframe
        src="https://onedrive.live.com/embed?resid=3BB26557EDD297D2%21154&amp;authkey=%21AMHN2cGUzzncFTQ&amp;em=2&amp;wdAr=1.7777777777777777"
        width="1186px" height="691px" frameborder="0">This is an embedded <a target="_blank"
            href="https://office.com">Microsoft Office</a> presentation, powered by <a target="_blank"
            href="https://office.com/webapps">Office</a>.</iframe>
</center>
{% endblock body %}
```

Figure 5.44 Presentation Page HTML code

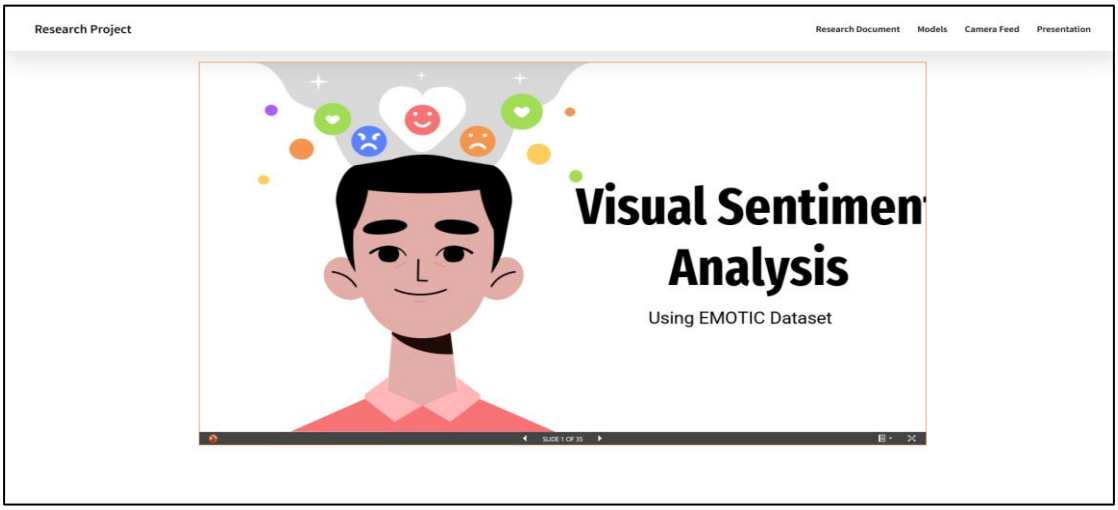

Figure 5.45 Presentation Page

5.8 Project Timeline

A Gantt chart is a type of bar chart that illustrates a project schedule. It is a visual representation of the timeline of a project, showing the start and end dates of each task or activity in the project. A Gantt chart typically lists all the tasks or activities on the vertical axis and the timeline on the horizontal axis. Each task or activity is represented by a horizontal bar, which shows its start date, end date, and duration. The length of the bar represents the amount of time required to complete the task, and bars may be color coded to indicate the status of each task, such as whether it is on track, behind schedule, or completed. Gantt charts can help project managers to plan and track the progress of a project, allocate resources effectively, and identify potential issues or delays. They are commonly used in project management software and can be updated in real-time as tasks are completed or delayed, allowing project managers to adjust the project schedule accordingly.

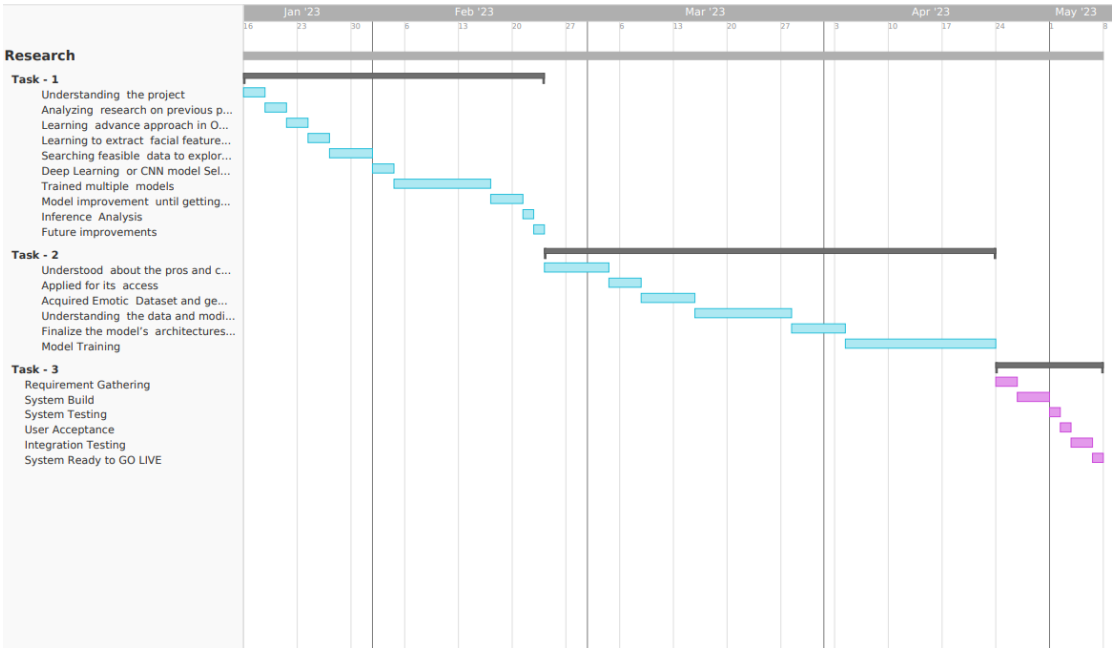

Figure 5.46 GANTT chart

Table 7 Legend

Task - 1
Understanding the project
Analyzing research on previous projects
Learning advance approach in OpenCV
Learning to extract facial features from an image
Searching feasible data to explore Emotion Detection
Deep Learning or CNN model Selection
Trained multiple models
Model improvement until getting best parameters
Inference Analysis
Future improvements
Task - 2
Understood about the pros and cons of Emotic Dataset
Applied for its access
Acquired Emotic Dataset and generated its annotations file
Understanding the data and modifying the parameters
Finalize the modelâ€™s architectures
Model Training
Task - 3
Requirement Gathering
System Build
System Testing
User Acceptance
Integration Testing
System Ready to GO LIVE

6 SYSTEM REQUIREMENT STUDY

6.1 Functional Requirement

6.1.1 Home Page

Table 8 Home Page Requirements

Term	Description
Input	In the home page of our Research Website, User can access: 1. Presentation 2. Documentation 3. Models 4. Predictions 5. Camera Feed
Output	1. Presentation i) When you click on the Presentation present on the navigation bar of the home page, you will be redirected to the presentation page of the website. ii) Microsoft PowerPoint Presentation of the project will be visible to you on this page. iii) You can change the slide and also able to enlarge the presentation. 2. Documentation i) When you click on the 'Documentation' present on the navigation bar of the home page, you will be able to see the Project Report. 3. Models i) When you click on 'Models' present on the navigation bar of the home page, you will see the Models Statistics of all the models which we trained. 4. Predictions i) When you click on the "Proceed" button you will send to the emotion detection page of the website where you will able to predict via Camera or via uploading image from existing system. 5. Camera Feed i) When you click on the 'Camera Feed' present on the navigation bar of the home page, you will see the live camera feed of yourself from your camera with your eyes, nose being detected.

6.1.2 Prediction

Table 9 Prediction Page Requirements

Term	Description
Input	In the Prediction Page, you will be able to detect emotion of image whichever you want. 1. First User need to select the model through which he/she want to detect emotion. 2. Model can be selected from the drag down list of models available to the user. 3. After Selecting the necessary model, user has two options: a) Camera: Go for Emotion Detection using Real time Camera Feed b) Upload: Selecting Image and predicting emotion for the image existing in your system.
Output	After selecting model from the dropdown, if user has chosen: 1. Real Time Camera Feed Image: a) The First frame from your camera device will be used to detect emotion, if the image captured properly with Face visible clearly a face detection mark will be displayed on the image. b) The output of this will the user's real time Image, Top 3 Emotion Predicted by the model, Confidence of the top emotion will display and bar graph of all the emotion that were detected by the model will be displayed. 2. Using Upload a) The output of this will the user's uploaded Image, Top 3 Emotion Predicted by the model, Confidence of the top emotion will display and bar graph of all the emotion that were detected by the model will be displayed.

6.2 Non-Functional Requirements

6.2.1 Software Flexibility

The main attribute of the system is its ability to enable flexible access to information and resources. Flexible access to resources means this project is going to be used on any desktop or laptop devices with some basic requirements e.g., webcam or camera.

6.2.2 Scalability Requirements

The system's scalability is one of its most important features. The system must be able to add any additional functionality even after the project is developed once. This allows for easy expansion and modification as the needs of the user base change.

6.2.3 Usability Requirements

The application is user-friendly and provides ease of access for users. All the user need is a camera or image in a system. The user interface is designed in a way that makes it easy to navigate and find the desired functionality. All the features and tools are easily accessible and can be used with little to no training.

7 TESTING

Table 10 Testing

Sr. No	Description	Input	Expected Value	Actual Value	Result
1	To Open Presentation	Click on "Presentation" on navigation bar.	User will move from home page to Presentation page and PPT of will be displayed	Moved to Presentation Page and PPT of the Project is displayed.	PASS
2	To Open Documentation	Click on "Documentation" on navigation bar.	User will move from home page to Documentation page and Project Report will be displayed	Moved from home page to Documentation page and Project Report is displayed	PASS
3	To Open Model Statistics	Click on "Models" on navigation bar.	User will move from home page to Models page and Models Stats. will be displayed	Moved from home page to Models Page and Models Stats were displayed	PASS
4	To open Camera Feed	Click on "Camera Feed" option on navigation bar	User will be redirected to camera feed page.	Moved from home page to Camera feed page.	PASS
5	To do Emotion Prediction	Click on proceed button on home page or the proceed button on camera feed page	User will be redirected to Prediction Page	Redirected to Prediction Page	PASS
6	To Select Model	Models can be selected from the drop-down menu available on Prediction Page	User will be selecting the model from the drop-down menu	Able to select model	PASS
7	To Capture Image for real time Emotion Detection	Click on capture button on prediction page	User's first frame from its camera after clicking button will take for prediction	User's Image was captured	PASS
8	To Upload Existing Image for Emotion Detection	Click on Upload button on prediction page	Upload window will be appeared from which user can choose Image	Image was uploaded	PASS
9	Predict Emotion	Click on predict button	Top 3 Emotion Confidence Score and Emotion's Bar graph.	Emotion was predicted and Graph was displayed	PASS

8 SYSTEM DESIGN

8.1 Data Flow Diagram

8.1.1 DFD Level 0

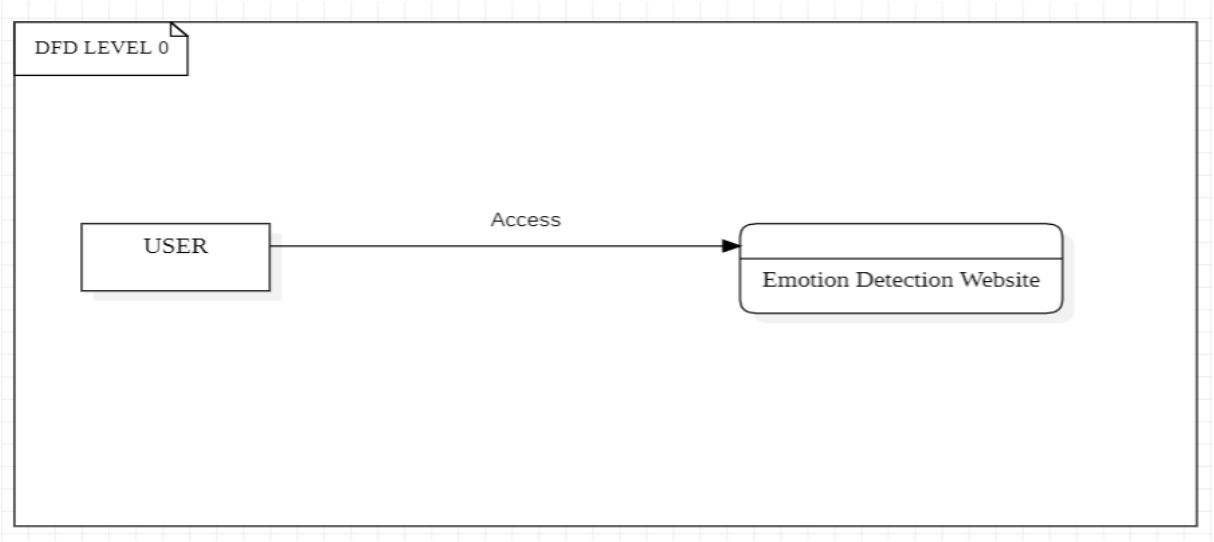

Figure 8.1 DFD Level 0

8.1.2 DFD Level 1

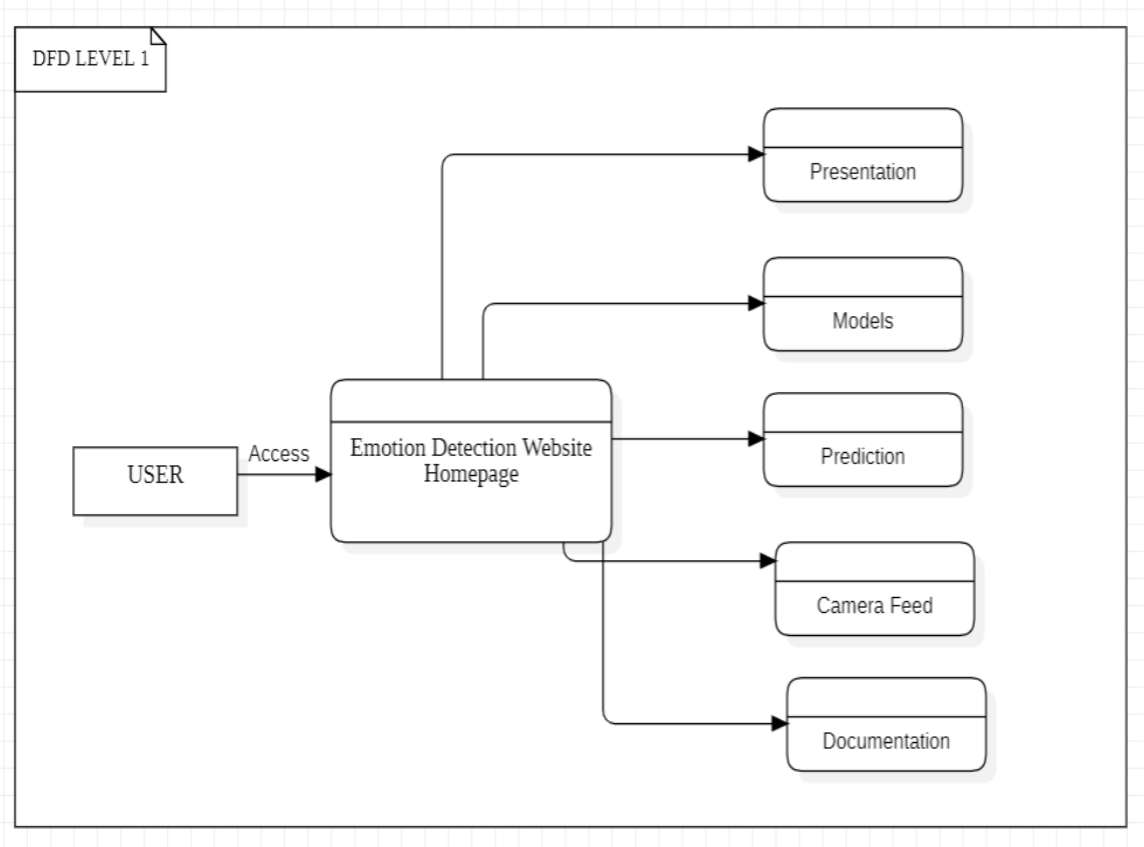

Figure 8.2 DFD Level 2

8.1.3 DFD Level 2

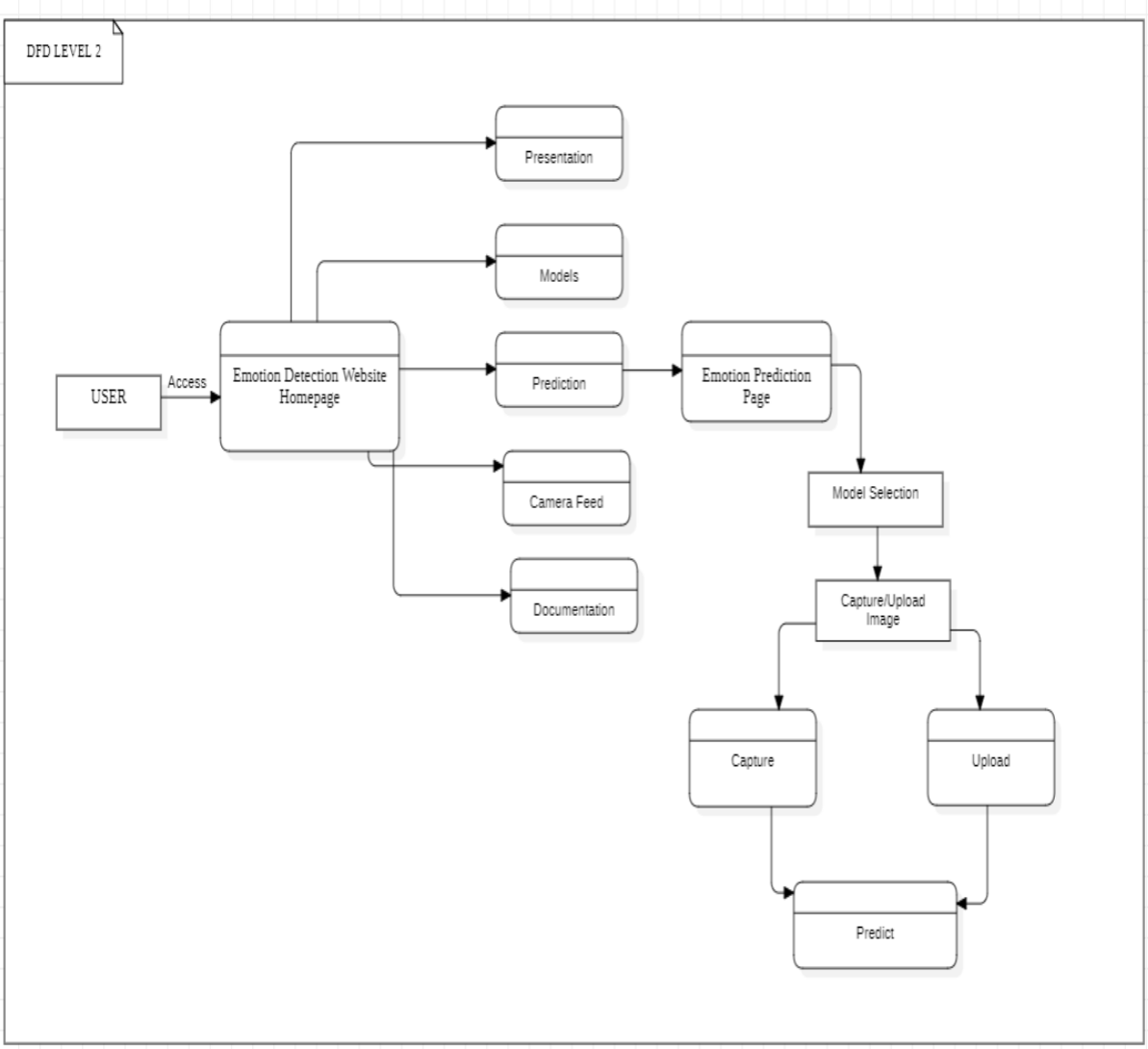

Figure 8.3 DFD Level 2

8.2 Class Diagram

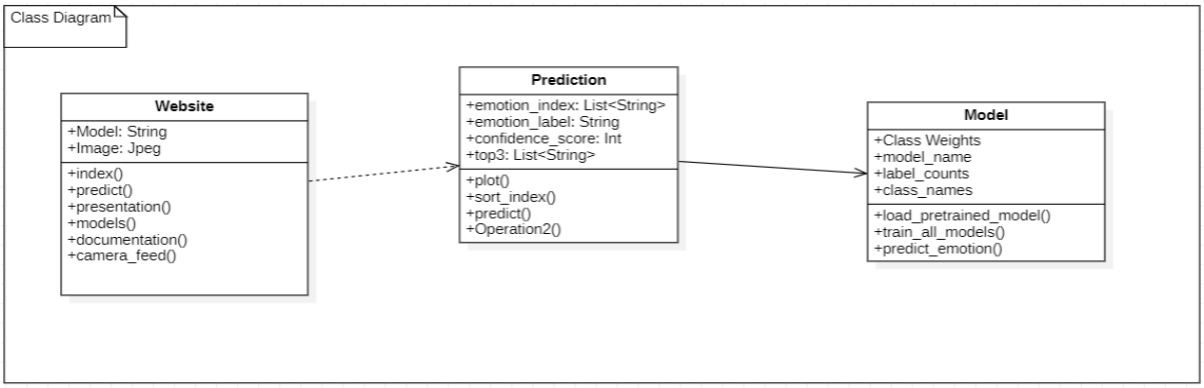

Figure 8.4 Class Diagram

8.3 Use Case Diagram

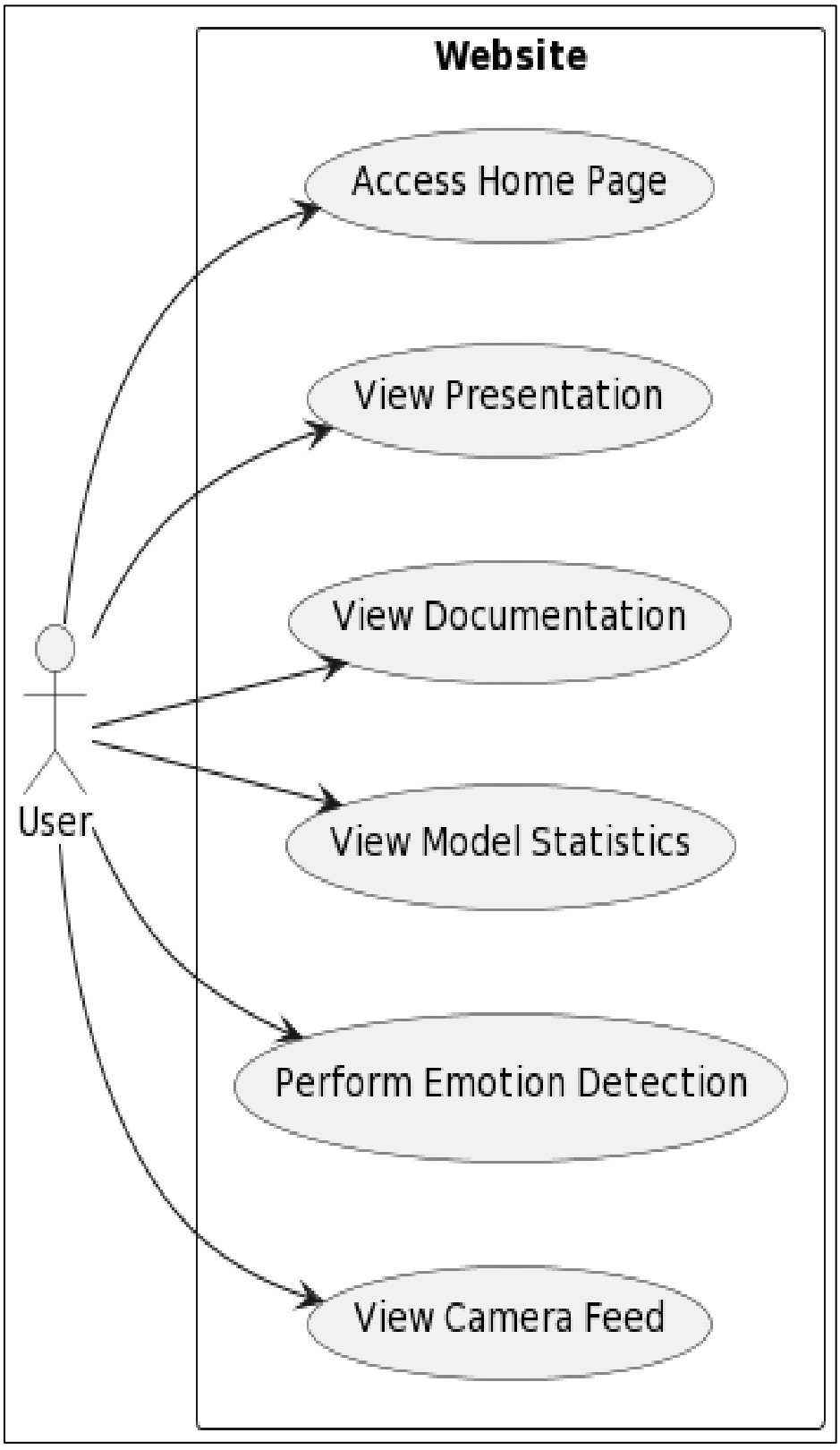

Figure 8.5 Use Case Diagram

8.4 Sequence Diagram

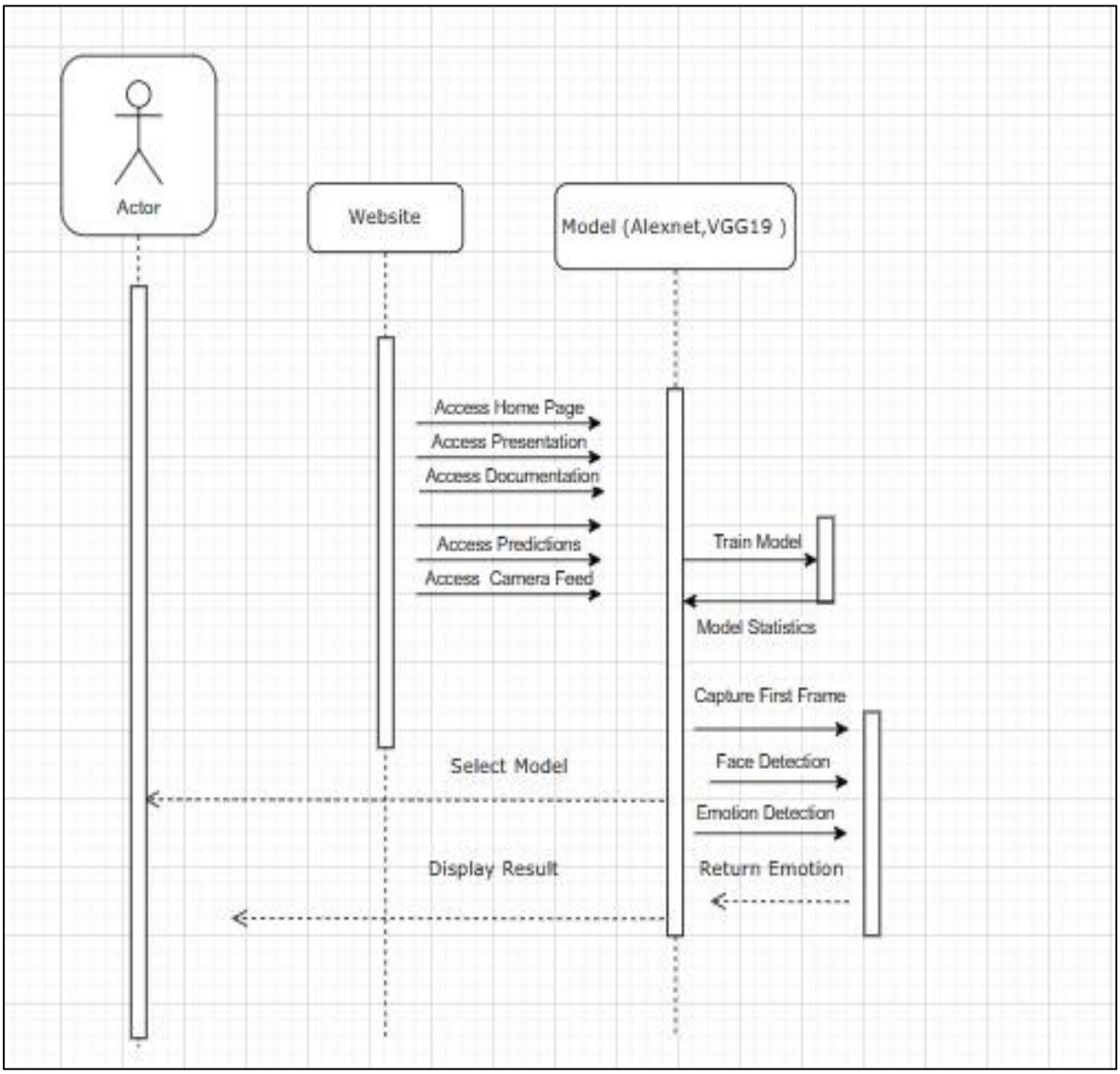

Figure 8.6 Sequence Diagram

8.5 Activity Diagram

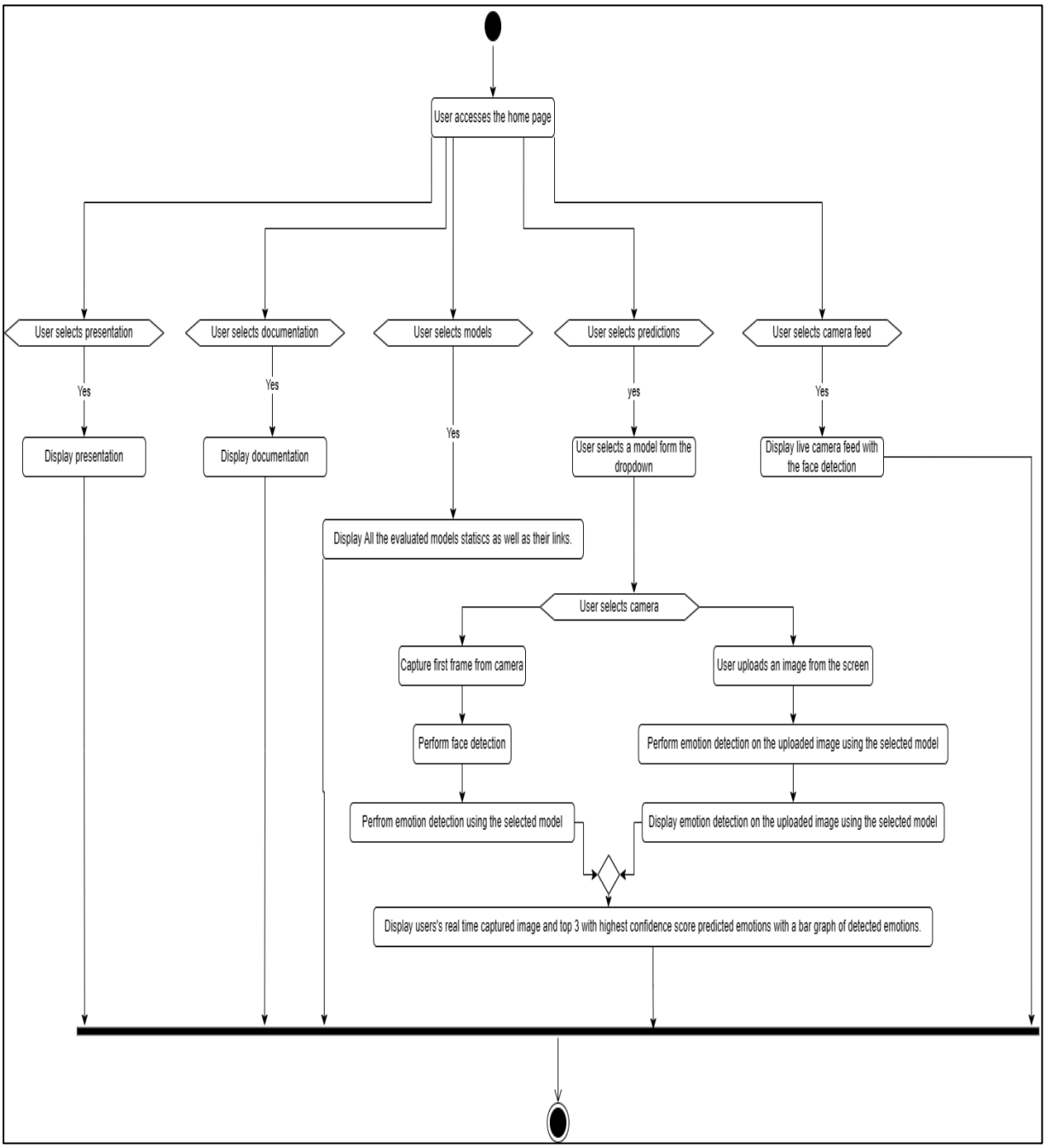

Figure 8.7 Activity Diagram

9 USER MANUAL

1. Home Page of the Website

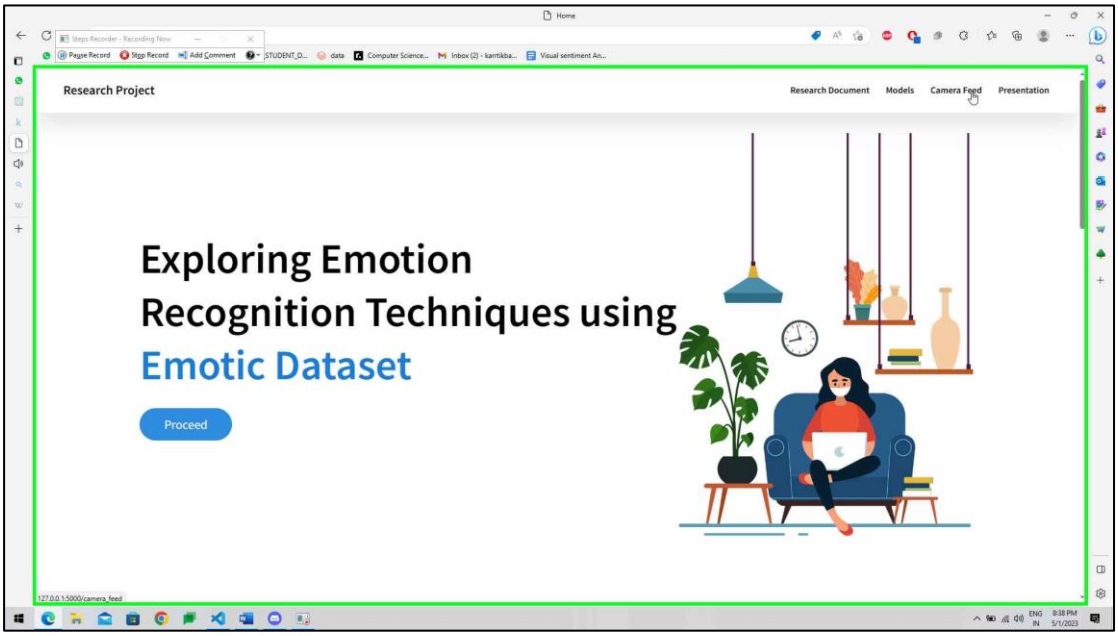

Figure 9.1 User using Homepage

2. User using Camera Feed

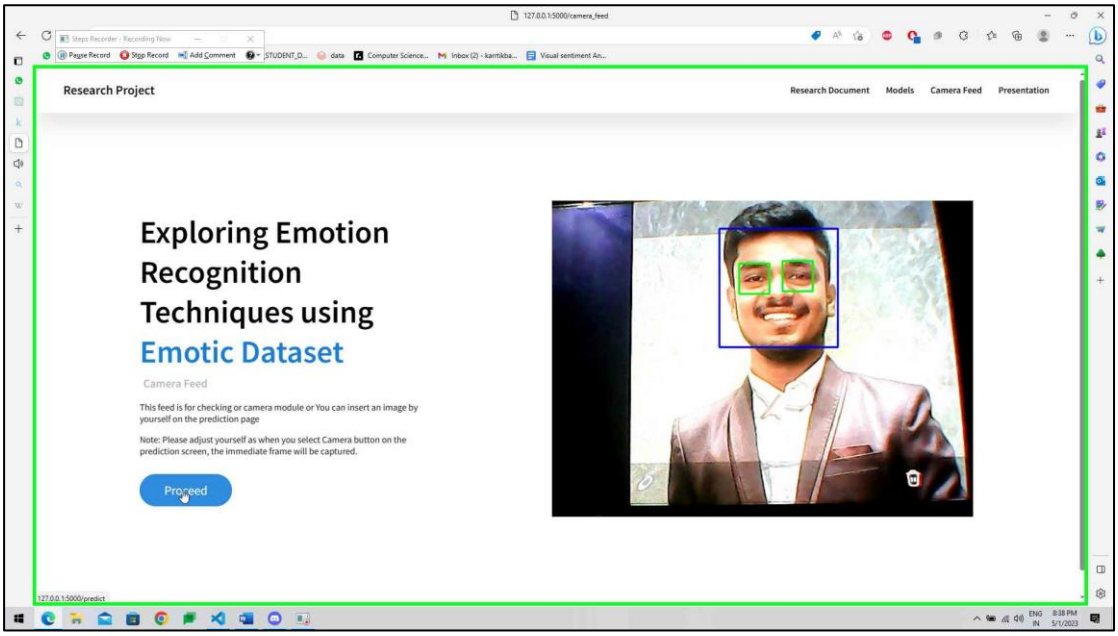

Figure 9.2 User using Camera Feed page

3. User using Prediction Page after clicking proceed button on home page or in camera feed

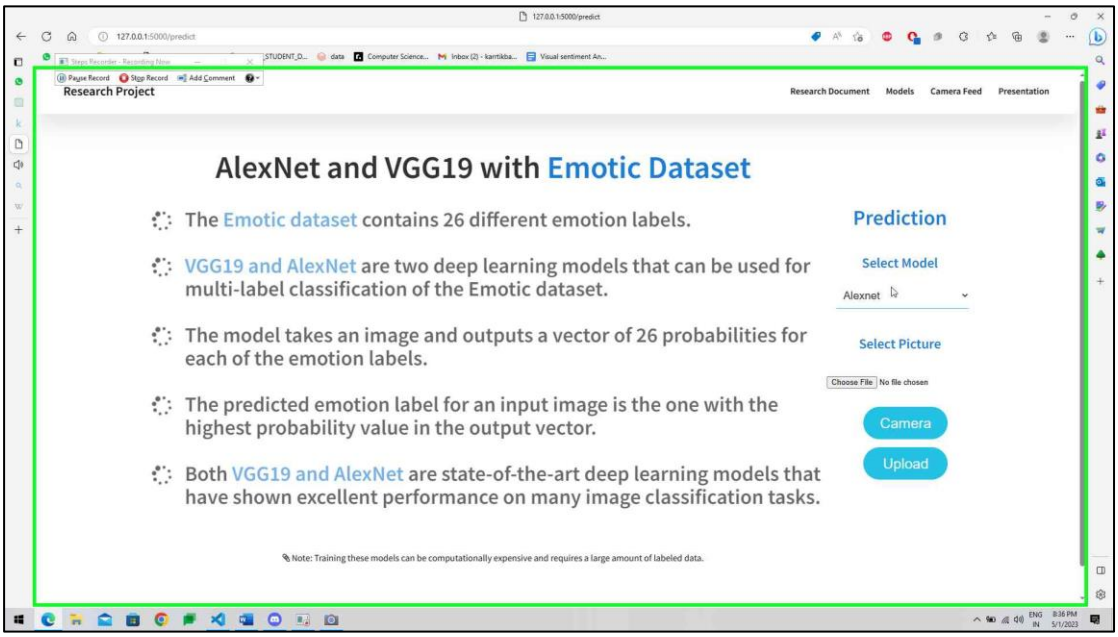

Figure 9.3 User using Prediction page

4. User selecting model from Drop-Down and using Camera mode

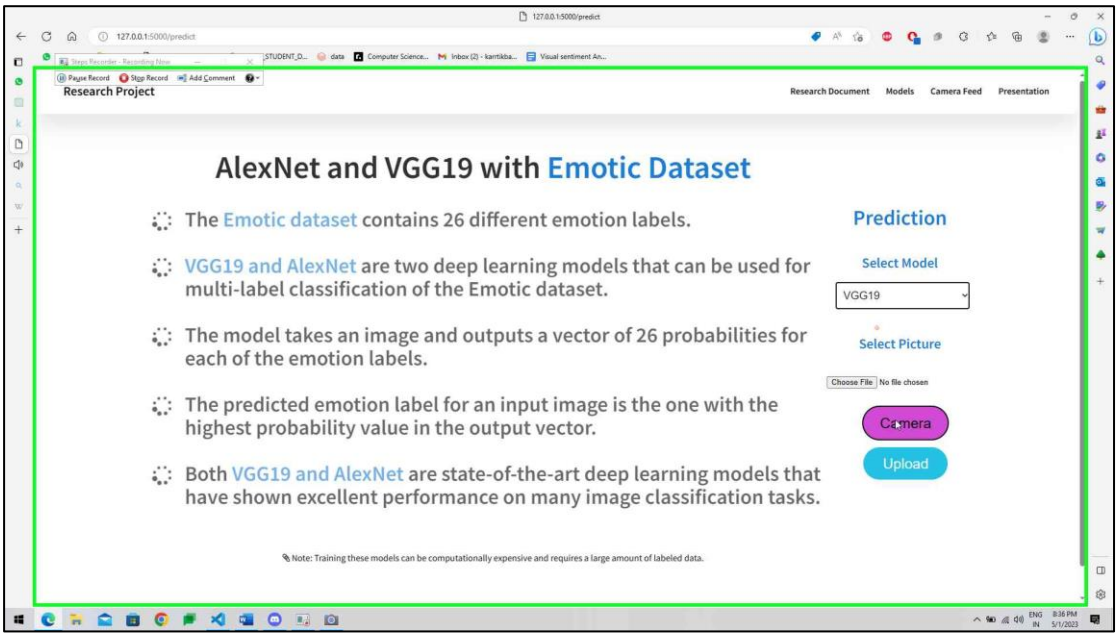

Figure 9.4 User selecting model and using Camera Mode

5. User using Upload Feature for prediction

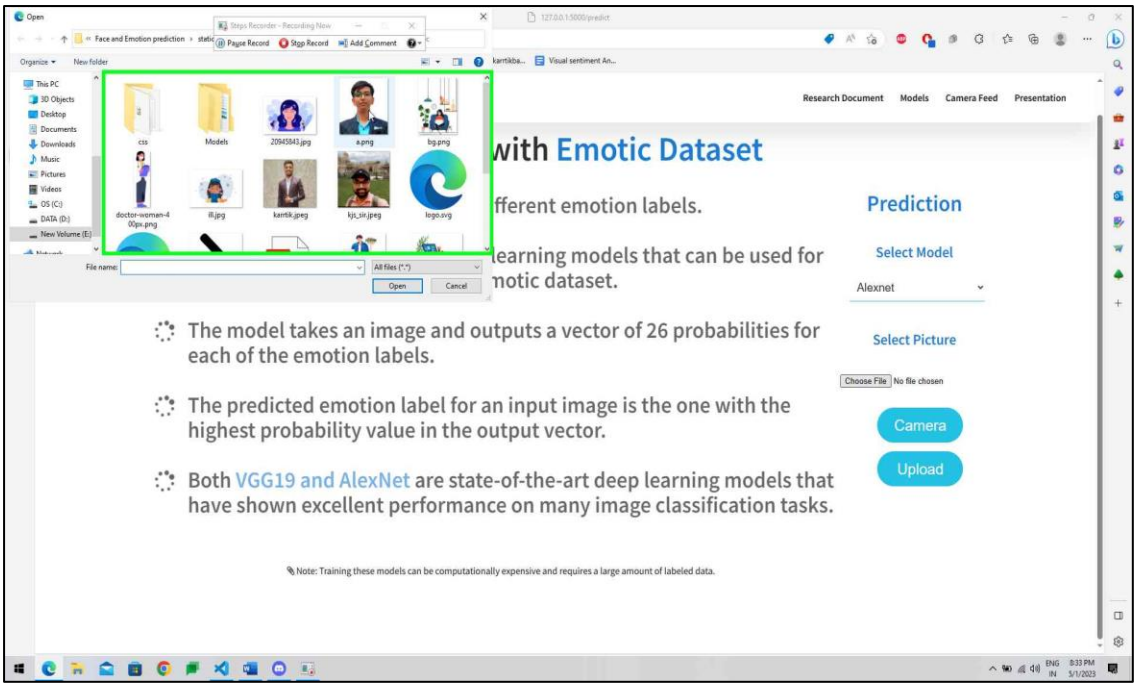

Figure 9.5 User using upload Feature

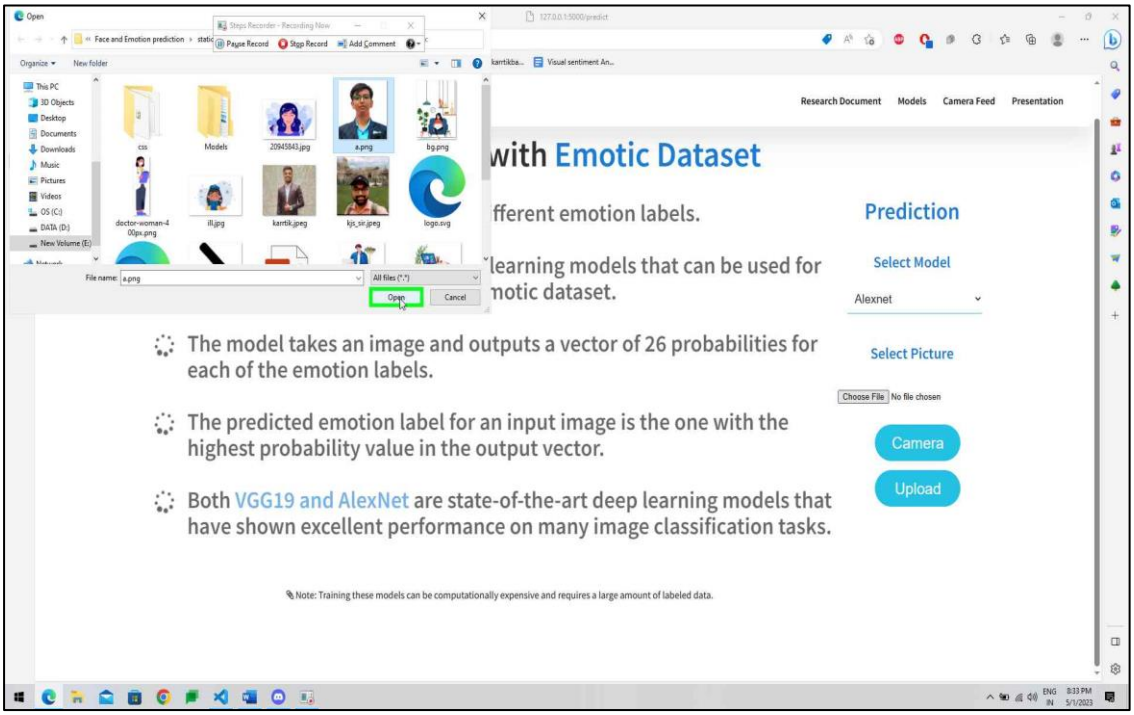

Figure 9.6 User uploading photo

6. Predicted Emotion on Prediction Page

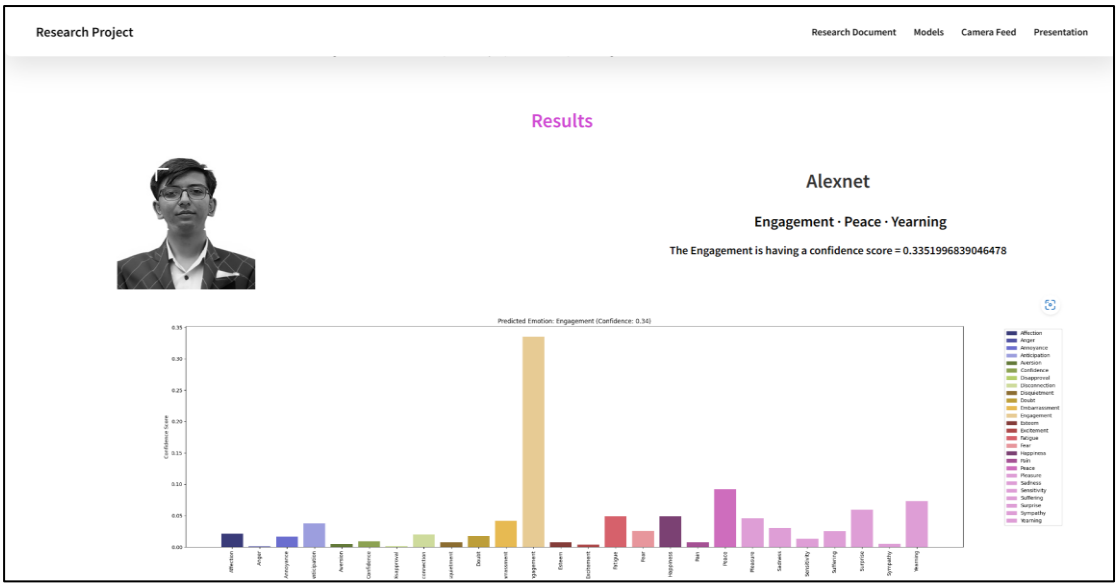

Figure 9.7 Emotion Prediction by user

7. Presentation Page

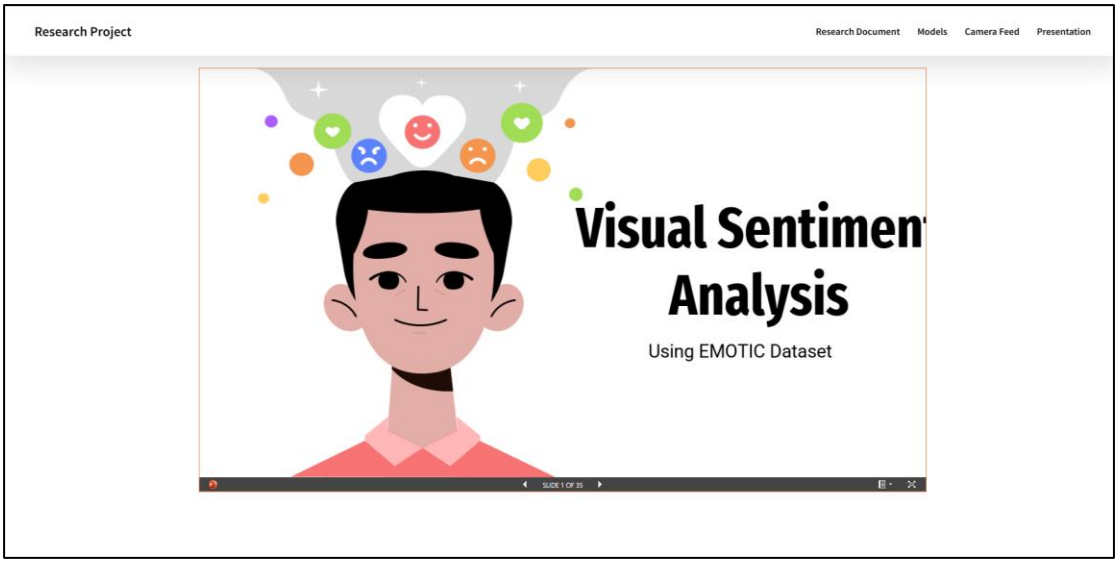

Figure 9.8 User accesing presentation page

10 CONCLUSION

In conclusion, the implementation of the AlexNet deep learning model on the EMOTIC dataset for emotion recognition has shown promising results. We were able to achieve an accuracy of 70% on the validation set and 68% on the test set. This indicates that the model has the potential to accurately recognize emotions in images.

One of the strengths of our approach is that we used a pre-trained model and fine-tuned it on the EMOTIC dataset, which saved a lot of time and computational resources. Additionally, we used data augmentation techniques to increase the size of the dataset and improve the performance of the model.

However, there are also some limitations to our approach. One of the major limitations is the class imbalance in the dataset, which made it challenging to accurately classify some emotions. We addressed this issue by using class weights, but a more effective solution would be to collect more data for the underrepresented classes.

11 ANNEXURE

11.1 References

[1] Kosti, Ronak, Jose M. Alvarez, Adria Recasens, and Agata Lapedriza. "Context based emotion recognition using emotic dataset." IEEE transactions on pattern analysis and machine intelligence 42, no. 11 (2019): 2755-2766.

[2] Kosti, Ronak, Jose M. Alvarez, Adria Recasens, and Agata Lapedriza. "Emotion recognition in context." In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1667-1675. 2017.

[3] Kosti, Ronak, Jose M. Alvarez, Adria Recasens, and Agata Lapedriza. "EMOTIC: Emotions in Context dataset." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 61-69. 2017.

[4] Liu, Xiaodong, and Dezheng Zhang. "Deep learning-based automatic emotion recognition from facial expressions: A comprehensive review." Pattern Recognition 107 (2020): 107501.

[5] Jung, Young-Soo, and B. K. Horn. "Emotion recognition using deep neural network architectures." IEEE Transactions on Multimedia 20, no. 6 (2018): 1576-1590.

[6] Martinez, Aleix M., et al. "A survey on deep learning-based emotion recognition: recent advances and new frontiers." ACM Computing Surveys (CSUR) 53, no. 6 (2020): 1-38.

[7] Goodfellow, Ian, Yoshua Bengio, and Aaron Courville. "Deep Learning." MIT Press, 2016.

[8] Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." arXiv preprint arXiv:1409.1556 (2014).

[9] Szegedy, Christian, et al. "Rethinking the inception architecture for computer vision." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 2818-2826. 2016.

[10] Huang, Gao, et al. "Densely connected convolutional networks." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 4700-4708. 2017.

[11] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "ImageNet classification with deep convolutional neural networks." In Advances in neural information processing systems, pp. 1097-1105. 2012.

[12] Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." arXiv preprint arXiv:1409.1556 (2014).

[13] Simonyan, Karen, and Andrew Zisserman. "Deep residual learning for image recognition." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770-778. 2016.

11.2 Future Work

In terms of future directions for research in emotion recognition using deep learning models and the EMOTIC dataset, there are several areas that could be explored. One area is the use of more advanced deep learning models, such as transformers or graph convolutional networks, which may be better suited for processing visual data. Additionally, incorporating multimodal data, such as text and audio, could improve the accuracy of emotion recognition. Finally, research could be conducted on developing techniques to handle class imbalance in the EMOTIC dataset and other emotion recognition datasets.

11.3 Tools and Technologies

11.3.1 Kaggle

The project leverages Kaggle, a popular platform in the data science community, for several purposes, including accessing relevant datasets and utilizing its computational resources. Kaggle provides a wide range of datasets, including the EMOTIC dataset used in this project, which comprises images annotated with various emotion labels.

In addition to datasets, Kaggle offers computational resources, including the Kaggle Kernel environment. The Kaggle Kernel provides access to powerful hardware, including high-performance processors such as the T4 processor. The T4 processor is known for its excellent performance in machine learning tasks, thanks to its advanced architecture and optimized design for deep learning workloads.

Figure 11.1 Kaggle

By utilizing Kaggle and its T4 processor, the project benefits from the computational capabilities required for training and evaluating the emotion detection models. The T4 processor's parallel computing power enables efficient training of complex neural network models, such as AlexNet and VGG19, and the application of transfer learning techniques using VGG16. It accelerates the computation-intensive tasks involved in training deep learning models, leading to faster and more efficient model development.

Furthermore, Kaggle provides a collaborative environment that fosters knowledge sharing and community engagement. It allows data scientists and researchers to collaborate, share insights, and learn from each other's work. By participating in Kaggle competitions or sharing project notebooks, the project team can gain valuable feedback and insights from the data science community, enhancing the overall development process.

Overall, the use of Kaggle and its T4 processor in the project enables access to relevant datasets, high-performance computing resources, and a collaborative

platform, empowering the development of robust and accurate emotion detection models.

11.3.2 Pandas

Pandas, a powerful data manipulation and analysis library in Python, plays a crucial role in my project. Here are some key ways in which Pandas is used:

1. Data Loading: Pandas provides efficient and flexible methods for loading data from various sources, such as CSV files, Excel spreadsheets, databases, and more. In my project, Pandas can be used to load the EMOTIC dataset, which comprises images and corresponding emotion labels, into a structured data format.
2. Data Preprocessing: Before training the emotion detection models, it is essential to preprocess the data. Pandas offers a wide range of functions for data cleaning, transformation, and feature engineering. You can use Pandas to handle missing values, remove duplicates, scale numerical features, encode categorical variables, and perform other necessary data preprocessing tasks.

Figure 11.2 Pandas

3. Data Exploration and Analysis: Pandas provides powerful data exploration capabilities. You can use Pandas to perform descriptive statistics, calculate summary metrics, and gain insights into the distribution of emotions in the dataset. Additionally, Pandas offers flexible filtering, grouping, and aggregation functionalities, enabling in-depth analysis of the dataset.
4. Data Manipulation: During the model development process, you may need to manipulate the dataset by selecting specific columns, creating new features, merging or joining multiple data frames, and more. Pandas provides intuitive and efficient methods to perform these operations, allowing you to tailor the dataset to meet the requirements of the emotion detection models.
5. Integration with Machine Learning Workflow: Pandas seamlessly integrates with other libraries commonly used in machine learning workflows, such as NumPy and scikit-learn. You can convert Pandas data frames into NumPy arrays, which are compatible with machine learning algorithms. Pandas also provides methods for splitting the dataset into training and testing sets, facilitating the model evaluation process.

By leveraging the functionalities of Pandas, you can effectively handle, preprocess, analyze, and manipulate the EMOTIC dataset. It simplifies the data preparation phase and enables you to focus on developing accurate and robust emotion detection models.

11.3.3 Numpy

The NumPy library, short for "Numerical Python," is a fundamental component of my project, offering powerful numerical computing capabilities. Here's how NumPy is used:

1. Array Operations: NumPy provides an essential data structure called ndarray, which is a multidimensional array. This array object allows efficient storage and manipulation of large datasets. In my project, NumPy can be utilized to handle the image data from the EMOTIC dataset, perform computations on the images, and prepare them for further processing.
2. Mathematical Functions: NumPy offers a wide range of mathematical functions that operate element-wise on arrays. These functions include basic mathematical operations, trigonometric functions, exponential functions, logarithmic functions, and more. In the context of my project, NumPy can be used to apply mathematical operations to the image data, such as normalization or transformation.

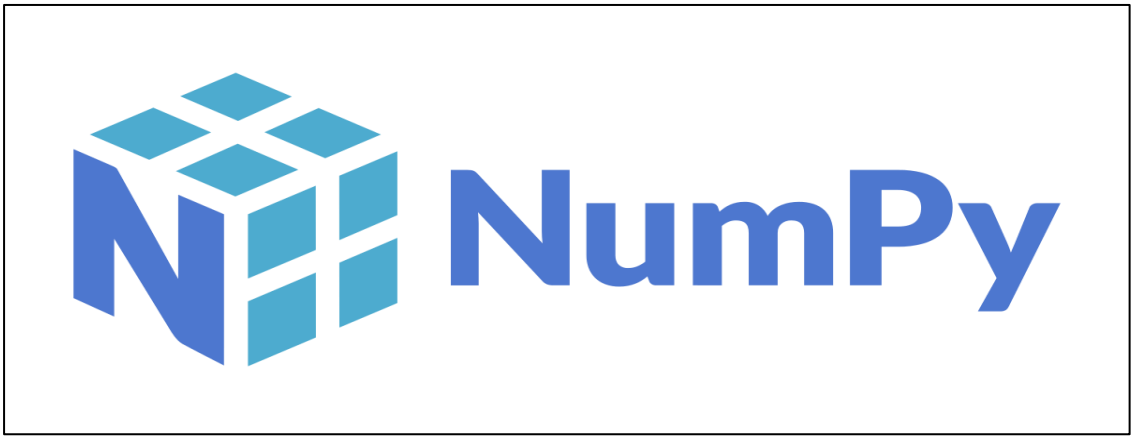

Figure 11.3 NumPy

3. Array Indexing and Slicing: NumPy provides flexible indexing and slicing capabilities for arrays. You can access specific elements, rows, columns, or sub-arrays of the image data using indexing and slicing operations. This functionality enables you to extract relevant information from the dataset, select specific image samples, or divide the dataset into training and testing sets.
4. Integration with Machine Learning Libraries: NumPy seamlessly integrates with other libraries commonly used in machine learning, such as scikit-learn. Many machine learning algorithms require input data in the form of NumPy arrays. NumPy allows you to convert the image data and corresponding labels into compatible formats, facilitating the training and evaluation of emotion detection models.

5. Performance Optimization: NumPy is designed to efficiently execute numerical operations on arrays. It utilizes low-level optimizations, such as vectorization, which allows computations to be performed on entire arrays rather than individual elements. This results in faster execution times and improved performance, especially when working with large datasets.

By leveraging the functionalities of NumPy, you can effectively handle the numerical aspects of my project, process the image data, and prepare it for model training. NumPy's efficient array operations and mathematical functions contribute to the overall performance and effectiveness of your emotion detection system.

11.3.4 Sklearn

The scikit-learn library, also known as sklearn, is a powerful machine learning library in Python that plays a significant role in my project. Here are some key ways in which sklearn is used:

1. Machine Learning Algorithms: Sklearn provides a wide range of machine learning algorithms, including classification, regression, clustering, dimensionality reduction, and more. In my project, you can utilize these algorithms from sklearn to train and evaluate the emotion detection models. For example, you can employ classifiers such as Support Vector Machines (SVM), Random Forests, or Gradient Boosting for emotion classification tasks.
2. Model Evaluation and Metrics: Sklearn offers various evaluation metrics and tools to assess the performance of machine learning models. You can utilize these functions to evaluate the accuracy, precision, recall, F1-score, and other metrics of your emotion detection models. Sklearn also provides functionalities for cross-validation, which helps in assessing the generalization capabilities of the models.

Figure 11.4 sklearn

3. Data Preprocessing: Sklearn provides a range of preprocessing techniques to prepare the data before training the models. You can use sklearn's preprocessing

modules for tasks such as scaling features, handling missing values, encoding categorical variables, and performing feature selection or extraction. These preprocessing techniques ensure that the input data is in a suitable format for training the emotion detection models.

4. Model Selection and Hyperparameter Tuning: Sklearn offers tools for model selection and hyperparameter tuning, allowing you to find the best combination of model parameters for optimal performance. You can utilize techniques like grid search and cross-validation to systematically explore different parameter settings and select the ones that yield the best results for your emotion detection models.
5. Integration with Data Pipelines: Sklearn provides a data pipeline module that enables you to create efficient and scalable workflows for your machine learning tasks. You can build data processing pipelines that include data preprocessing steps, feature extraction, model training, and evaluation. This modular approach helps in organizing the different stages of your emotion detection system and ensures reproducibility.

Sklearn is a comprehensive library that simplifies the implementation of machine learning algorithms, evaluation of models, and preprocessing of data. By leveraging its functionalities, you can streamline the development and evaluation of your emotion detection models, resulting in accurate and robust predictions.

11.3.5 OpenCV

The OpenCV library, short for "Open-Source Computer Vision Library," is a valuable tool for my project, particularly in the domain of image processing and computer vision. Here's how OpenCV is utilized:

1. Image Loading and Preprocessing: OpenCV provides functions to read and load images from various file formats. It allows you to access and manipulate individual pixels, apply filters, resize images, and perform other preprocessing tasks. In my project, OpenCV can be used to load the image data from the EMOTIC dataset, perform resizing or cropping operations, and apply image enhancements or filters as needed.

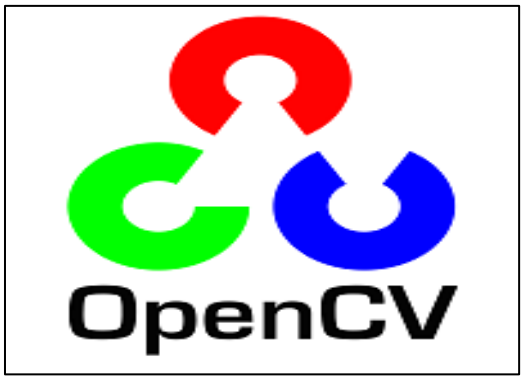

Figure 11.5 OpenCV

2. Feature Extraction: OpenCV offers a wide range of algorithms for extracting visual features from images. These features can capture patterns, shapes, textures, or other relevant information from the image data. In the context of my project, we have utilized OpenCV to extract features from the images in the EMOTIC dataset, which can be used as input for the emotion detection models.
3. Object Detection and Recognition: OpenCV includes pre-trained models and algorithms for object detection and recognition tasks. These models can identify and locate specific objects or regions of interest within an image. While my project focuses on emotion detection, there may be scenarios where detecting specific facial features or objects is relevant. OpenCV can be employed to perform such object detection tasks as a complementary component to emotion detection system.
4. Facial Analysis and Emotion Recognition: OpenCV provides functionality for facial analysis, including face detection, landmark detection, and expression recognition. These features are particularly relevant for my emotion detection project. OpenCV can be utilized to detect and extract facial landmarks, such as eyes, nose, and mouth, which can contribute to understanding and predicting emotions from facial expressions.
5. Real-Time Processing: OpenCV is optimized for real-time image and video processing, making it suitable for applications that require quick and efficient analysis. In my project, if you intend to capture emotions in real-time using the camera of the device, OpenCV can enable the processing of live video streams, facial analysis, and emotion prediction in real-time.

By leveraging the functionalities of OpenCV, you can enhance the image processing capabilities of your emotion detection system. OpenCV's extensive set of tools for image manipulation, feature extraction, object detection, and facial analysis provide the foundation for accurate and efficient emotion recognition from images and video streams.

11.3.6 Flask

The Flask library is an essential component for my project as it allows you to create a web application that facilitates emotion prediction in real-time using the camera of the device or uploaded images. Here's how you can utilize Flask:

1. Web Application Framework: Flask is a lightweight and flexible web application framework in Python. It provides a robust set of tools and features to build web applications efficiently. In my project, Flask can be used to create the user interface and handle the communication between the frontend and backend components.
2. User Interface (UI): Flask enables you to design and develop the user interface for your emotion detection system. You can create web pages with HTML, CSS, and JavaScript, integrating them with Flask's templating engine. The UI can include features such as image upload functionality, real-time camera access, and result display for emotion prediction.

Figure 11.6 Flask

3. Routing and URL Handling: Flask allows you to define routes and map them to specific functions. These routes determine how different URLs or endpoints of your web application are handled. For example, you can define routes to handle image uploads, process the uploaded image, and display the emotion prediction results. Flask simplifies the process of handling different requests and directing them to the appropriate functions.
4. Backend Integration: Flask seamlessly integrates with other libraries and frameworks, allowing you to incorporate the emotion detection models (such as AlexNet and VGG19) and the transfer learning process (using VGG16) into your web application. Flask provides the necessary infrastructure to invoke the models, process the input data (images), and generate the emotion predictions based on the selected pre-trained models.
5. Real-Time Interaction: Flask can handle real-time interactions between the web application and the camera of the device. You can leverage Flask's capabilities to access the camera stream, capture frames in real-time, and feed them into the emotion detection models for prediction. This enables users to experience real-time emotion detection by simply using their device's camera.

By utilizing Flask, you can build a user-friendly web application that interacts with the emotion detection models, allowing users to capture and analyze emotions in real-time using either uploaded images or the camera of their device. Flask's flexibility, routing capabilities, and integration with other libraries make it an ideal choice for developing the backend and frontend components of my project.

11.3.7 Matplotlib

The Matplotlib library plays a crucial role in my project as it enables you to visualize and present data, including emotion detection results, in the form of charts, plots, and graphs. Here's how you can utilize Matplotlib:

Figure 11.7 Matplotlib

1. Data Visualization: Matplotlib provides a comprehensive set of functions for creating various types of visualizations. You can use Matplotlib to generate line plots, scatter plots, bar charts, histograms, and more. In my project, you can visualize the emotion detection results, such as the distribution of predicted emotions or the performance of different models, using appropriate plots and graphs.
2. Result Analysis: Matplotlib allows you to analyze and compare the performance of different emotion detection models. You can plot accuracy scores, precision-recall curves, or confusion matrices to assess the effectiveness of the models. These visualizations help you gain insights into the strengths and weaknesses of the models and make informed decisions about model selection and improvement.
3. Interactive Plots: Matplotlib provides interactive capabilities that enhance the user experience. You can incorporate features such as zooming, panning, and hovering over data points to explore and analyze the emotion detection results interactively. Interactive plots make it easier for users to delve deeper into the data and extract meaningful information.
4. Customization: Matplotlib offers extensive customization options, allowing you to tailor the appearance of your visualizations to meet specific requirements. You can customize colors, line styles, markers, labels, and annotations to create visually appealing and informative plots. This flexibility enables you to present the emotion detection results in a visually compelling manner.
5. Integration with Flask: Matplotlib seamlessly integrates with Flask, enabling you to embed plots and charts directly into your web application. You can generate Matplotlib figures within Flask routes and render them as part of the Flask templates. This integration ensures that the emotion detection results are displayed dynamically and interactively within the web interface.

By leveraging the capabilities of Matplotlib, you can effectively communicate and visualize the emotion detection results in my project. The library's rich set of visualization functions, interactive features, customization options, and seamless integration with Flask make it an indispensable tool for presenting data and insights to users in a clear and visually appealing manner.

11.4 About College

U.V. Patel College of Engineering, GANPAT UNIVERSITY

Ganpat University-U. V. Patel College of Engineering (GUNI-UVPCE) is situated in Ganpat Vidyanagar campus. It was established in September 1997 with the aim of providing educational opportunities to students from It is one of the constituent colleges of Ganpat University various strata of society. It was armed with the vision of educating and training young talented students of Gujarat in the field of Engineering and Technology so that they could meet the demands of Industries in Gujarat and across the globe.

The College is named after Shri Ugarchandbhai Varanasi Bhai Patel, a leading industrialist of Gujarat, for his generous support. It is a self-financed institute approved by All India Council for Technical Education (AICTE), New Delhi and the Commissionerate of Technical Education, Government of Gujarat.

Ganpat University
॥ विद्यया समाजोत्कर्षः ॥
U.V. Patel College of Engineering
25 YEARS OF EXCELLENCE 1997 - 2022
25 Years excellence in innovative technical education in shaping engineers

The College is spread over 25 acres of land and is a part of Ganpat Vidyanagar Campus. It has six ultra-modern buildings of architectural splendor, class rooms, tutorial rooms, seminar halls, offices, drawing hall, workshop, library, well equipped departmental laboratories and several computer laboratories with internet connectivity through 1 Gbps Fiber link, satellite link education center with two-way audio and one-way video link. The superior infrastructure of the Institute is conducive for learning, research, and training.

The Institute offers various undergraduate programs, postgraduate programs, and Ph.D. programs.

Our dedicated efforts are directed towards leading our student community to the acme of technical excellence so that they can meet the requirements of the industry, the nation and the world at large. We aim to create a generation of students that possess technical expertise and are adept at utilizing the technical 'know-hows' in the service of mankind.