FACIAL EMOTIONAL, GENDER, AGE, FACE DETECTION USING WIDE RESIDUAL ARCHITECTURE

A PROJECT REPORT

Submitted by

VADDI JASWANTH REDDY (211418104298)

TIRUPATI KIRAN RAJ (211418104295)

KANALA ARAVIND (211418104109)

in partial fulfillment for the award of the

degree of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

PANIMALAR ENGINEERING COLLEGE

(An Autonomous Institution, Affiliated to Anna University, Chennai)

MAY 2022

PANIMALAR ENGINEERING COLLEGE

(An Autonomous Institution, Affiliated to Anna University, Chennai)

BONAFIDE CERTIFICATE

Certified that this project report **"FACIAL EMOTIONAL, AGE, GENDER, FACE DETECTION USING WIDE RESIDUAL ARCHITECTURE"** is the bonafide work of "**VADDI JASWANTH REDDY [211418104298], TIRUPATI KIRAN RAJ [211418104295] ,KANALA ARAVIND [211418104109]"** who carried out the project work under my supervision.

SIGNATURE

Dr.S.MURUGAVALLI M.E.,Ph.D.,
HEAD OF THE DEPARTMENT
DEPARTMENT OF CSE,
PANIMALAR ENGINEERING COLLEGE,
NASARATHPETTAI,
POONAMALLEE,
CHENNAI-600 123.

SIGNATURE

Mr.D.ELANGOVAN, M.E.,
ASSOCIATE PROFESSOR
DEPARTMENT OF CSE,
PANIMALAR ENGINEERING COLLEGE,
NASARATHPETTAI,
POONAMALLEE,
CHENNAI-600 123

Certified that the above mentioned students were examined in the End Semester project viva-voce held on_____.

INTERNAL EXAMINER

EXTERNAL EXAMINER

DECLARATION BY THE STUDENT

We **VADDI JASWANTH REDDY [211418104298] ,TIRUPATI KIRANRAJ [211418104295] , KANALA ARAVIND [211418104109]** hereby declare that this project report titled " **FACIAL EMOTIONAL, GENDER, AGE, FACE DETECTION USING WIDE RESIDUAL ARCHITECTURE"**, under the guidance of **Mr.D.ELANGOVAN, M.E.,** is the original work done by us and we have not plagiarized or submitted to any other degree in any university by us.

ACKNOWLEDGEMENT

We would like to express our deep gratitude to our respected Secretary and Correspondent **Dr.P.CHINNADURAI, M.A., Ph.D.,** for his kind words and enthusiastic motivation, which inspired us a lot in completing this project.

We express our sincere thanks to our Directors **Tmt.C.VIJAYARAJESWARI**, **Dr.C.SAKTHI KUMAR, M.E., Ph.D**. and **Dr.SARANYASREE SAKTHI KUMAR B.E.,M.B.A.,Ph.D.,** for providing us with the necessary facilities to undertake this project.

We also express our gratitude to our Principal **Dr.K.Mani, M.E., Ph.D.** who facilitated us in completing the project.

We thank the Head of the CSE Department, **Dr. S.MURUGAVALLI , M.E.,Ph.D.,** for the support extended throughout the project.

We would like to thank my **Project Coordinator Dr.N.PUGHAZENDI M.E.,Ph.D.,** and **Project Guide Mr.D.ELANGOVAN, M.E.,** and all the faculty members of the Department of CSE for their advice and encouragement for the successful completion of the project.

VADDI JASWANTH REDDY

TIRUPATI KIRAN RAJ

KANALA ARAVIND

ABSTRACT

Automatic emotion recognition based on facial expression is an interesting research field, which has been presented and applied in several areas such as safety, health, and human-machine interfaces. Various techniques to interpret, code facial expressions and extract these features are being developed in order to have a better prediction, with high accuracy and precision by the computer. With the remarkable success of deep learning, the different types of architectures of this technique are exploited to achieve better performance. Quick and accurate emotion recognition may increase the possibilities of computers, robots, and integrated environments to recognize human emotions, and respond accordingly to the social rules. The state-of-the-art techniques are only capable of detecting age, gender, and face behavior. New techniques are required for the accurate estimation of human facial emotions. I cover deep learning-based Wide Resonant Architecture to analyze granular level emotional changes.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ACRONYMS AND ABBREVIATIONS

CV - Computer Vision

CNN - Convolutional Neural Network

HCI – Human Computer Interaction

LSTM – Long Short Term Memory

FER – Facial Emotion Recognition

DCNN – Deep Convolutional Neural Network

CHAPTER 1
INTRODUCTION

1.1 ABOUT THE PROJECT

Deep Learning has found huge applications in the fields of Computer vision. Some of the most important applications of computer vision are in the fields that deal with facial data. Face Detection and recognition are being widely used in security-based applications.

The human face may be a storehouse of various information about personal characteristics including identity, emotional expression, gender, age. The looks of the face are affected considerably by aging. This plays a significant role in non-verbal communication between humans. Age and gender are two key facial attribute that play a really foundational role in social interactions making age and gender estimation from one face image a very important task in machine learning applications like access control, human-computer interaction, law enforcement and marketing intelligence and visual surveillance.

Automatic gender classification and age detection may be a fundamental task in computer vision which has recently attracted immense attention. It plays a very important role in an exceedingly wide selection of real-world applications like targeted advertisement, forensic science, visual surveillance, content-based searching and human-computer interaction systems. For instance, wide residual architecture is used to display advertisement-supported different gender and different age brackets. This method may be employed in different mobile applications where there is some age restricted content in order that only appropriate users can see this content. However, gender classification and age approximation is a still difficult task. A model is proposed which can initially perform feature extraction on the input image which can classify eyes, lips, beard and hair.

Supporting these features the model will classify the gender as male or female. Haar Cascade is used for feature extraction purposes. The Age is estimated with the assistance of the Caffe Model. The age classifier takes an image of an individual's face of size 256×256 as an input to the algorithm that is then cropped to 227×227. The age classifier returns an integer representing the age range of the individual. There are 8 possible age ranges and the age classifier returns an integer between 0 and seven. The gender classifier returns a binary result where 1 indicates male and 0 represents female.

CHAPTER 2

LITERATURE SURVERY

CHAPTER 2
LITERATURE REVIEWS

TITLE 1: Face Emotion Detection Using Deep Learning, September 2021, 2nd International Conference On Advances In Computing, Communication, Embedded And Secure Systems (ACCESS)

AUTHOR: Pradnya Kedari; Mihir Kapile; Divya Kadole; Sagar Jaikar

DESCRIPTION: Using deep learning algorithm to identify the basic human emotions (e.g., anger, fear, neutral, happy, sad, surprise, etc.) on multiple datasets, including FER-2013 (Facial Expression Recognition 2013) and CK+ (Extended Cohn-Kanade), the accuracy is 60% for FER 2013 dataset.

ADVANTAGES : For CK +, the model achieved significant improvement, highest accuracy was 99.1% and average accuracy was 93%.

DISADVANTAGES : The average accuracy can still be improved

TITLE 2: Deep Facial Expression Recognition: A Survey, March 2020, IEEE Transactions on Affective Computing

AUTHOR: Shan Li and Weihong Deng

DESCRIPTION: Deep neural networks have increasingly been leveraged to learn discriminative representations for automatic FER. Recent deep FER systems generally focus on two important issues: overfitting caused by a lack of sufficient training data and expression-unrelated variations, such as

illumination, head pose and identity bias.

ADVANTAGES : The Effeciency of the model is really great such that it supports very high column datasets without changing the approach.

DISADVANTAGES : When the dataset is changed, there is a dip in the accuracy.

TITLE 3: Efficient Facial Expression Recognition Algorithm Based on Hierarchical Deep Neural Network Structure, March 2019, Department of IT Engineering, Sookmyung Women's University, Seoul, South Korea

AUTHOR: Ji-hae Kim, Byung-gyu Kim, Partha Pratim Roy, Da-mi Jeong

DESCRIPTION: An efficient facial expression recognition algorithm combining appearance feature and geometric feature based on deep neural networks for more accurate and efficient facial expression recognition. The appearance feature-based network extracts the holistic feature of the LBP feature containing the AUs information. The geometric feature-based network extracts the dynamic feature, which is the face landmark change centered on the coordinate movement between the neutral face and the peak emotion.

ADVANTAGES : Combining appearance feature and geometric feature turns out to be effective strategy for this project.

DISADVANTAGES : The facial landmarks sometimes not extracted properly.

TITLE 4: Survey on AI-Based Multimodal Methods for Emotion Detection, March 2019, High-Performance Modelling and Simulation for Big Data Applications

AUTHOR: Catherine Marechal, Dariusz Mikołajewski, Krzysztof Tyburek,Piotr Prokopowicz, Lamine Bougueroua, Corinne Ancourt, Katarzyna Węgrzyn-Wolska

DESCRIPTION: A novel multimodal implicit emotion recognition system can be built upon an AI-based model designed to extract information on the emotion from different devices. To feed such a model, a video data captured by the camera embedded in the user's device (laptop, desktop, tablet, etc.), an audio signals collected from microphones embedded in mobile devices, and motion signals generated by sensors in wearable devices can be used.

ADVANTAGES : The system was integrated on all the devices like tablet, desktop and mobile

DISADVANTAGES : At times, the audio input has an improper frequency which impacted the output.

TITLE 5: Facial emotion recognition using deep learning: review and insights, August 2020, The 2nd International Workshop on the Future of Internet of Everything (FIoE), Belgium

AUTHOR: Wafa Mellouk, Wahida Handouzi

DESCRIPTION: Recent research on FER systems has been discussed, different architectures of CNN and CNN LTSM have also been elaborated. FERis one of the most important ways of providing information about the emotional state, but they are always limited by learning only the six-basic emotion plus neutral.

ADVANTAGES : CNN LSTM proves to be an best deep learning algorithms for detecting human emotions

DISADVANTAGES : Only few layers of CNN were used to build the Convolutional architecture.

TITLE 6: Deep Learning for Understanding Faces: Machines May Be Just as Good, or Better than Humans, Jan. 2018, IEEE Signal Processing Magazine, Volume: 35, Issue: 1

AUTHOR: Rajeev Ranjan; Swami Sankaranarayana Ankan Bansal; Navaneeth Bodla

DESCRIPTION: Different modules involved in designing an automatic face recognition system and the role of deep are discussed and learned for each of them. Some open issues regarding DCNNs for face recognition problems are then discussed.

ADVANTAGES : The comparison study between different models helped in determining the best suited algorithm for face recognition.

DISADVANTAGES : Since many algorithms are trained , the efficiency of the model is not great

TITLE 7: Cost-effective real-time recognition for human emotion-age-gender using deep learning with normalized facial cropping preprocess, March 2021, Multimedia Tools and Applications

AUTHOR: Ta-Te Lu, Sheng-Cheng Yeh, Chia-Hui Wang, Min-Rou Wei

DESCRIPTION: The EAGR system first applies normalized facial cropping (NFC) as a preprocessing method for training data before data augmentation, then uses convolution neural network (CNN) as three training models for recognizing seven emotions (six basics plus one neutral emotion), four age groups, and two genders.

ADVANTAGES : The training accuracy of the model is 93% and the testing accuracy of the model is 96%

DISADVANTAGES : The input data was taken from an image , not from an real time web camera.

TITLE 8: Face Recognition and Age Estimation Implications of Changes in Facial Features: A Critical Review Study June 2018, Universiti Sains Malaysia, Penang

AUTHOR: Rasha Ragheb Atallah, Amirrudin Kamsin, Maizatul Akmar Ismail, Sherin Ali Abdelrahman, Saber Zerdoumi

DESCRIPTION: A complete survey of the state-of the art techniques for age estimation and face recognition have been reviewed and discussed via face images.

ADVANTAGES : The results of this study indicated that the SVM (99.80%) and the LBP (98.7%) had the highest detection accuracy rates, along with GAP (99.85 %).

DISADVANTAGES : The state of the art algorithm sometimes give low accuracy because of its architecture.

TITLE 9: A Study on Facial Expression Recognition in Assessing Teaching Skills: Datasets and Methods, The Fifth Information Systems International Conference 2019

AUTHOR: Pipit Utami, Rudy Hartanto, Indah Soesanti

DESCRIPTION: The trend of developing recognition classifier is the use of CNN and modified CNN. The things that need to be explored are the description of the types of FET(needs to be oriented towards academic emotion) and the suitability of the CNN algorithm modifications as the solution to four problems that may arise during teaching

ADVANTAGES : The ROC Curve and the epoch value of the project is great.

DISADVANTAGES : The dataset that has been used in the project has very less number of images.

.

TITLE 10: Age estimation from faces using deep learning: a comparative analysis, July 2020, Computer Vision and Image Understanding Volume 196, 102961

AUTHOR: Alice Othmania, Abdul Rahman Taleb, Hazem Abdelkawy, Abdenour Hadid

DESCRIPTION: The framework based on Xception network outperforms the state-of-the-art methods based on deep or shallow learning for automatic age estimation with an MAE of 2.35 years when pre-trained on ImageNet and an MAE of 2.01 when pre-trained on CASIA Web face dataset.

ADVANTAGES : The accuracy of the project is very high when compared with other methodologies.

DISADVANTAGES : Since many algorithms are trained , the efficiency of the model is not great

CHAPTER 3

SYSTEM ANALYSIS

CHAPTER 3
SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

Facial emotion recognition is to analyze a given facial expression and use the results to classify specific emotions. With the emergence of the convolutional neural network, many scholars started deploying CNN (convolutional neural network) algorithms to extract image features. The existing system detects age, gender, and face behavior only. One major issue in age, gender, and human emotion estimation are how to extract effective representation features from a facial image. Through the analysis of the works of literature on existing methods, it can be found that the facial emotional features extracted by state-of-the-art methods have the problem that the original emotional information is easy to be lost. In addition, the generalization and robustness of these network models are also poor and the accuracy of facial expression recognition is not high.

The existing systems are simple and effective but are not robust in terms of pose, illumination, and human expression changes. The existing system detects age, gender, and facial behavior only through the analysis of the works of literature on existing methods, it can be found that the facial emotional features extracted by state-of-the-art methods have the problem that the original emotional information is easy to be lost. The typical human-computer interaction (HCI) lacks users' emotional state and loses a great deal of information during the process of interaction. In addition, the generalization and robustness of these network models are also poor and the accuracy of facial expression recognition is not high.

3.1.1 DISADVANTAGES OF EXISTING SYSTEM

- The existing systems are simple and effective but are not robust in terms of

pose, illumination, and human expression changes.

- The existing system detects age, gender, and face behavior only
- Through the analysis of the works of literature on existing methods, it can be found that the facial emotional features extracted by state-of-the-art methods have the problem that the original emotional information is easy to be lost.
- The typical human-computer interaction (HCI) lacks users' emotional state and loses a great deal of information during the process of interaction.
- In addition, the generalization and robustness of these network models are also poor and the accuracy of facial expression recognition is not high.

3.2 PROPOSED SYSTEM

An automated, low-cost, and real-time system is proposed for age, gender, and facial emotional estimation from face images. To achieve this, face detection and pose estimation methods are adopted to acquire frontal face images. The proposed architecture tracks and responds to human behavior in real-time. It integrates eye-tracking for deeper insights into the effect of various stimuli on emotions. The Face recognition analysis detects faces in images or video and then uses face tracking and provides unity of action to accurately deliver the gender, emotion and age of faces in a roughly frontal position. The use of generic CNN is absent in the proposed system. A specialized system called Wide Resonant Architecture is used to analyze granular level emotional changes. The proposed system ensures that human emotional behavior is detected with high-level accuracy. In emotion detection three steps are used namely face detection, features extraction and emotion classification using deep learning with our proposed model which gives better results than the previously used models.

The proposed system consists of Four modules –

a) Face Detection
b) Gender Detection
c) Age Detection
d) Emotion estimation

In the proposed method, computation time is reduced, validation accuracy is increased and loss also decreased, and further performance evaluation is achieved which compares our model with the previous model. The proposed model emphasizes that emotion detection using deep convolutional neural networks can improve the performance of a network with more information. The main contributions and advantages of the proposed system can be summarized as follows: FER is usually carried out in three stages involving face detection, feature extraction, and expression classification. The Lightweight model & fast processing - low data size and memory usage ensure fast yet accurate gender, age, and emotion estimation in milliseconds. Platform & device independent - Face analysis works flawlessly on any mobile or web application Secured - No personal data such as photos or names is stored or processed by default, ensuring privacy. Interpretability - The model is easy to integrate into any environment. High accuracy - In an attempt to use this facial pose estimation system, it is noticed that wide residual architecture gives a high accuracy rate, very precise measurements, permits high deployment and authentication.

3.2.1 ADVANTAGES OF PROPOSED SYSTEM

- FER is usually carried out in three stages involving face detection, feature extraction, and expression classification
- Lightweight model & fast processing - Low data size and memory usage ensure fast yet accurate gender, age, and emotion estimation in milliseconds

- Platform & device independent - Face analysis works flawlessly on any mobile or web application
- Secured - No personal data such as photos or names is stored or processed by default, ensuring privacy.
- Interpretability - The model is easy to integrate into any environment
- High accuracy - In our attempt to use this facial pose estimation system, we noticed that it gives a high accuracy rate, has very precise measurements, and permits for high deployment and authentication.

3.3 REQUIREMENT SPECIFICATION

3.3.1. HARDWARE REQUIREMENTS

Processor	: Pentium Dual Core 2.00GHZ
Hard disk	: 120 GB
Mouse	: Logitech.
RAM	: 2GB (minimum)
Keyboard	: 110 keys enhanced

3.3.2. SOFTWARE REQUIREMENTS

Operating system	: Windows7 (with service pack 1), 8, 8.1 and 10
IDE	:Anaconda1
Backend	: Python
Frontend	:Html, CSS

3.4. SOFTWARE SPECIFICATIONS - ANACONDA

Anaconda is an open-source package manager for Python and R. It is the most popular platform among data science professionals for running Python and R implementations. There are over 300 libraries in data science, so having a robust distribution system for them is a must for any professional in this field. Anaconda simplifies package deployment and management. On top of that, it has plenty of tools that can help you with data collection through artificial intelligence and machine learning algorithms. With Anaconda, It can easily set up, manage, and share Conda environments. Moreover, you can deploy any required project can be deployed with a few clicks when you're using Anaconda. There are many advantages to using Anaconda and the following are the most prominent ones among them : Anaconda is free and open-source. This means you can use it without spending any money. In the data science sector, Anaconda is an industry staple. It is open-source too, which has made it widely popular. It must be known how to become a data science professional, you must know how to use Anaconda for Python because every recruiter expects this skill. It is a must-have for data science.

It has more than 1500 Python and R data science packages, so you don't face any compatibility issues while collaborating with others. For example, suppose a colleague sends a project which requires packages called A and B but only have package A. Without having package B, it wouldn't be able to run the project. Anaconda mitigates the chances of such errors. It can easily collaborate on projects without worrying about any compatibility issues. It gives a seamless environment which simplifies deploying projects. It can deploy any project with just a few clicks and commands while managing the rest. Anaconda has a thriving community of data scientists and machine learning professionals who use it regularly. When an issue is encountered chances are, the community has already answered the same. On the other hand, when asked with people in the community about the issues faced, it's a very helpful community ready to help new learners. With Anaconda, it is easy to create and train machine learning and

deep learning models as it works well with popular tools including TensorFlow, Scikit-Learn, and Theano. You can create visualizations by using Bokeh, Holoviews, Matplotlib, and Datashader while using Anaconda.

How to Use Anaconda for Python

Now all the basics in our Python Anaconda are discussed. To start using this package manager.

Listing All Environments

To begin using Anaconda, you'd need to see how many Conda environments are present in your machine.

conda env list

It will list all the available Conda environments in themachine.

Creating a New Environment

It can create a new Conda environment by going to the required directory and use this command:

conda create -n <your_environment_name>

It can replace <your_environment_name> with the name of your environment. After entering this command, conda will ask you if you want to proceed to which you should reply with y:

proceed ([y])/n)?

On the other hand, for creating an environment with a particularversion of Python, you should use the following command:

conda create -n <your_environment_name> python=3.6

Similarly, for creating an environment with a particular package, you can use the following command:

conda create -n <your_environment_name>pack_name
Here, replace pack_name with the name of the package you want to use.

If there is a .yml file, use the following command to create a new

Conda environment based on that file:

conda env create -n <your_environment_name> -f <file_name>.yml

Here discussed how you can export an existing Conda environment to a .yml file later in this article.

Activating an Environment

For activating a Conda environment by using the following command:

conda activate <environment_name>

The system should activate the environment before you start working on the same. Also, replace the term <environment_name> with the environment name to activate. On the other hand, to deactivate an environment use the following command:

conda deactivate

Installing Packages in an Environment

Now that with an activated environment, you can install packages into it by using the following command:

conda install <pack_name>

Replace the term <pack_name> with the name of the package to install in your Conda environment while using this command.

Updating Packages in an Environment

To update the packages present in a particular Conda environment,

use the following command:

conda update

The above command will update all the packages present in the environment. However, to update a package to a certain version, use the following command:

conda install <package_name>=<version>

Exporting an Environment Configuration

Suppose to share your project with someone else (colleague, friend, etc.). While sharing the directory on Github, it would have many Python packages, making the transfer process very challenging. Instead of that, create an environment configuration .yml file and share it with that person. Now, they can create an environment similarly by using the .yml file.

For exporting the environment to the .yml file, first it need to activate the same and run the following command:

conda env export ><file_name>.yml

The person who wants the same environment only has to use the exported file by using the 'Creating a New Environment' command shared before.

Removing a Package from an Environment

To uninstall a package from a specific Conda environment, use the following command:

conda remove -n <env_name><package_name>

On the other hand, to uninstall a package from an activated environment, you'd have to use the following command:

conda remove <package_name>

Deleting an Environment

Sometimes, to add a new environment but remove one. In such cases, the user must know how to delete a Conda environment, which can be done soby using the following command:

conda env remove –name <env_name>

The above command would delete the Conda environment right away.

CHAPTER 4

SYSTEM DESIGN

CHAPTER 4
SYSTEM DESIGN

4. SYSTEM ANALYSIS

This diagram is nothing but a simple description of all the entities that have been incorporated into the system. The diagram represents the relations between each of them and involves a sequence of decision-making processes and steps. You can simply call it a visual or the whole process and its implementation. All functional correspondences are explained in this diagram.

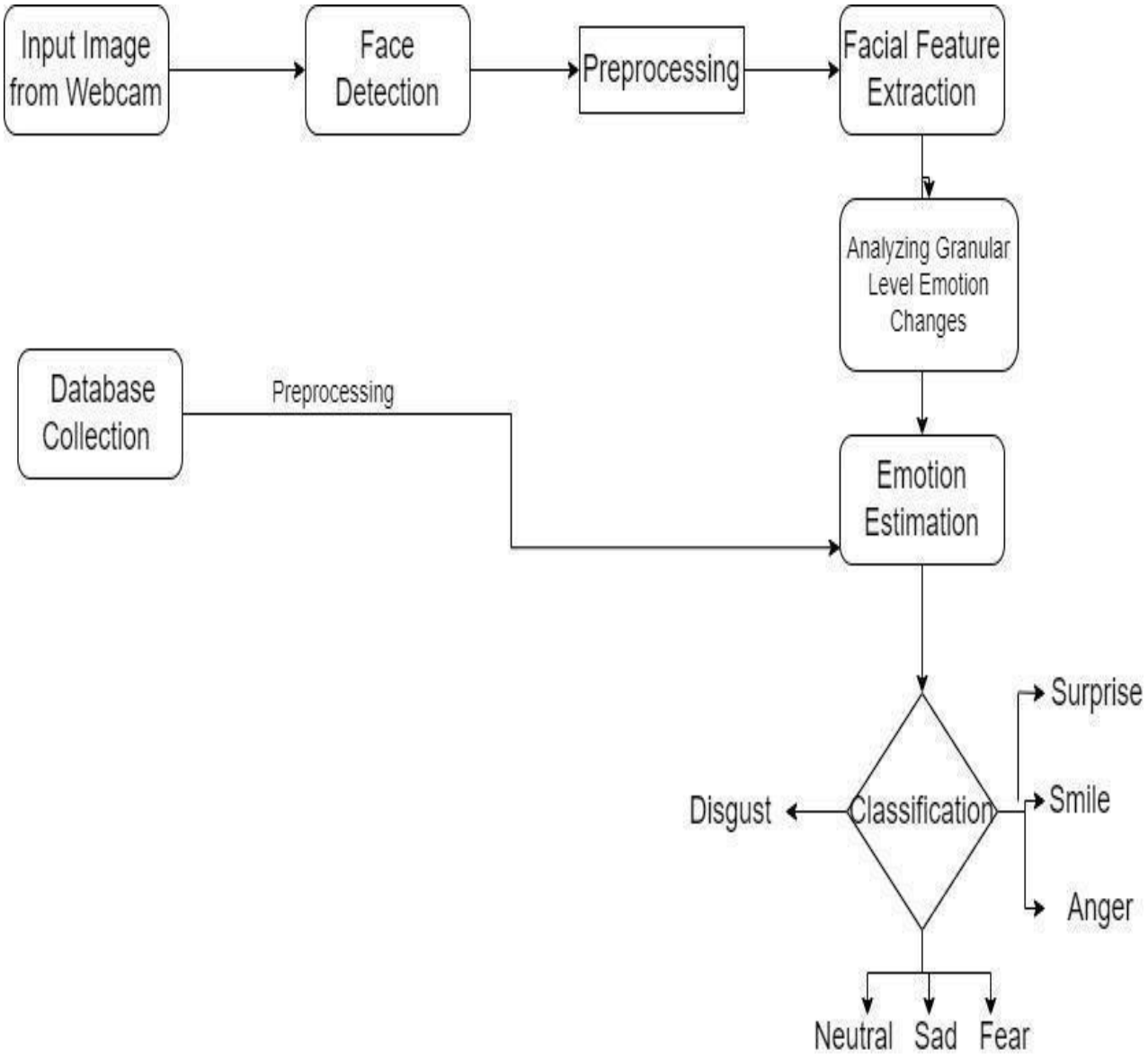


Fig 4.1 – System Architecture

4.1. DATA FLOW DIAGRAM

A Data-Flow Diagram (DFD) is a way of representing a flow of a data of a process or a system (usually an information system). The DFD also provides information about the outputs and inputs of each entity and the process itself. A data-flow diagram has no control flow, there are no decision rules and no loops. Specific operations based on the data can be represented by a flowchart. There are several notations for displaying data-flow diagrams. For each data flow, at least one of the endpoints (source and / or destination) must exist in a process. The refined representation of a process can be done in another data-flow diagram, which subdivides this process into sub-processes. The data-flow diagram is part of the structured-analysis modeling tools. While using UML, the activity diagram typically takes over the role of the data-flow diagram. A special form of data-flow plan is a site-oriented data-flow plan. Data-flow diagrams can be regarded as inverted Petri nets, because places in such networks correspond to the semantics of data memories. DFD consists of processes, flows, warehouses, and terminators.

Data Flow Diagram Symbols

- **Process**

A process transforms incoming data flow into outgoing data flow.

- **Data Store**

Data stores are repositories of data in the system. Sometimes it’s also referred to as files.

- **Data Flow**

Data flows are pipelines through which packets of information flow.

Label the arrows with the name of the data that moves through it.

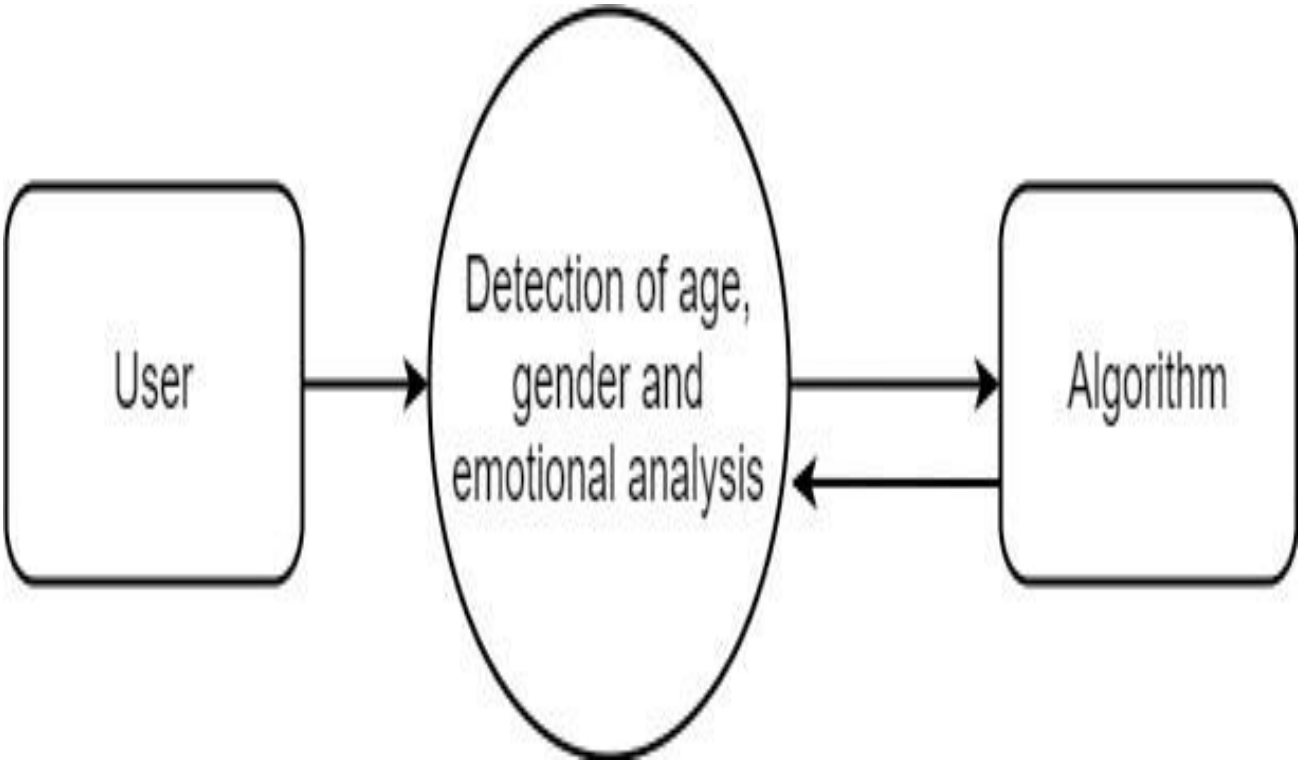
- **External Entity**

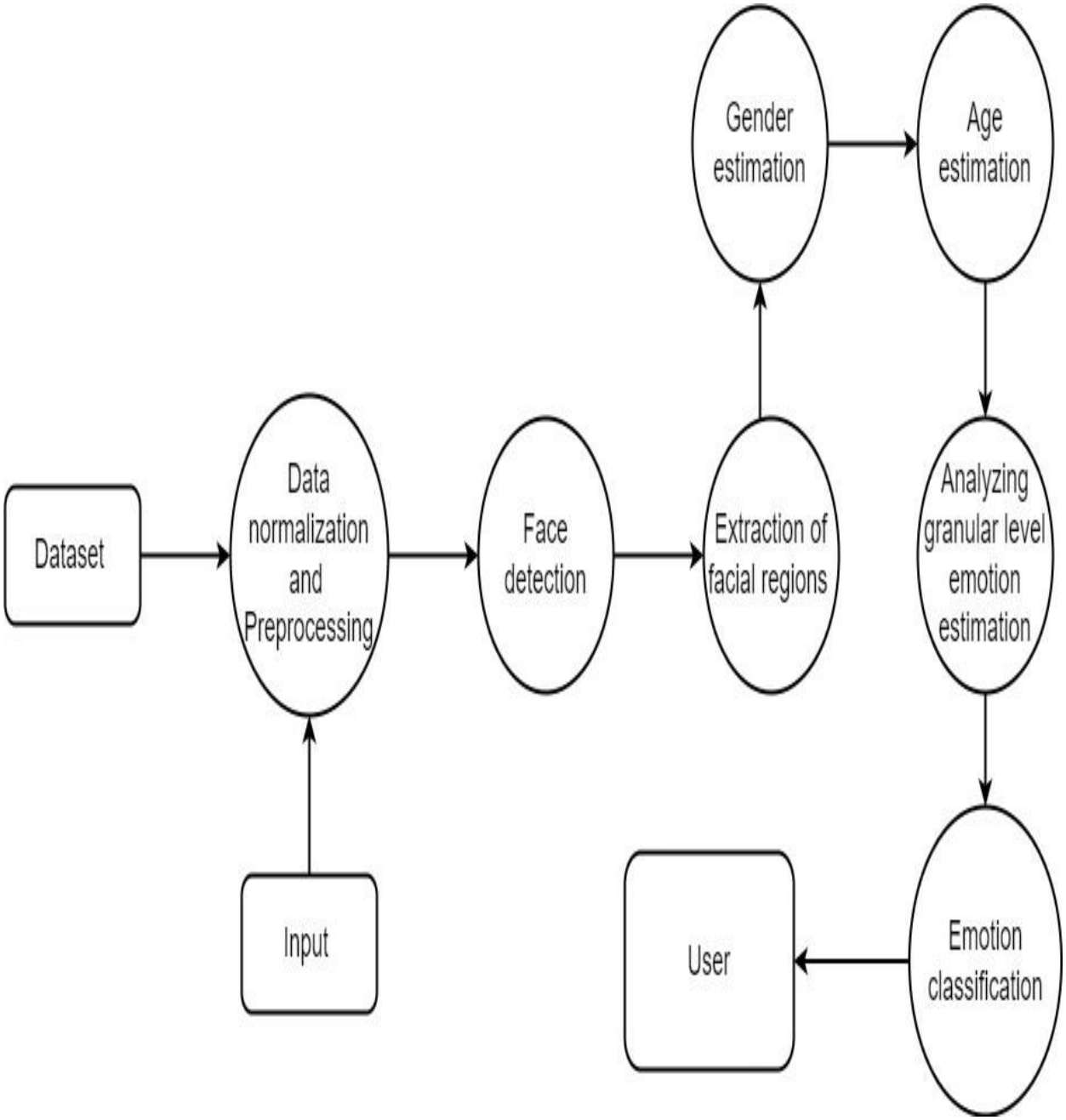
External entities are objects outside the system, with which the system communicates. These are sources and destinations of the system's inputs and outputs.

DATA FLOW DIAGRAM

Level 0

Fig 4.2 Data flow diagram (Level-0)





Level 1

Fig 4.3 Data flow diagram (level-1)

4.2. ENTITY RELATIONSHIP DIAGRAM

- **Definition**

Entity-relationship diagram depicts relationship between data objects. The attribute of each data objects noted in the entity-relationship diagram can be described using a data object description. In software engineering, an entity-relationship model (ERM) is an abstract and conceptual representation of data. Entity-relationship modeling is a database modeling method, used to produce a type of conceptual schema or semantic data model of a system, often a relational database, and its requirements in a top-down fashion. Diagrams created by this process are called entity-relationship diagrams, ER diagrams, or ERDs. Data flow diagram serves two purposes.

1) To provide an indication of how data is transformed as it moves through the system.
2) To depict the functions that transform the data flow.

1. One-to-One

One instance of entity (A) is associated with one other instance of another entity (B).

For example, in a database of sign in, each customer name (A) is associated with only one security mobile number (B).

2. One-to-Many

One instance of an entity (A) is associated with zero, one or many instances of another entity (B), but for one instance of entity B there is only one instance of entry A.

For example, for a company with all employees working in one building, the building name (A) is associated with many different employees (B), but those employees all share the same singular association with entity A.

3. Many-to-Many

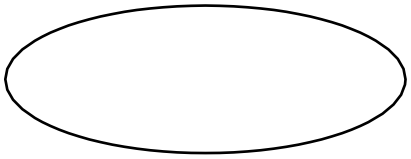
One instance of an entity (A) is associated with one, zero or many instances of entity A.

For example, for a company in which all of its employees work on multiple projects, each instance of an employee (A) is associated with many instances of a project (B), and at the same time, each instance of a project (B) has multiple employees (A) associated with it.

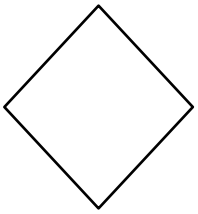
- **SYMBOLS USED**

External entity –

Attribute –

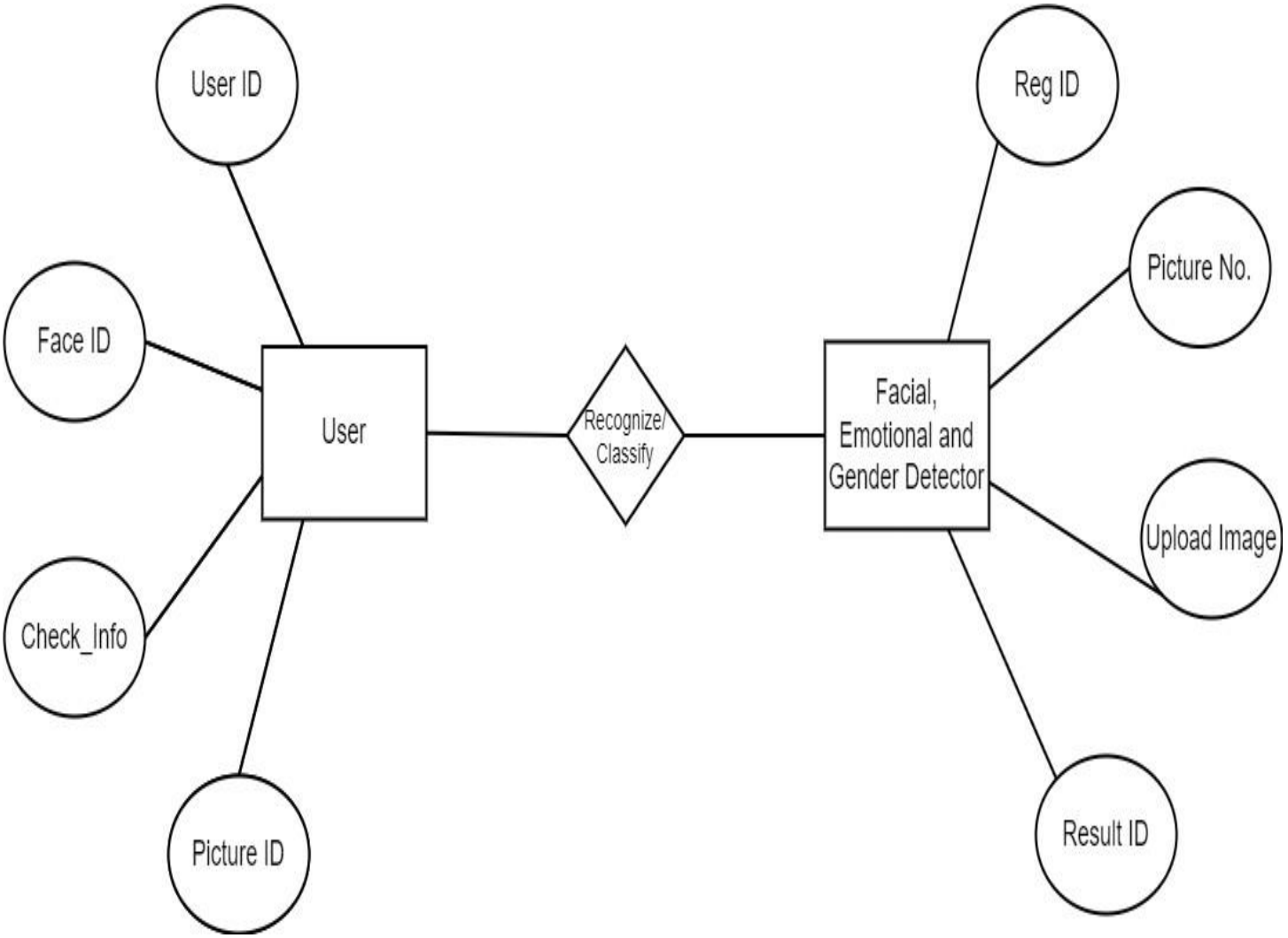


Relationship –



Data flow –

Fig 4.4 E-R Diagram



4.3 Use Case Diagram

An use case diagram is used to represent the dynamic behavior of a system. It encapsulates the system's functionality by incorporating use cases, actors, and their relationships. It models the tasks, services, and functions required by a system/subsystem of an application. It depicts the high-level functionality of a system and also tells how the user handles a system. Use case diagrams can summarize the details of your system's users (also known as actors) and their interactions with the system. To build one, you'll use a set of specialized symbols and connectors. A use case diagram doesn't go into a lot of detail—for example, don't expect it to model the order in which steps are performed. Instead, a proper use case diagram depicts a high-level overview of the relationship between use cases, actors, and systems. Experts recommend that use case diagrams be used to supplement a more descriptive textual use case.An effective use case diagram can help your team discuss and represent:

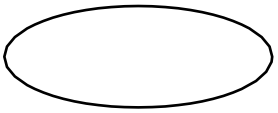
- Scenarios in which your system or application interacts with people, organizations, or external systems
- Goals that your system or application helps those entities (known as actors) achieve
- The scope of your system

➢ **Symbols Used**

Actor –

Data flow–

System Function–



- **Use Case Diagram**

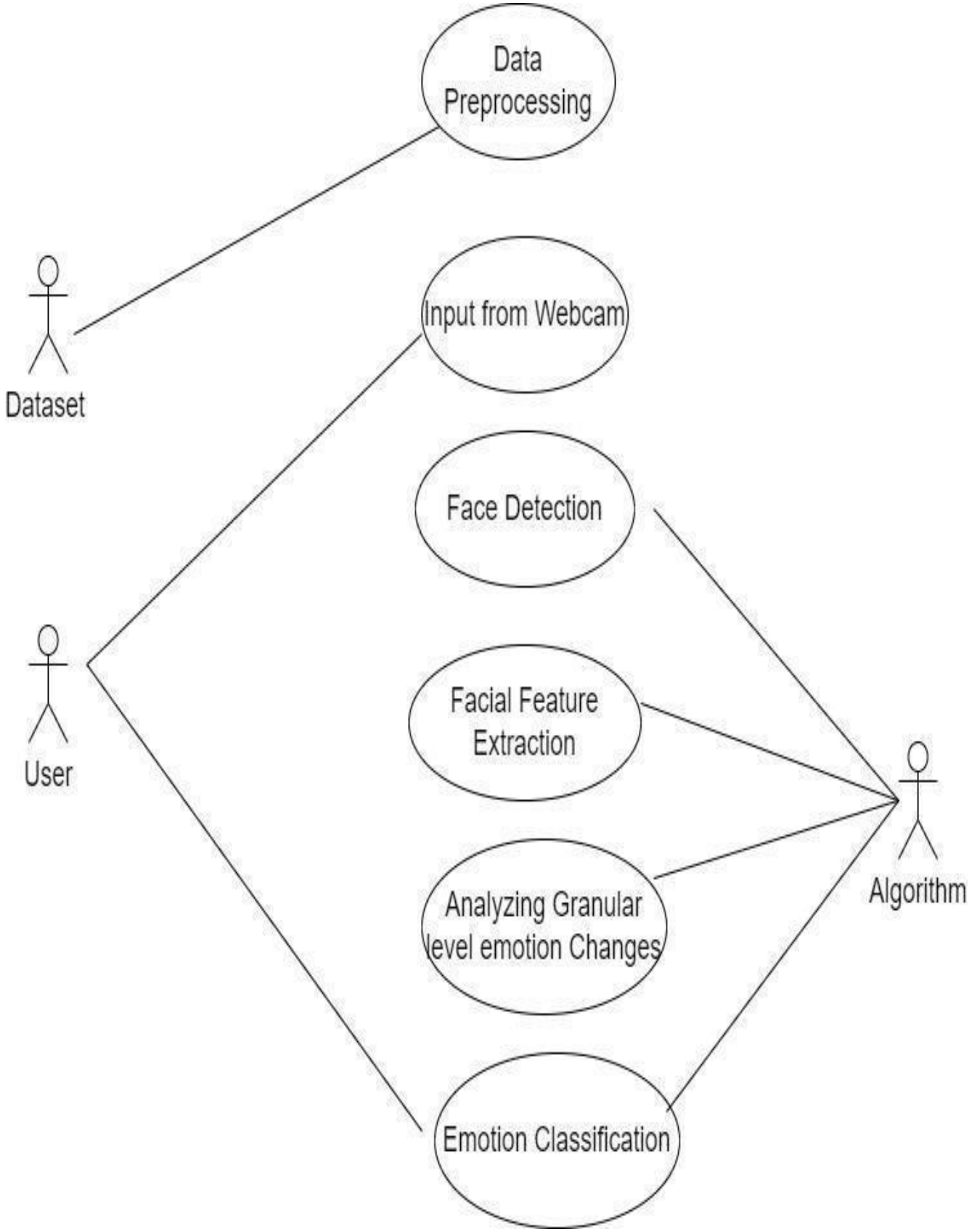


Fig 4.5 Use Case Diagram

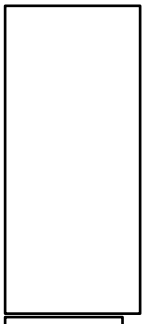
4.4 Sequence Diagram

A sequence diagram is a type of interaction diagram because it describes how—and in what order—a group of objects works together. These diagrams are used by software developers and business professionals to understand requirements for a new system or to document an existing process. Sequence Diagrams are interaction diagrams that detail how operations are carried out. They capture the interaction between objects in the context of a collaboration. Sequence Diagrams are time focus and they show the order of the interaction visually by using the vertical axis of the diagram to represent time what messages are sent and when.Sequence Diagrams captures:

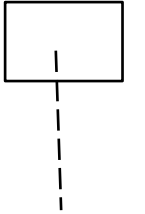
- the interaction that takes place in a collaboration that either realizes a use case or an operation (instance diagrams or generic diagrams)
- high-level interactions between user of the system and the system, between the system and other systems, or between subsystems (sometimes known as system sequence diagrams)

Object symbol -

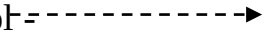
Activation Box -



Lifeline Symbol -



Asynchronous create message symbol



Reply message symbol

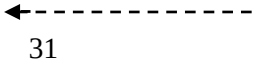
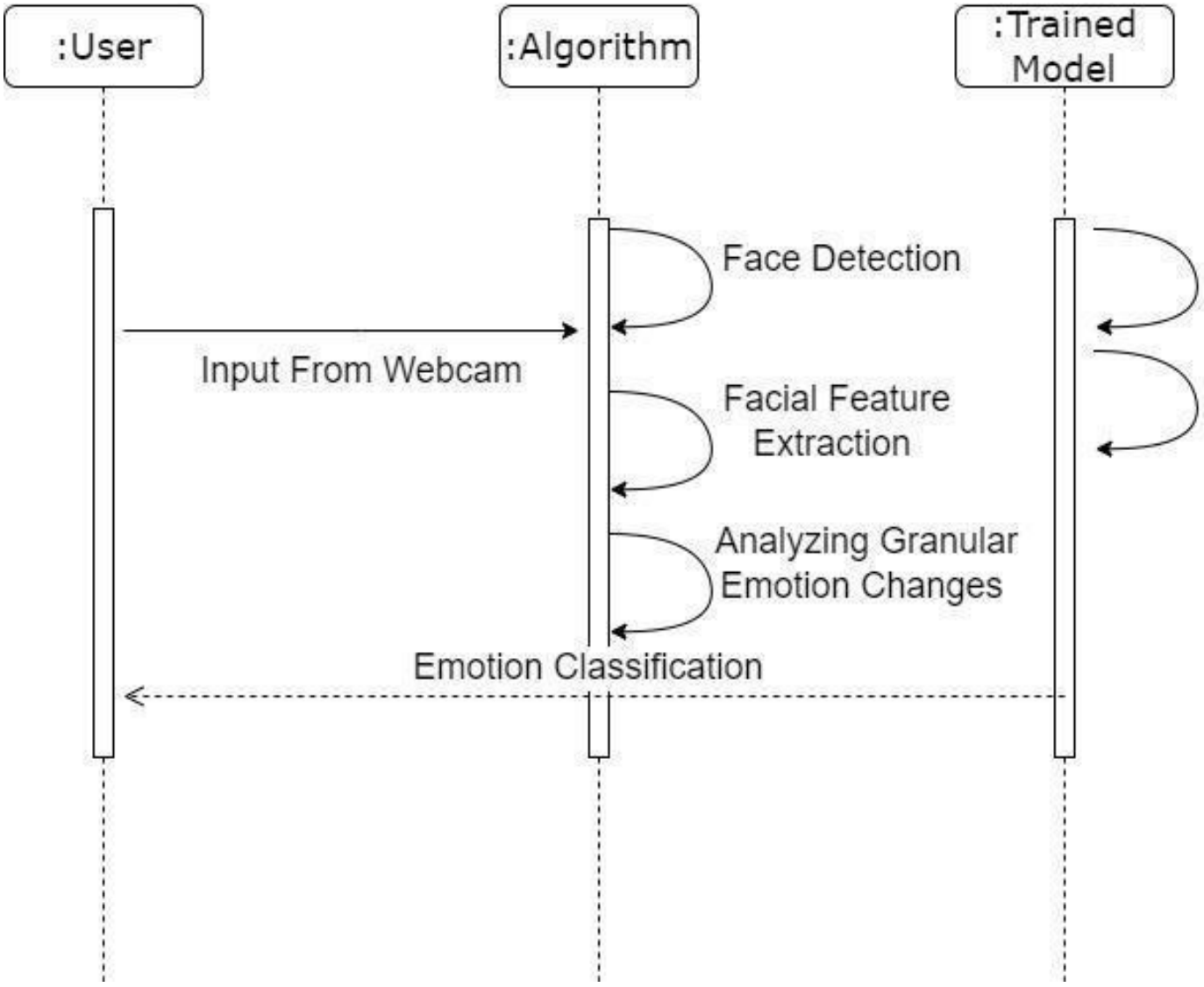


Fig 4.6 Sequence Diagram



4.5. Activity Diagram

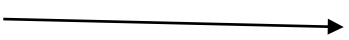
The basic purpose of activity diagrams is similar to the other four diagrams. It captures the dynamic behavior of the system. Other four diagrams are used to show the message flow from one object to another but the activity diagram is used to show the message flow from one activity to another.Activity is a particular operation of the system. Activity diagrams are not only used for visualizing the dynamic nature of a system, but they are also used to construct the executable system by using forward and reverse engineering techniques. The only missing thing in the activity diagram is the message part.It does not show any message flow from one activity to another. Activity diagram is sometimes considered as the flowchart. Although the diagrams look like a flowchart, they are not. It shows different flows such as parallel, branched,concurrent, and single.The purpose of an activity diagram can be described as −

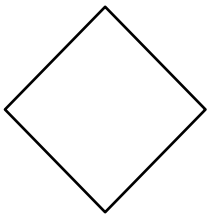
- Draw the activity flow of a system.
- Describe the sequence from one activity to another.
- Describe the parallel, branched and concurrent flow of the system

Symbols Used

Initial State

Activity State 

Action Flow 

Decision Node 

Activity Diagram

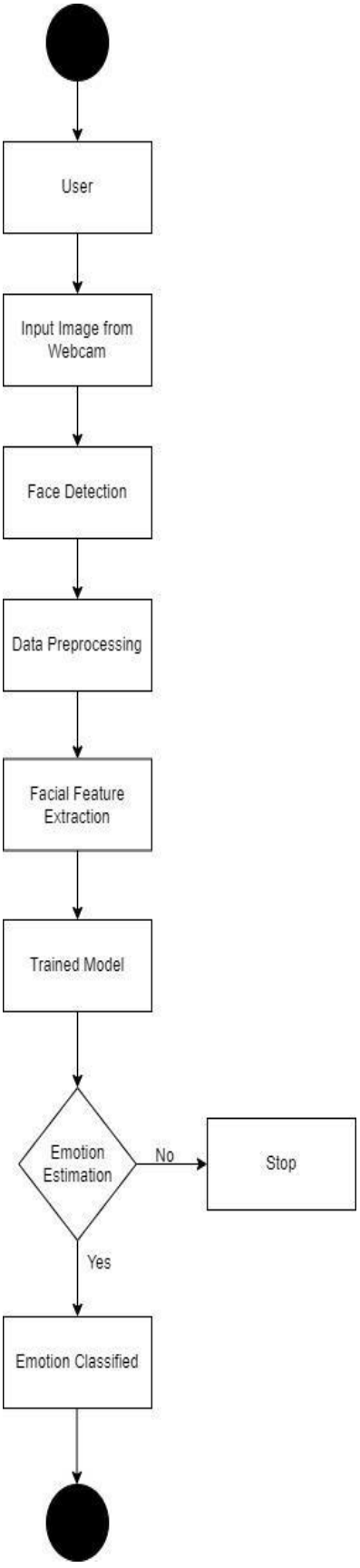
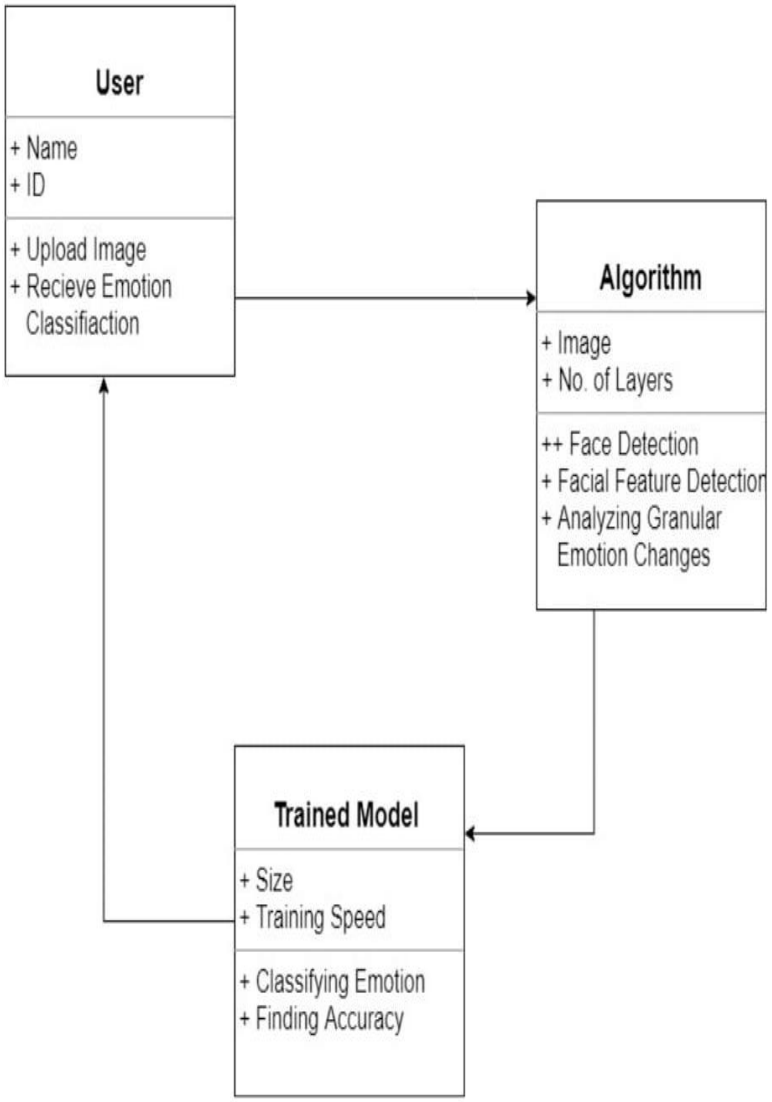


Fig 4.7 Activity Diagram

4.6 **CLASS DIAGRAM**



CHAPTER 5

MODULE DESCRIPTION

CHAPTER 5
MODULE DESCRIPTION

LIST OF MAIN MODULES

- Module 1: Face Detection
- Module 2: Gender Detection
- Module 3: Age Detection
- Module 4: Emotion Estimation

5.1 MODULE 1: Face Detection

The process of searching for faces is called face detection. In real-time, searching for face in the sequential images containing face and background is seen as the first step of the systems, and the corresponding tasks of face analysis are implemented. Face detection is to search for faces with different expressions, sizes and angles in images in possession of complicated light and background, and then feedback parameters of the face.

5.2 MODULE 2: Gender Detection

There are some common features in the human face that identify male and female. The first component of the proposed architecture deals with gender estimation from face images.

One major issue in gender estimation is how to extract effective representation features from a facial image. To achieve this, face detection and pose estimation methods are adopted to acquire frontal face images. An advanced machine learning technique is used to provide gender classification. The gender estimation process consists of three steps: Detection and extraction of the facial region from the input image/video. selection of the frontal face images from the, extracted facial regions using head pose estimation, and gender estimation using statistical facial features. The features extracted from the face image are used to

estimate the gender of the recipient.

5.3 MODULE 3: Age Detection

Human features are very sensitive to texture and skin tone, and most of the selected features are located around the meaningful areas for recognition, such as eyebrows, nose, cheekbones, and jaw-line. Automated age estimation of the human face involves detecting, tracking, and normalizing the face in an image sequence. After the face and eyes are detected, the facial image can be normalized as a fixed-size image using the localized eye positions.

This module features features such as the location of the pupils, eye corners, lip boundaries, etc. This is because these features are bound to change with age. This algorithm is trained on a large data set of different faces to detect the approximate age of a person based on such features. The accuracy of the prediction depends on conditions such as lighting, head pose, etc.

5.4 MODULE 4: Emotion Estimation

Facial emotion recognition refers to the separation of specific facial states from a given static image or dynamic video sequence, so as to determine the psychological emotions of the object to be recognized. Facial emotions can be divided into seven categories: happy, sad, fearful, angry, surprised, disgusted, and neutral. The first thing to do for facial expression recognition is to preprocess the collected images, and then carry out feature extraction and classification recognition.

The third component, Emotion estimation, detects facial expressions from images or videos and returns the probability distribution of each of the universal emotions: happiness, sadness, anger, fear, surprise, disgust, and additionally neutral. Automatic facial expression analysis of one or multiple faces in real-time can also be performed.

CHAPTER 6

TESTING

CHAPTER 6
TESTING

Testing Methodologies

There are many different types of testing methods or techniques used as part of the software testing methodology. Some of the important testing methodologies are:

6.1 Unit Testing

A unit is the smallest possible testable software component. A unit can be function or procedure implemented in a procedural programming languages. A unit may also be a small-sized COTS component purchased from an outside vendor that is undergoing evaluation by the purchaser, or a simple module retrieved from an in-house reuse library. Unit test results are recorded for future testing process. This result document used for integration and system tests.

Some of the phases for unit test planning are;

- ❖ Describe Unit Test Approach and Risks.
- ❖ Identify Unit features to be tested.
- ❖ Add levels of detail to the test plan.

TEST NO.	INPUT	EXPECTED BEHAVIOUR	TESTING	STATUS
1	User uploading video as an input	Video file updated	Unit Testing	Pass
2	User uploading image an input	File not accepted because only video inputs	Unit Testing	Fail
3	Capturing face in video	Age, gender, emotion, face emotion prediction	Output Testing	Pass
4	Capturing other objects in video	Age, gender, emotion , face recognition not detected	Output Testing	Fail

CHAPTER 7

CONCLUSION AND FUTURE ENHANCEMENTS

CONCLUSION AND FUTURE ENHANCEMENTS

7.1 CONCLUSION

Emotion is an important topic in different fields such as biomedical engineering, psychology, neuroscience, and health. Emotion recognition could be useful for the diagnosis of brain and psychological disorders. Human faces are different for different people on the basis of various parameters. The uniqueness and measurement of the different parameters help us to detect the age, gender, and emotional behavior of the person. In summary, the state-of-the-art methods for human face recognition have already achieved a high accuracy rate which led to its practical applications. However, it cannot be denied that using the existing method poses many challenges, mainly slow functioning, and inefficiency. Due to increased applications in fields such as authentication, targeted advertisements, video surveillance, and human-robot interaction, we developed a fast and robust, real-time facial emotion, age, and gender estimation system.

The proposed framework is not only much faster than the previous work but also maintains competitive accuracy with state-of-the-art facial emotion detection systems.

Our proposed model for face recognition and pose estimation systems is beneficial to the world for advanced applications such as access and security, payments, and criminal identifications.

7.2 FUTURE ENHANCEMENTS

The proposed system has shown excellent performance in the face recognition systems with a high accuracy rate and a much higher speed up rate as compared to the previously used state-of-the-art methods. It also shows promising performance and higher estimation rates than the existing models for age, gender, and emotion detection and classification.

The Present approach may be successful, but the limitations of the current

knowledge and experience still concern tools for automatic non-invasive emotion measurement and analysis.

Despite some limitations and challenges of the facial emotion detection architecture, there is scope for us to improvise these frameworks in the near future.

APPENDIX 1

CODING

```
# -*- coding: utf-8 -*-
from pathlib import Path
import cv2
import dlib
import sys
import numpy as np
import argparse
from contextlib import contextmanager
from wide_resnet import WideResNet
from keras.utils.data_utils import get_file
from keras.models import load_model
from keras.preprocessing.image import img_to_array
classifier = load_model('model/emotion_little_vgg_2.h5')

from os import listdir
from os.path import isfile, join
import os
import cv2

# Define Image Path Here
image_path = "images/"
```

```
emotion_classes = {0: 'Angry', 1: 'Fear', 2: 'Happy', 3: 'Neutral', 4: 'Sad', 5:
'Surprise'}

def draw_label(image, point, label, font=cv2.FONT_HERSHEY_SIMPLEX,
               font_scale=0.8, thickness=1):
    size = cv2.getTextSize(label, font, font_scale, thickness)[0]
    x, y = point
    cv2.rectangle(image, (x, y - size[1]), (x + size[0], y), (255, 0, 0),
cv2.FILLED)
    cv2.putText(image, label, point, font, font_scale, (255, 255, 255), thickness,
lineType=cv2.LINE_AA)



# Define our model parameters
depth = 16
k = 8
weight_file = None
margin = 0.4
image_dir = None

# load model and weights
weight_file = 'weights.28-3.73.hdf5'
img_size = 64
model = WideResNet(img_size, depth=depth, k=k)()
model.load_weights(weight_file)

detector = dlib.get_frontal_face_detector()
```

```
image_names = [f for f in listdir(image_path) if isfile(join(image_path, f))]

for image_name in image_names:
    frame = cv2.imread("images/" + image_name)
    preprocessed_faces_emo = []

    input_img = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
    img_h, img_w, _ = np.shape(input_img)
    detected = detector(frame, 1)
    faces = np.empty((len(detected), img_size, img_size, 3))

    preprocessed_faces_emo = []
    if len(detected) > 0:
        for i, d in enumerate(detected):
            x1, y1, x2, y2, w, h = d.left(), d.top(), d.right() + 1, d.bottom() + 1,
d.width(), d.height()
            xw1 = max(int(x1 - margin * w), 0)
            yw1 = max(int(y1 - margin * h), 0)
            xw2 = min(int(x2 + margin * w), img_w - 1)
            yw2 = min(int(y2 + margin * h), img_h - 1)
            cv2.rectangle(frame, (x1, y1), (x2, y2), (255, 0, 0), 2)
            # cv2.rectangle(img, (xw1, yw1), (xw2, yw2), (255, 0, 0), 2)
            faces[i, :, :, :] = cv2.resize(frame[yw1:yw2 + 1, xw1:xw2 + 1, :],
(img_size, img_size))
            face =  frame[yw1:yw2 + 1, xw1:xw2 + 1, :]
            face_gray_emo = cv2.cvtColor(face, cv2.COLOR_BGR2GRAY)
            face_gray_emo = cv2.resize(face_gray_emo, (48, 48), interpolation =
cv2.INTER_AREA)
            face_gray_emo = face_gray_emo.astype("float") / 255.0
```

```
            face_gray_emo = img_to_array(face_gray_emo)
            face_gray_emo = np.expand_dims(face_gray_emo, axis=0)
            preprocessed_faces_emo.append(face_gray_emo)
        # make a prediction for Age and Gender
        results = model.predict(np.array(faces))
        predicted_genders = results[0]
        ages = np.arange(0, 101).reshape(101, 1)
        predicted_ages = results[1].dot(ages).flatten()

        # make a prediction for Emotion
        emo_labels = []
        for i, d in enumerate(detected):
            preds = classifier.predict(preprocessed_faces_emo[i])[0]
            emo_labels.append(emotion_classes[preds.argmax()])

        # draw results
        for i, d in enumerate(detected):
            label = "{}, {}, {}".format(int(predicted_ages[i]),
                                        "F" if predicted_genders[i][0] > 0.4 else "M",
emo_labels[i])
            draw_label(frame, (d.left(), d.top()), label)

    cv2.imshow("Emotion Detector", frame)
    filename = "output_images/"+image_name
    cv2.imwrite(filename,frame)
    cv2.waitKey(0)

cv2.destroyAllWindows()
# Deep-Surveillance-Monitor-Facial-Emotion-Age-Gender-Recognition-System
```

Computer Vision module for detecting emotion, age and gender of a person in any given image, video or real time webcam. A custom VGG16 model was developed and trained on open source facial datasets downloaded from Kaggle and IMDB. OpenCV,dlib & keras were used to aid facial detection and video processing. The final system can detect the emotion, age and gender of people in any given image, video or real time webcam.

Screenshots
Detect Emotion, Age, Gender in Any Image!

Detect Emotion, Age, Gender in Webcam!

Technical Concepts
VGG The VGG network architecture was introduced by Simonyan and Zisserman in their 2014 paper, Very Deep Convolutional Networks for Large Scale Image Recognition.
This network is characterized by its simplicity, using only 3×3 convolutional layers stacked on top of each other in increasing depth. Reducing volume size is handled by max pooling. Two fully-connected layers, each with 4,096 nodes are then followed by a softmax classifier

Resnet Unlike traditional sequential network architectures such as AlexNet, OverFeat, and VGG, ResNet is instead a form of “exotic architecture” that relies on micro-architecture modules (also called “network-in-network architectures”).
The term micro-architecture refers to the set of “building blocks” used to construct the network. A collection of micro-architecture building blocks (along

with your standard CONV, POOL, etc. layers) leads to the macro-architecture

Technologies used

  OpenCv
  Keras
  Numpy
  Dlib
  Python

How to Install and Use

  Install Python. Download my repo and change to directory of repo.
  On command prompt, run pip install -r requirements.txt
  On commmand prompt, run jupter notebook
  Open and run my jupyter notebook

# -*- coding: utf-8 -*-
from pathlib import Path
import cv2
import dlib
import sys
import numpy as np
import argparse
from contextlib import contextmanager
from wide_resnet import WideResNet

```
from keras.utils.data_utils import get_file
from keras.models import load_model
from keras.preprocessing.image import img_to_array
classifier = load_model('model/emotion_little_vgg_2.h5')
# from keras.layers.normalization import BatchNormalization
from tensorflow.keras.layers import (
    BatchNormalization, SeparableConv2D, MaxPooling2D, Activation, Flatten,
Dropout, Dense
)

from os import listdir
from os.path import isfile, join
import os
import cv2
#import urllib

image_path = "images/"

modhash = 'fbe63257a054c1c5466cfd7bf14646d6'
emotion_classes = {0: 'Angry', 1: 'Fear', 2: 'Happy', 3: 'Neutral', 4: 'Sad', 5:
'Surprise'}

def draw_label(image, point, label, font=cv2.FONT_HERSHEY_SIMPLEX,
               font_scale=0.8, thickness=1):
    size = cv2.getTextSize(label, font, font_scale, thickness)[0]
    x, y = point
    cv2.rectangle(image, (x, y - size[1]), (x + size[0], y), (255, 0, 0),
cv2.FILLED)
    cv2.putText(image, label, point, font, font_scale, (255, 255, 255), thickness,
```

```
lineType=cv2.LINE_AA)

# Define our model parameters
depth = 16
k = 8
weight_file = None
margin = 0.4
image_dir = None

weight_file = 'weights.28-3.73.hdf5'
img_size = 64
model = WideResNet(img_size, depth=depth, k=k)()
model.load_weights(weight_file)

detector = dlib.get_frontal_face_detector()

# Initialize Webcam
cap = cv2.VideoCapture(0)

while True:
    ret, frame = cap.read()
    #imgResp=urllib.request.urlopen(url)
    #imgNp=np.array(bytearray(imgResp.read()),dtype=np.uint8)
    #img=cv2.imdecode(imgNp,-1)
    #frame=img

    preprocessed_faces_emo = []

    input_img = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
```

```
    img_h, img_w, _ = np.shape(input_img)
    detected = detector(frame, 1)
    faces = np.empty((len(detected), img_size, img_size, 3))

    preprocessed_faces_emo = []
    if len(detected) > 0:
        for i, d in enumerate(detected):
            x1, y1, x2, y2, w, h = d.left(), d.top(), d.right() + 1, d.bottom() + 1,
d.width(), d.height()
            xw1 = max(int(x1 - margin * w), 0)
            yw1 = max(int(y1 - margin * h), 0)
            xw2 = min(int(x2 + margin * w), img_w - 1)
            yw2 = min(int(y2 + margin * h), img_h - 1)
            cv2.rectangle(frame, (x1, y1), (x2, y2), (255, 0, 0), 2)
            # cv2.rectangle(img, (xw1, yw1), (xw2, yw2), (255, 0, 0), 2)
            faces[i, :, :, :] = cv2.resize(frame[yw1:yw2 + 1, xw1:xw2 + 1, :],
(img_size, img_size))
            face =  frame[yw1:yw2 + 1, xw1:xw2 + 1, :]
            face_gray_emo = cv2.cvtColor(face, cv2.COLOR_BGR2GRAY)
            face_gray_emo = cv2.resize(face_gray_emo, (48, 48), interpolation =
cv2.INTER_AREA)
            face_gray_emo = face_gray_emo.astype("float") / 255.0
            face_gray_emo = img_to_array(face_gray_emo)
            face_gray_emo = np.expand_dims(face_gray_emo, axis=0)
            preprocessed_faces_emo.append(face_gray_emo)

        # make a prediction for Age and Gender
        results = model.predict(np.array(faces))
        predicted_genders = results[0]
```

```
        ages = np.arange(0, 101).reshape(101, 1)
        predicted_ages = results[1].dot(ages).flatten()

        # make a prediction for Emotion
        emo_labels = []
        for i, d in enumerate(detected):
            preds = classifier.predict(preprocessed_faces_emo[i])[0]
            emo_labels.append(emotion_classes[preds.argmax()])

        # draw results
        for i, d in enumerate(detected):
            label = "{}, {}, {}".format(int(predicted_ages[i]),
                                        "F" if predicted_genders[i][0] > 0.4 else "M",
emo_labels[i])
            draw_label(frame, (d.left(), d.top()), label)

    cv2.imshow("Emotion Detector", frame)
    if cv2.waitKey(1) == 13: #13 is the Enter Key
        break

cap.release()
cv2.destroyAllWindows()
import logging
import sys
import numpy as np
from tensorflow.keras.models import Model
from tensorflow.keras.layers import Input, Activation, add, Dense, Flatten,
Dropout
from keras.layers.convolutional import Conv2D, AveragePooling2D
```

```
from tensorflow.keras.layers import BatchNormalization

from tensorflow.keras.layers import (
    BatchNormalization, SeparableConv2D, MaxPooling2D, Activation, Flatten,
Dropout, Dense
)
from tensorflow.keras.regularizers import l2
from tensorflow.keras import backend as K

sys.setrecursionlimit(2 ** 20)
np.random.seed(2 ** 10)


class WideResNet:
    def __init__(self, image_size, depth=16, k=8):
        self._depth = depth
        self._k = k
        self._dropout_probability = 0
        self._weight_decay = 0.0005
        self._use_bias = False
        self._weight_init = "he_normal"

        if K.image_data_format() == "channels_first":
            logging.debug("image_dim_ordering = 'th'")
            self._channel_axis = 1
            self._input_shape = (3, image_size, image_size)
        else:
            logging.debug("image_dim_ordering = 'tf'")
            self._channel_axis = -1
```

```
        self._input_shape = (image_size, image_size, 3)

# Wide residual network http://arxiv.org/abs/1605.07146
def _wide_basic(self, n_input_plane, n_output_plane, stride):
    def f(net):
        # format of conv_params:
        #               [ [kernel_size=("kernel width", "kernel height"),
        #               strides="(stride_vertical,stride_horizontal)",
        #               padding="same" or "valid"] ]
        # B(3,3): orignal <<basic>> block
        conv_params = [[3, 3, stride, "same"],
                       [3, 3, (1, 1), "same"]]

        n_bottleneck_plane = n_output_plane

        # Residual block
        for i, v in enumerate(conv_params):
            if i == 0:
                if n_input_plane != n_output_plane:
                    net = BatchNormalization(axis=self._channel_axis)(net)
                    net = Activation("relu")(net)
                    convs = net
                else:
                    convs = BatchNormalization(axis=self._channel_axis)(net)
                    convs = Activation("relu")(convs)

                convs = Conv2D(n_bottleneck_plane, kernel_size=(v[0], v[1]),
                               strides=v[2],
                               padding=v[3],
```

```
                        kernel_initializer=self._weight_init,
                        kernel_regularizer=l2(self._weight_decay),
                        use_bias=self._use_bias)(convs)
    else:
        convs = BatchNormalization(axis=self._channel_axis)(convs)
        convs = Activation("relu")(convs)
        if self._dropout_probability > 0:
            convs = Dropout(self._dropout_probability)(convs)
        convs = Conv2D(n_bottleneck_plane, kernel_size=(v[0], v[1]),
                        strides=v[2],
                        padding=v[3],
                        kernel_initializer=self._weight_init,
                        kernel_regularizer=l2(self._weight_decay),
                        use_bias=self._use_bias)(convs)

# Shortcut Connection: identity function or 1x1 convolutional
# (depends on difference between input & output shape - this
# corresponds to whether we are using the first block in each
#  group; see _layer() ).
if n_input_plane != n_output_plane:
    shortcut = Conv2D(n_output_plane, kernel_size=(1, 1),
                        strides=stride,
                        padding="same",
                        kernel_initializer=self._weight_init,
                        kernel_regularizer=l2(self._weight_decay),
                        use_bias=self._use_bias)(net)
else:
    shortcut = net
```

```
            return add([convs, shortcut])
        return f

    # "Stacking Residual Units on the same stage"
    def _layer(self, block, n_input_plane, n_output_plane, count, stride):
        def f(net):
            net = block(n_input_plane, n_output_plane, stride)(net)
            for i in range(2, int(count + 1)):
                net = block(n_output_plane, n_output_plane, stride=(1, 1))(net)
            return net

        return f

#    def create_model(self):
    def __call__(self):
        logging.debug("Creating model...")

        assert ((self._depth - 4) % 6 == 0)
        n = (self._depth - 4) / 6

        inputs = Input(shape=self._input_shape)

        n_stages = [16, 16 * self._k, 32 * self._k, 64 * self._k]

        conv1 = Conv2D(filters=n_stages[0], kernel_size=(3, 3),
                       strides=(1, 1),
                       padding="same",
                       kernel_initializer=self._weight_init,
                       kernel_regularizer=l2(self._weight_decay),
```

```
                    use_bias=self._use_bias)(inputs)  # "One conv at the
beginning (spatial size: 32x32)"

        # Add wide residual blocks
        block_fn = self._wide_basic
        conv2 = self._layer(block_fn, n_input_plane=n_stages[0],
n_output_plane=n_stages[1], count=n, stride=(1, 1))(conv1)
        conv3 = self._layer(block_fn, n_input_plane=n_stages[1],
n_output_plane=n_stages[2], count=n, stride=(2, 2))(conv2)
        conv4 = self._layer(block_fn, n_input_plane=n_stages[2],
n_output_plane=n_stages[3], count=n, stride=(2, 2))(conv3)
        batch_norm = BatchNormalization(axis=self._channel_axis)(conv4)
        relu = Activation("relu")(batch_norm)

        # Classifier block
        pool = AveragePooling2D(pool_size=(8, 8), strides=(1, 1),
padding="same")(relu)
        flatten = Flatten()(pool)
        predictions_g = Dense(units=2, kernel_initializer=self._weight_init,
use_bias=self._use_bias,
                              kernel_regularizer=l2(self._weight_decay),
activation="softmax",
                              name="pred_gender")(flatten)
        predictions_a = Dense(units=101, kernel_initializer=self._weight_init,
use_bias=self._use_bias,
                              kernel_regularizer=l2(self._weight_decay),
activation="softmax",
                              name="pred_age")(flatten)
        model = Model(inputs=inputs, outputs=[predictions_g, predictions_a])
```

```
        return model

def main():
    model = WideResNet(64)()
    model.summary()

if __name__== '__main__':
    main()
```

APPENDIX 2

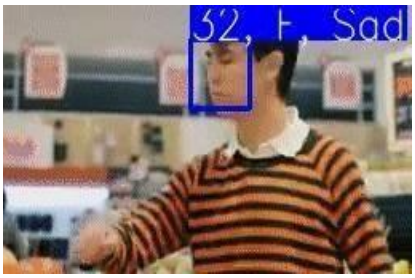


Fig 9.1 Age, sex, emotion – detection

In this picture, an input is taken as video and a single image person is taken as an video input. The input is then compared with the trained model andface, emotion and gender estimation is done in real time.

Fig 9.2 Group of people – age, sex, emotion – detection

The above image is same as the previous image, but this image consists of multiple persons and it captures input from all the people who are in the video and also it detects their age, gender and emotion in real time.

REFERENCES

[1] Pradnya Kedari; Mihir Kapile; Divya Kadole; Sagar Jaikar, “Face Emotion Detection Using Deep Learning”, 2nd International Conference On Advances In Computing, Communication, Embedded And Secure Systems (ACCESS), September 2021

[2] Shan Li; Weihong Deng, “Deep Facial Expression Recognition: A Survey”, IEEE Transactions on Affective Computing, March 2020

[3] Ji-Hae Kim; Byung-gyu Kim; Partha Pratim Roy; Da-mi Jeong, “Efficient Facial Expression Recognition Algorithm Based on Hierarchical Deep Neural Network Structure”, Department of IT Engineering, Sookmyung Women’s University, Seoul, South Korea, March 2019

[4] Catherine Marechal; Dariusz Mikołajewski; Krzysztof Tyburek; Piotr Prokopowicz; Lamine Bougueroua; Corinne Ancourt; Katarzyna Węgrzyn-Wolska, “Survey on AI-Based Multimodal Methods for Emotion Detection”, High-Performance Modeling and Simulation for Big Data Applications, March 2019

[5] Wafa Mellouk; Wahida Handouzi, “Facial emotion recognition using deep learning: review and insights”, The 2nd International Workshop on the Future of Internet of Everything (FIoE), Belgium, Aug 2020

[6] Rajeev Ranjan; Swami Sankaranarayana Ankan Bansal; Navaneeth Bodla, “Deep Learning for Understanding Faces: Machines May Be Just as Good, or Better than Humans”, IEEE Signal Processing Magazine, Volume: 35, Issue: 1, Jan. 2018,

[7] Ta-Te Lu; Sheng-Cheng Yeh; Chia-Hui Wang; Min-Rou Wei, “Cost-effective real-time recognition for human emotion-age-gender using deep learning with normalized facial cropping preprocess”, Multimedia Tools and Applications, March 2021,

[8] Rasha Ragheb Atallah; Amirrudin Kamsin; Maizatul Akmar Ismail; Sherin Ali Abdelrahman; Saber Zerdoumi, “Face Recognition and Age Estimation Implications of Changes in Facial Features: A Critical Review Study”, Universiti Sains Malaysia, Penang, June 2018,

[9] Pipit Utami; Rudy Hartanto; Indah Soesanti, “A Study on Facial Expression Recognition in Assessing Teaching
Skills: Datasets and Methods”, The Fifth Information Systems International Conference, 2019

[10] Alice Othmania; Abdul Rahman Taleb; Hazem Abdelkawy; Abdenour Hadid, “Age estimation from faces using deep learning: a comparative analysis”, Computer Vision and Image Understanding, Volume 196, 102961, July 2020