A Project Report

Emotion Recognition System using CNN & RNN

Submitted in partial fulfilment of the requirements for the award of degree

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

by

Jahnavi Jonnalagadda (160118733064)
Siri Chandana Desiraju (160118733080)

Department of Computer Science and Engineering,
Chaitanya Bharathi Institute of Technology (Autonomous)
(Affiliated to Osmania University, Hyderabad)
Hyderabad, TELANGANA (INDIA) –500 075
[2021-2022]

CERTIFICATE

Certified that project work entitled "Emotion Recognition System using CNN & RNN" is the bonafide work carried out by, Jahnavi Jonnalagadda (160118733064) and Siri Chandana Desiraju (160118733080), in partial fulfilment for the award of Bachelor of Engineering in Computer Science and Engineering from Chaitanya Bharathi Institute of Technology (A), Gandipet during the academic year 2021-2022.

Supervisor,

Dr. Rupesh Mishra
Associate Professor

Head, CSE Dept

Prof Y. Ramadevi
Professor

Place: CBIT, Hyderabad
Date:

DECLARATION

We hereby declare that the project entitled "**Emotion Recognition System using CNN & RNN**" submitted for the B.E (CSE) degree is my original work and the seminar has not formed the basis for the award of any other degree, diploma, fellowship or any other similar titles.

Signature of the Students

Jahnavi Jonnalagadda
(160118733064)

Siri Chandana Desiraju
(160118733080)

Place: CBIT, Hyderabad
Date:

ABSTRACT

Human emotions play an important role in interpersonal relationships. The recognition of facial emotions has been an active research topic for many years. As a result, there are several advances being made in this field. Emotions are reflected from speech, hand and gestures of the body and through facial expressions.

Our aim is to identify the emotion of the facial image provided. We aspire to classify these emotions into five categories - happiness, sadness, anger, disgust and surprise. This research plays a significant role in the field of feedback systems, face unlocking, shopping malls to predict unusual activities such as robbery as well as law enforcement.

In this project, we build a VGG Model (a variant of Convolutional Neural Network) for detecting the emotions. Further training of the model efficiently so that it can recognize the emotion and testing of the model in real-time using webcam.

ACKNOWLEDGMENT

We would like to express our sincere thanks to Prof P. Ravinder Reddy, Principal, CBIT, for providing the working facilities in college.

We wish to express our sincere thanks and gratitude to Prof Y Rama Devi, Professor and HOD, Department of CSE, CBIT.

We are extremely thankful to Dr. T. Sridevi, Associate Professor and Smt. E. Kalpana, Assistant Professor, Department of CSE, CBIT for their encouragement and support throughout the project.

We are extremely thankful and indebted to our Internal guide Dr. T. Sridevi, Associate Professor, Department of CSE, for her constant guidance, encouragement and moral support throughout the project.

Finally, we would also like to thank all the faculty and staff of CSE Department who helped us directly or indirectly, for completing this project.

Table of Contents

LIST OF FIGURES

TIMELINE

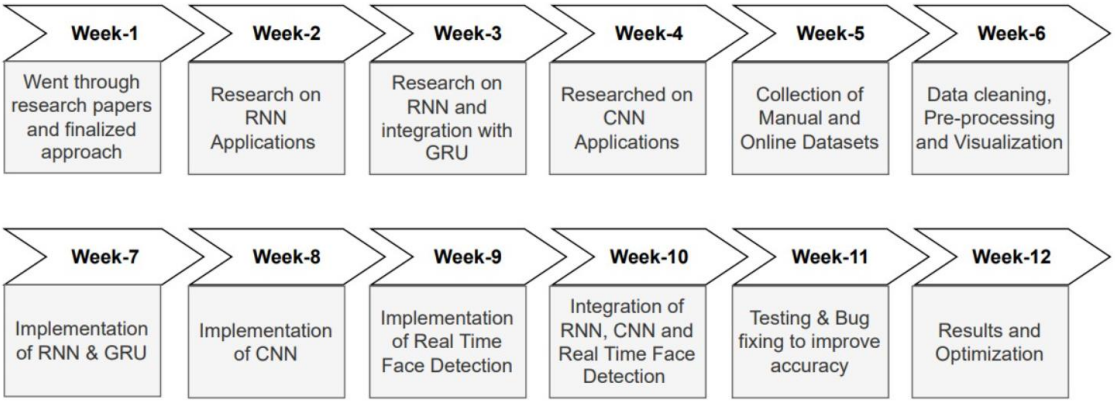


Figure 1Project Timeline

1. INTRODUCTION

This chapter will contain the objective of the project and its significance. This chapter will also serve the purpose of outlining the results and describing the scope of the project, while giving a brief description on what is to be expected in the upcoming sections.

1.1 Problem Definition including the significance and objective

One of the important ways humans display emotions is through facial expressions. Facial Expression Recognition is one of the most powerful, natural and immediate means for human beings to communicate their emotions and intensions. It is the process of detecting human emotions from facial expressions. Humans can be in some circumstances restricted from showing their emotions, such as hospitalized patients, or due to deficiencies. Hence, better recognition of other human emotions will lead to effective communication.

In social situations, humans will naturally express their personal emotions. An accurate understanding of each other's emotions will help build mutual understanding and trust. The expression and understanding of emotions is an essential skill for humans. We mainly convey personal emotions in three ways, namely language, voice and facial expressions. Scholars have found that facial expressions are the most important way of expressing human emotion information.

Understanding contextual emotion has widespread consequences for society and business. In the public sphere, governmental organizations could make good use of the ability to detect emotions like guilt, fear, and uncertainty.

Facial expression recognition system is a computer-based technology and therefore, it uses algorithms to instantaneously detect faces, code facial expressions, and recognize emotional states. It does this by analyzing faces in images or video through computer powered cameras embedded in laptops, mobile phones, and digital signage systems, or cameras that are mounted onto computer screens.

1.2 Methodologies

Neural Nets have a critical benefit that's immensely helpful in emotion recognition: they do feature engineering automatically. Convolutional Neural Networks (CNNs) are very effective for the use of images as inputs. These networks further feature engineer the input images and can help achieve greater accuracy in emotion recognition by finding features. Recurrent Neural Networks use these features to classify the facial expression. Recurrent Networks are able to robustly derive information bi-directionally from sequences of data vectors by exploiting the fact that data represented by successive vectors is also connected semantically and therefore interdependent.

1.3 Objectives

The purpose of the resulting application is to help humans display emotions through facial expressions which are being captured through real time web camera.
The application recognizes emotions like angry, disgust, scared, happy, sad, surprised, neutral. The application can be run on any of the system with python installed, given that it has a webcam. This application also helps hospitalized patients or people who cannot speak, they can express their intensions through facial expressions.

1.4 Scope of the Project

In this project, we will create an application that will provide the following functionalities:

1) Real-time image capturing
2) Face Detection
3) Emotion Classification
4) Emotion Recognition - angry, disgust, scared, happy, sad, surprised, neutral
5) Display multiple emotions probability
6) Display emotion output
7) Display accuracy of the model

By emotion recognition from images captured by real time web camera. As mentioned in multiple sections above, the principal objective of this project is to help humans display emotions through facial expressions.

1.5 Organization of the Report

This report will clearly explain the terminologies used, tools used, proposed implementations, results of the implementations, and conclusions drawn from the results. After the introduction, Chapter 2, the literature survey, will introduce the problem in detail. It will go over the necessary domain terminology, mention the tools, and discuss how they will be used. We will then go over the existing solutions that have already been presented along with any other related works.

Chapter 3 will discuss the design of the proposed system. We will look into the details of how exactly we implement this tool and how this tool is run and implemented differently from the existing solutions. Block diagrams, implementation architectures, module descriptions, and any other relevant descriptions of the tool's design will be clearly presented in this chapter.

Next, Chapter 4 will show the implementation of the proposed system. This chapter will also depict the implementation of the tool through flowcharts, DFDs, ER Diagrams, etc. Any algorithms or pseudo code will also be presented in this chapter. Any data sets would also be presented and described in this chapter.

Chapters 5 and 6 are very similar. In chapter 5, we will discuss the results of the implementation. Finally, in chapter 6, we'll conclude, based on our implementation and results. Finally, after observing all the results and drawing conclusions, we present our recommendations on how to use this tool, who should use this tool, what scenarios is it most effective in, how we can improve it, and so on. The future works and scopes of this project will also be discussed in this chapter.

2. LITERATURE SURVERY

2.1 Introduction to the problem domain terminology

Visual Emotion Classification

The visual system is the primary mode of perception for the way people receive emotional information. Emotional cues can be in the form of facial expressions. While the visual system is the means by which emotional information is gathered, it is the cognitive interpretation and evaluation of this information that assigns it emotional value. Visual Emotion Classification can be understood and categorized by analyzing facial expressions, vocal tones, gestures, and physiological signals.

Convolutional Neural Network

Convolutional Neural Network is a class of deep neural networks, most commonly applied to analyze visual imagery. CNN is composed of multiple layers of artificial neurons. CNNs have fundamentally changed our approach towards image recognition as they can detect patterns and make sense of them. They are considered the most effective architecture for image classification, retrieval and detection tasks as the accuracy of their results is very high.

Dual Convolutional Neural Network

Dual CNN extracts different levels of stimuli features from the local attribute to global semantic. Dual CNN jointly estimates the structures and details. A Dual CNN consists of two branches, one shallow sub-network to estimate the structures and one deep sub-network to estimate the details. The modular design of a Dual CNN makes it a flexible framework for a variety of low-level vision problems. It extracts the characteristics of power equipment through two independent CNN models. The Dual CNN is a flexible framework for low-level vision tasks and can be easily incorporated into existing CNNs.

Multi-Task Learning

Multi-Task learning is a sub-field of Machine Learning that aims to solve multiple different tasks at the same time, by taking advantage of the similarities between different

tasks. This approach in which we try to learn multiple tasks simultaneously, optimizing multiple loss functions at once. Rather than training independent models for each task, we allow a single model to learn to complete all of the tasks at once. Multi-task learning improves performance when there are underlying principles or information shared between tasks.

Gated Recurrent Unit

GRUs are improved version of standard recurrent neural network. GRU uses, so-called, update gate and reset gate. Basically, these are two vectors which decide what information should be passed to the output. The special thing about them is that they can be trained to keep information from long ago, without washing it through time or remove information which is irrelevant to the prediction. Each recurrent unit as Gated Recurrent Unit networks consist of gates which modulate the current input and the previous hidden state.

Recurrent Neural Networks

RNNs are a powerful and robust type of neural network, and belong to the most promising algorithms in use because it is the only one with an internal memory. Because of their internal memory, RNN's can remember important things about the input they received, which allows them to be very precise in predicting what's coming next. Recurrent neural networks can form a much deeper understanding of a sequence and its context. In a RNN the information cycles through a loop. When it makes a decision, it considers the current input and also what it has learned from the inputs it received previously.

2.2 Existing Solutions

In the literature of visual emotion classification, low-level visual stimuli, such as shape or color, were first utilized to classify emotions. Then, many researchers used middle-level visual stimuli, such as texture or composition, for emotion classification. Recently, some studies have shown that high-level semantic information in images plays an important role in emotion recognition. However, many previous works predominantly have ignored

the dependencies between low-level and high-level features, which has degraded recognition accuracy.

Plenty of studies have been proposed to comprehend the psychological meaning surrounding different levels of representations learned by CNNs and validate this assumption in some of those works. The results of these works proved that feature representations of different layers of deep models prefer different levels of visual stimulus.

Some existing works try to fuse the feature by reducing the dimensionality from the concatenated vectors, or exploiting the max aggregation function, or a CNN-RNN combination. However, existing fusion models do not exploit the dependencies among different levels of stimulus.

This paper proposes minimal image pre-processing steps and Convolutional Neural Network (CNN). Seven facial emotions (happy, anger, sadness, disgust, neutral, fear, surprise) are detected. The approach used detects faces from input images based on Haar Cascades method, which uses the Adaboost learning algorithm and provides effective result of classifiers by choosing significant features over a large set. Further Image Augmentation (rotation, shifts, shear, and zoom, flip) and CNN with four layers is implemented producing maximum accuracy. [1]

This paper proposes a Deep Learning based framework for human emotion recognition. The proposed framework uses the Gabor filters for feature extraction and then a CNN for classification. The methodology is to first apply a Gabor filter for texture analysis, edge detection and feature extraction to the images and then convey the output results as inputs to the convolutional neural network, returning a vector with seven modes - angry, disgust, fear, happy, neutral hence improving the accuracy. [2]

This paper proposes a method for real time emotion recognition captures real-time images and detects the face using Local Binary Patterns (LBP) cascade classifier. Pre-

processing of the image - cropping, resizing, and normalisation are incorporated. This system implements a CNN for classification of emotion using the features, performing feature extraction is by combining the features extracted by CNN with Histogram oriented gradients (HOG) and facial landmarks improved accuracy. [3]

This paper proposes Facial Expression Recognition based on valence dimension prediction CNN. This system includes face detection, feature extraction and valence grade prediction. Image pre-processing is performed which includes valence dimension annotation and face detection of the image. CNN network model is designed based on the Keras Deep Learning framework to predict the valence dimension of facial expression and verify the performance by recognizing the facial expressions. [4]

This paper proposes a methodology which aims to perform real-time facial expression recognition using CNN. The method is divided into three steps - obtain and input the images to the CNN architecture for recognition, obtain the results of facial expression recognition, average the recognition result with the previous output and re-output. In this way, real-time recognition can be achieved through integrating the CNN architecture and the webcam. Moreover, overall robustness and accuracy of facial expression recognition are greatly improved by using the proposed method. [5]

The paper proposes using Supervised Machine Learning Algorithms for the recognition of facial emotion; Skybiometry and AffactNet have been employed. Skybiometry is a state of the art Face recognition & Face detection cloud biometrics API allowing developers and marketers to do more with less. It evaluates 68 features on the face including geometric and appearance features that are used to represent facial emotions. These images are classified by AffectNet according to 5 emotions (happy, sad, surprise, angry, scared) these serve as the datasets. [6]

The paper proposes image pre-processing steps and different Convolutional Neural Network (CNN) structures providing better accuracy and greatly improved training time. Seven basic emotions of human faces: sadness, happiness, disgust, anger, fear, surprise

and neutral, were to be predicted. The approach used the special 3-D architecture of CNN and incorporating appropriate image pre-processing steps such as Converting colour image to grayscale, Synthetic Image Generation(rotation or translation) , Image Cropping, Down sampling(reducing the amount of data used without losing quality). Maximum accuracy was obtained with five convolution layers. [7]

This paper proposes a model to analyse physiological signals (such as heart-beat rate, re8spiratory rate) to determine if the individual is pleasant, neutral, unpleasant. We apply one-dimensional Convolution Neural Network (1D-CNN) to every channel of peripheral signals and then concatenate results for classification. [8]

This paper provides a six step process including input image, training data, template library, feature extraction, comparison and output result. It applied four different architectures: GoogleNet, ResNet, VGGNet and AlexNet and examined them on the same database. AlexNet achieved the best overall accuracy using five convolutional layers and three fully connected layers, proving the validity of CNN in the complex model. [9]

2.3 Related Works

Existing works on emotion recognition can be roughly grouped into two categories: single-level feature-based approaches and multi-level feature-based approaches. The single-level features can be further categorized into low-level, mid-level, and high-level features, and most of them are handcrafted.

Extraction of mid-level features based on affective audio-visual words and proposed a latent topic driving model for the video classification task. A method was proposed to infer emotions based on the understanding of visual concepts on a large-scale visual sentiment.

High-level features related to object detection were introduced for visual sentiment

analysis in the work. More recently, due to the complexity of emotions, more and more works have been proposed to exploit multiple levels of features. In terms of model structures, both the CNN and recurrent network have been widely adopted recently. For scene recognition and object detection, CNN-based methods have also been employed in image emotion analysis.

To fine tune the pre-trained CNN on ImageNet and showed that the CNN model outperforms previous methods relying on different levels of handcrafted features. Combining the CNN model with the support vector machine (SVM) to detect image emotion on a large-scale dataset of web images. On utilized three different networks to capture different levels of visual features with high expense of parameters. However, previous works did not explicitly exploit the dependency between low-level features and high-level features, and most of these works lack effective fusion methods for different levels of features. Second, RNNs can effectively model the long-term dependency in sequential data. It has been widely applied in many tasks, including image understanding, machine translation, sequential modeling, and so on. In other works, recurrent networks with hierarchical structures were adopted to generate images of different scales in a coarse-to-fine fashion.

This paper proposes a combination of recurrent neural network (RNN) and 3D convolutional networks (C3D). RNN takes appearance features extracted by convolutional neural network (CNN) over individual video frames as input and encodes motion later, while C3D models appearance and motion of video simultaneously. [10]

This paper proposes a hierarchical convolutional neural network (CNN)-recurrent neural network (RNN) approach to predict the emotion. First, dual CNN is introduced, to learn different levels of stimuli features from the local to global that can enhance the representation ability of high-level semantics. Further, a multi-task loss is designed to meet both a classification and contrastive objective to learn more discriminative features. Then, it is proposed that the stacked bi-directional RNN method for different levels of representation aggregation, where two directional sequential features are fed into the

model for exploring the dependency among different levels of features in visual emotion recognition. [11]

This paper proposes Emotion Recognition based on facial expressions using CNN with four convolutional layers which include three main steps - pre-processing, deep feature learning and deep feature classification pre-processing the image. Usually it is noticed that this phase includes face detection and image cropping using mostly the Viola-John algorithm (with Haar Cascade), converting to grayscale and resize the images, face alignment and augmentation (scaling, rotating, colours, noises) producing maximum accuracy and best results. [12]

This paper proposes a model for the identification/detection of compound emotion categories, i.e. combination of two basic emotions, for example: Happily Surprised, Sadly Surprised, etc. It uses a highway layers which offers assistance to transmit the error straightforwardly to the top of the given CNN. This improves the execution forward against typical CNNs because it provides with a direct as well as indirect mistake correction strategy, which helps train the network better. [13]

This paper proposes recognition of human facial expressions through a deep learning approach using Convolutional Neural Network CNN algorithm with five layers. The pre-training phase involves a face detection subsystem with noise removal, including feature extraction which aims to find the most fitting representation of facial images. The generated classification model used for prediction can identify seven emotions providing improved accuracy with best results. [14]

3. DESIGN OF PROPOSED SYSTEM

This chapter contains the description of the proposed system and diagrams to support the descriptions.

3.1 Block Diagrams

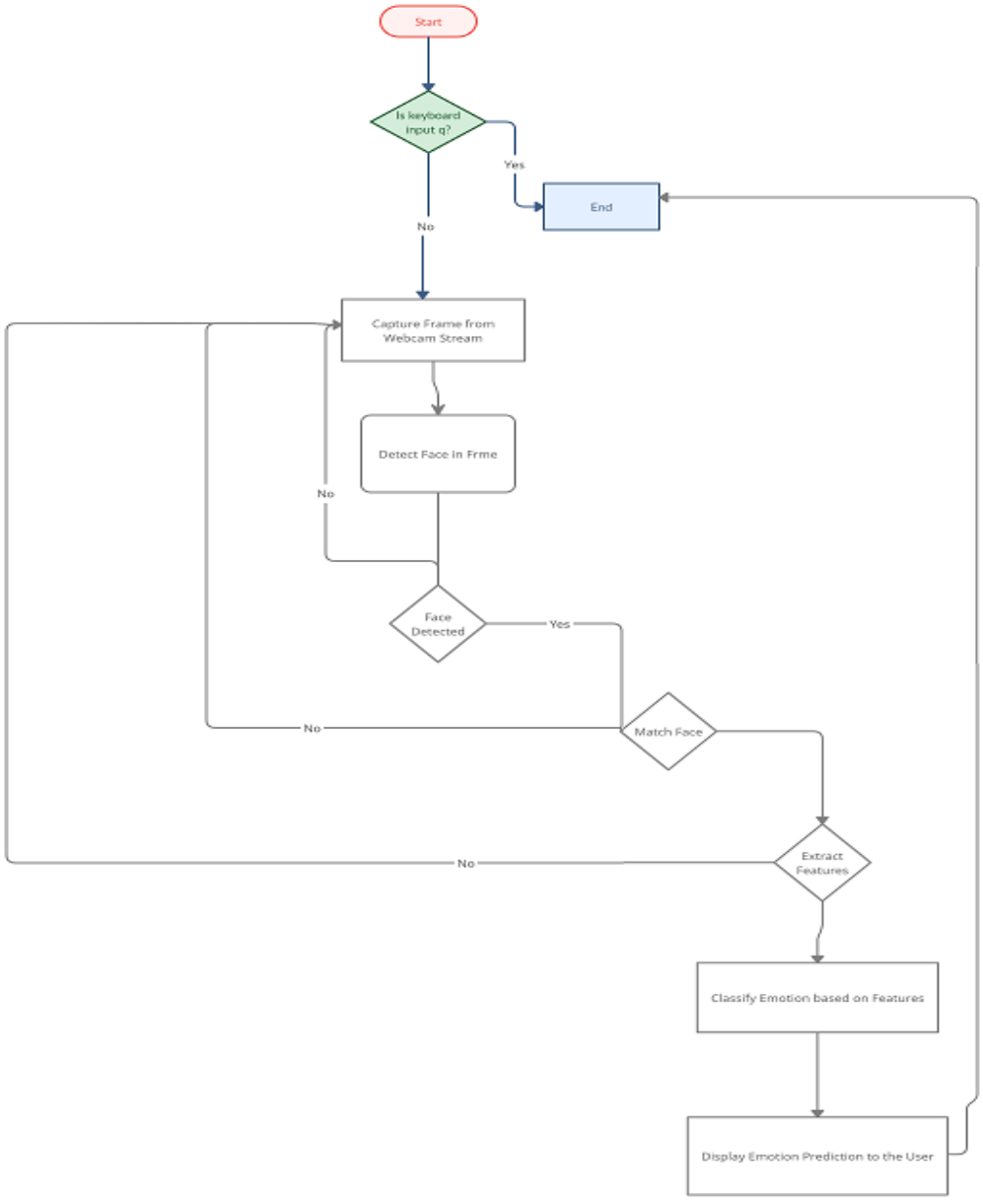


Figure 2: Block Diagram for the Proposed System

3.2 Module Description

Data Collection

Data Collection for Emotion Recognition helps in analyzing and comparing an individual's facial details. An important step in the entire process is face detection. It locates as well as detects human faces in videos as well as images.

The data collected for this application consists of 48×48 pixel gray scale images of faces. The faces have been automatically registered so that the face is more or less centered and occupies about the same amount of space in each image. The task is to categorize each face based on the emotion shown in the facial expression in to one of seven categories (0=Angry, 1=Disgust, 2=Fear, 3=Happy, 4=Sad, 5=Surprise, 6=Neutral).

The training set consists of 35,888 examples. train.csv contains two columns, "emotion" and "pixels". The "emotion" column contains a numeric code ranging from 0 to 6, inclusive, for the emotion that is present in the image. The "pixels" column contains a string surrounded in quotes for each image. The contents of this string a space-separated pixel values in row major order.

Data Augmentation

Data augmentation is a set of techniques for producing additional data points from existing data to artificially enhance the amount of data. Making modest adjustments to data or utilising deep learning models to produce additional data points are examples of this. By creating fresh and varied instances to train datasets, data augmentation can help improve the performance and results of
machine learning models. A machine learning model works better and more correctly when the dataset is rich and sufficient.

Data collection and labelling can be time-consuming and costly for machine learning models. Companies can lower these operational costs by transforming datasets using data augmentation techniques.

Training

The proposed work was trained on CNN and RNN networks using real time web camera and capturing images. We also used dataset consisting 35,888 images. The images captured from real time web camera and resized to proper face boxes that have been automatically registered so that the face is more or less centered and occupies about the same amount of space in each image.

Testing and Accuracy

The models are tested and their respective accuracies are calculated. The models are also tested with the real-time web camera image collection and thus validated the performance of the models.

Visualization

We can observe how different emotions are displayed on the image and categorized. The probability of different emotions are displayed to classify one emotion from the other. A bar graph is displayed in real time which displays the weightage of the emotion of each image on the face.

4. IMPLEMENTATION OF THE PROPOSED SYSTEM

This chapter aims to describe the implementation. It describes the architecture of various components of the project through apt depictions and description of related terms. It also presents the code of the project.

4.1 Architecture

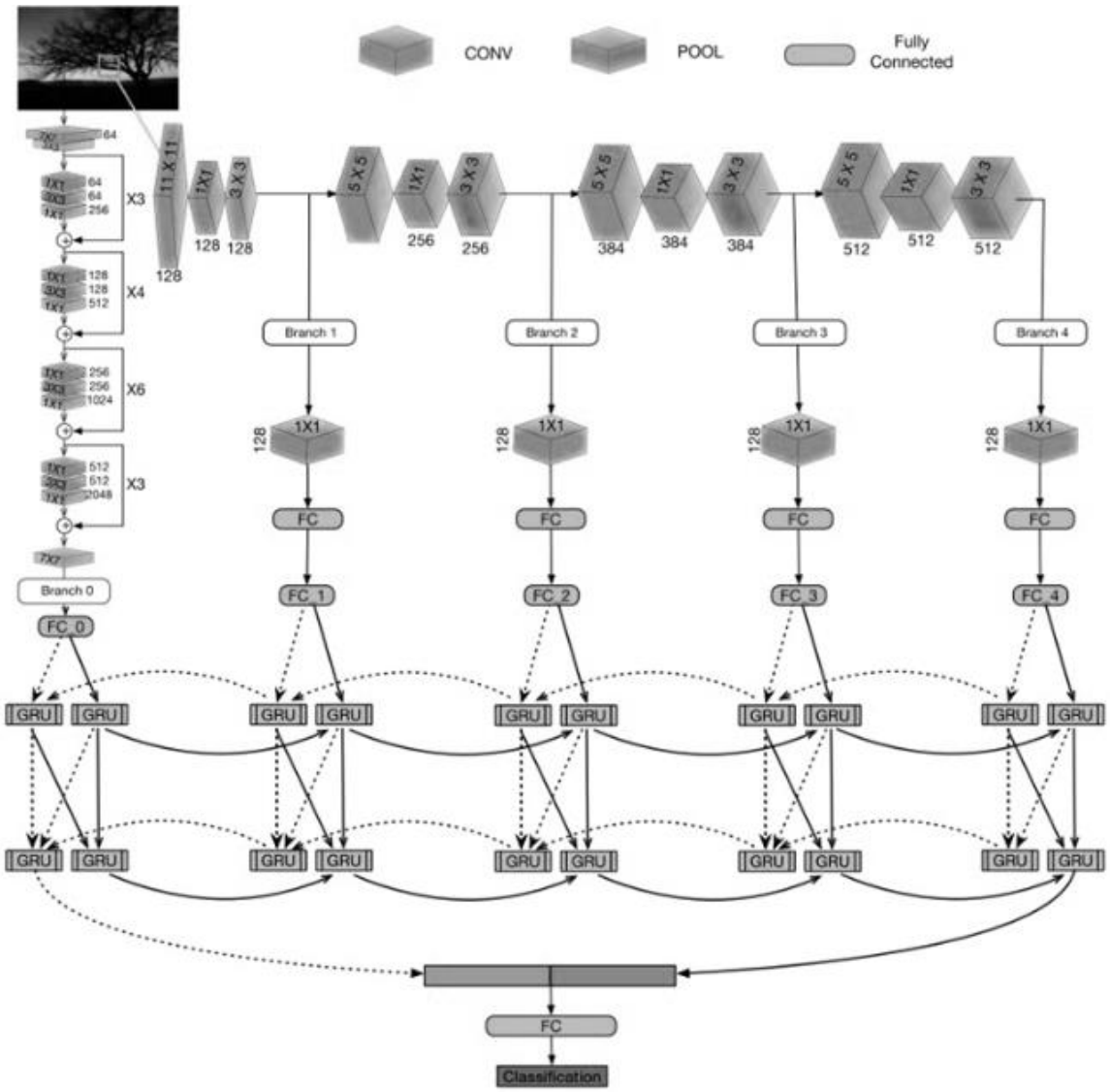


Figure 3 Hierarchical CNN-RNN

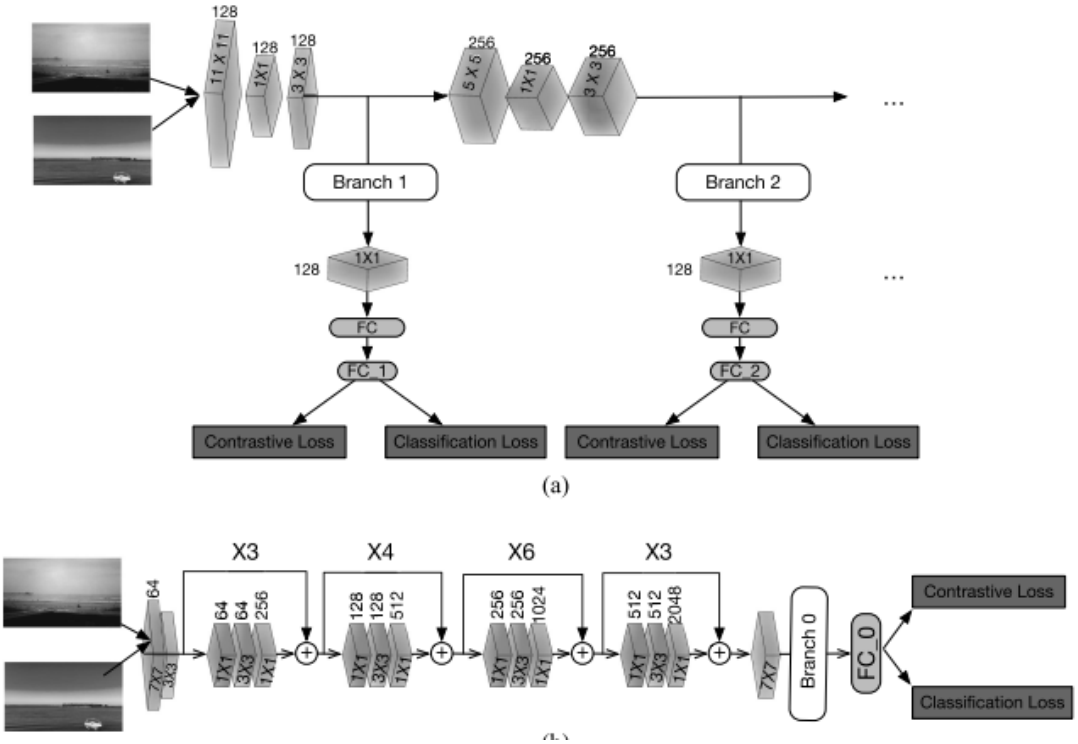


Figure 4 : CNN Architecture of the Proposed System

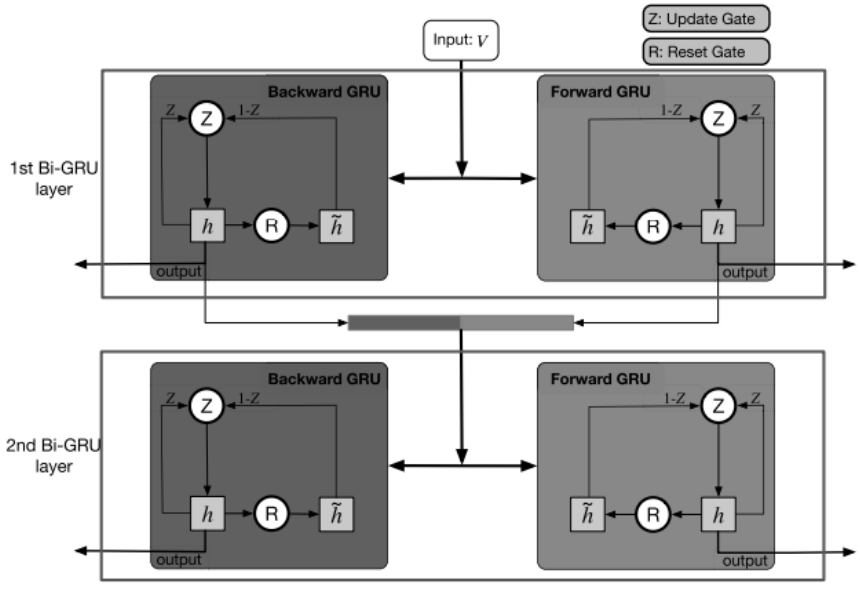


Figure 5 : RNN Architecture of the Proposed System Using GRU

4.2 UML Diagrams

ER Diagram

The below figure depicting the proposed system demonstrates the Entity Relationship (ER) diagram. In this system, User and Face Detector are the entities consider. The User entity has four attributes namely SNo, User_ID, Name and Face_ID. The Face Detector entity has two attributes namely Face Recognition and Emotion Recognition. The two entities User and Face Detector are combined with the relation detect or recognize. So, in this system, once the user entity gives its face as input image, the face detector entity detects the face of the user and recognizes the emotion.

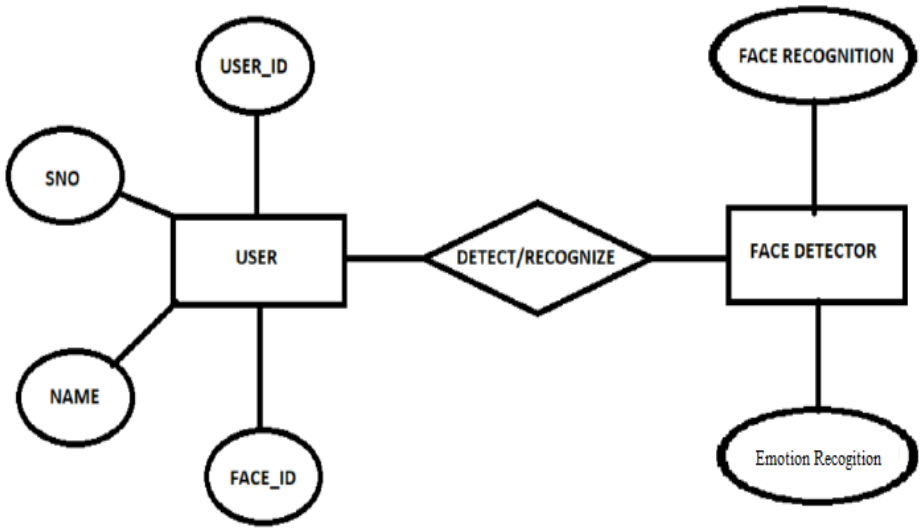


Figure 6 ER Diagram

Data Flow Diagram

The below figure depicting the proposed system demonstrates the Data Flow diagram. The user performs an emotion through image, which is captured by web camera. Initially

the model resizes the input image in such a way that only face is considered and highlighted. This input is given to Convolutional Neural Network (CNN) to perform the pre-processing and feature extraction. The results of this model is given to Recurrent Neural Network (RNN) which performs bi-directional feedback analysis on the features extracted. Based on the output, as a last step emotion is detected.

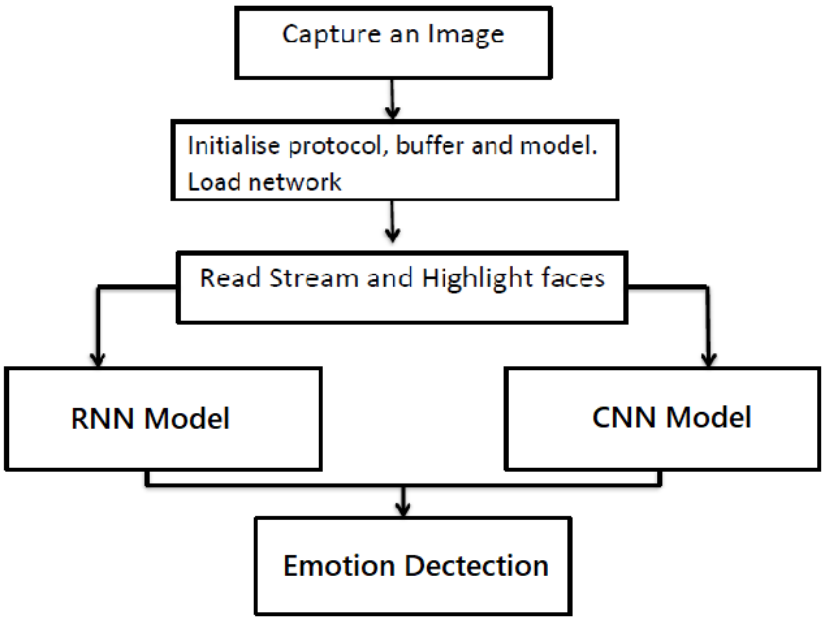


Figure 7 Data Flow Diagram

Data Flow Diagram Level 0

The below figure depicting the proposed system demonstrates the Data Flow diagram of Level 0. This diagram gives an abstract view of the Emotion Recognition system and considers the entire system as a single process. The user acts as the input where image with emotion is given and the captured image is given to CNN and RNN models for processing and feature extraction, and detecting the emotion. The emotion detected is given back to the user as output.

Figure 8 DFD Level 0

Data Flow Diagram Level 1

The below figure depicting the proposed system demonstrates the Data Flow diagram of Level 1. This diagram gives a broad overview of the Emotion Recognition system and breaks down the system into sub processes. The user gives an image with emotion as input to the CNN model which performs pre-processing and extraction of features. These features are given to the RNN model to perform the bi-directional analysis of the features and get their dependencies. With this feature analysis, it reconizes the emotion and gives back to the user as output.

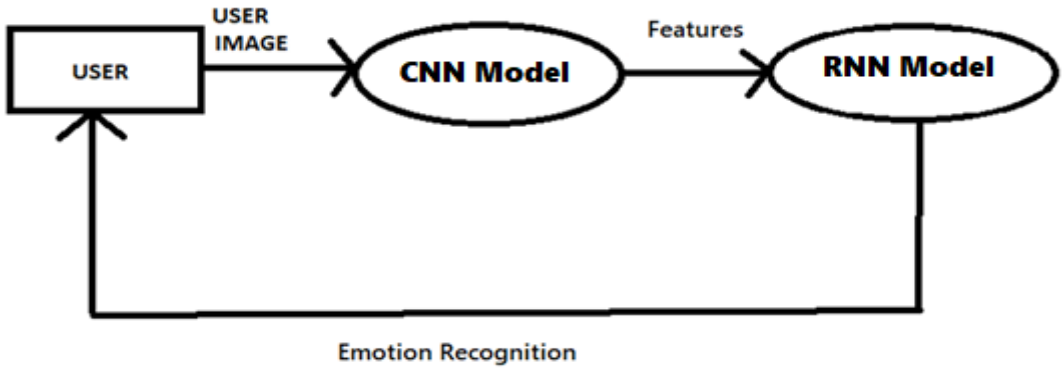


Figure 9 DFD Level 1

Data Flow Diagram Level 2

The below figure depicting the proposed system demonstrates the Data Flow diagram of Level 2. This diagram gives overall scenario of the Emotion Recognition system. The user gives the user image as input to the buffer model which initializes the mean values and ranges to detect faces. These detected faces are given as input to CNN model which does pre-processing and extracts the features. These features are given to RNN model to get further dependencies among the features and detection the emotion. This detected emotion is given as output to the user.

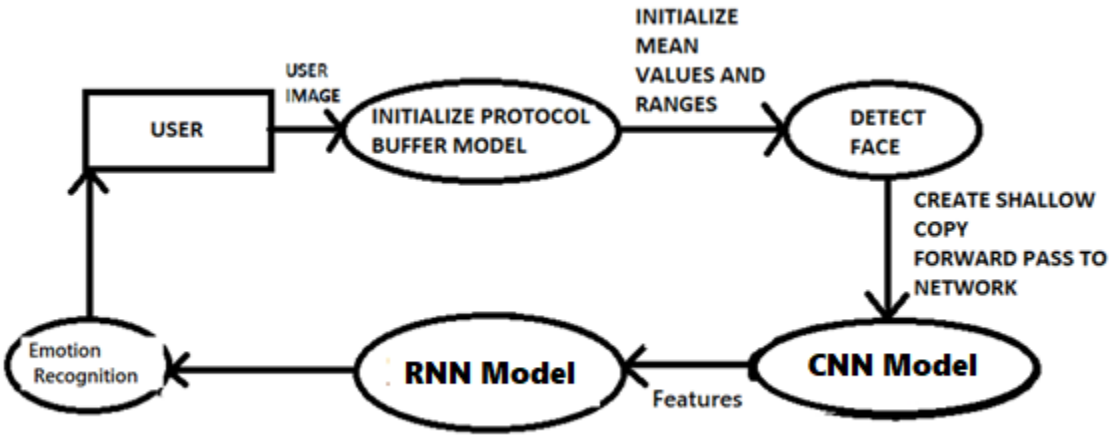


Figure 10 DFD Level 2

Sequence Diagram

The below figure depicting the proposed system demonstrates the Sequence Diagram. This diagram gives the interaction and operations carried out over the Emotion Recognition system. A video stream is being captured from the user and specific image is being extracted and used for face detection. The buffer model is initialized to perform the face detection and readNet() loads the networks for CNN and RNN model to do the pre-

processing and featue extraction. highlightFace() identifies the faces in the form of face boxes from the image and inputs it to the CNN model. This model extracts relevant features and passes them to RNN model which does further analysis by getting the dependencies between the features. It finally returns the predicted emotion as output to the user.

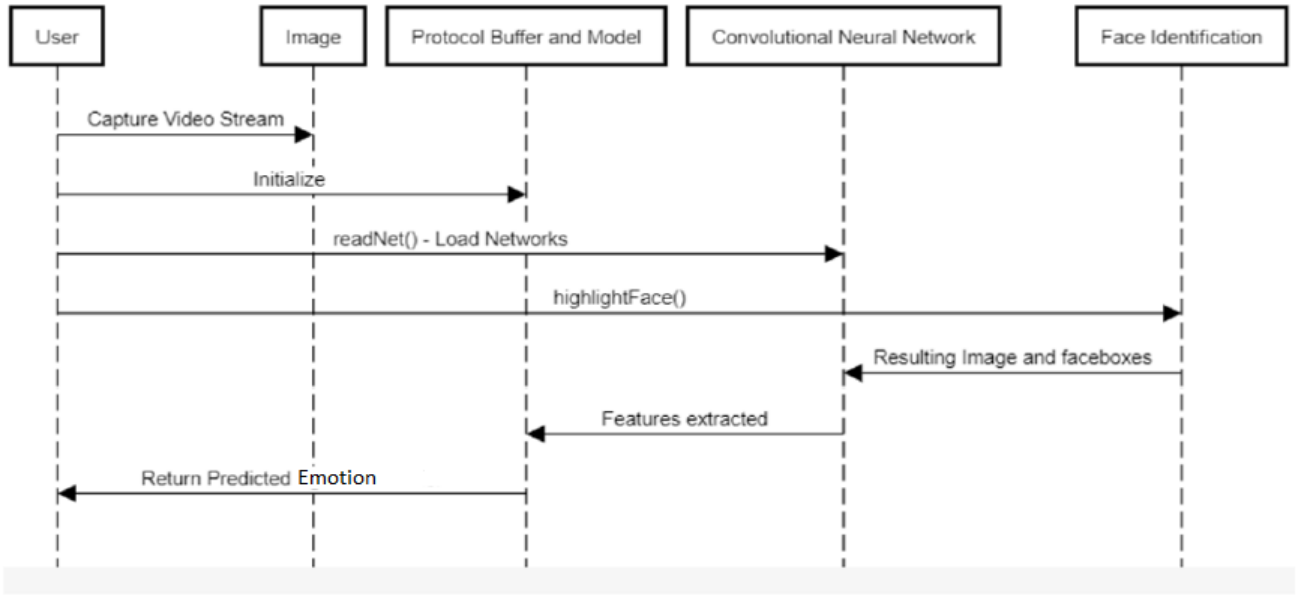


Figure 11 Sequence Diagram

Collaboration Diagram:

The below figure depicting the proposed system demonstrates the Collaboration Diagram. This diagram gives the interactions and behaviour among different objects in the system. A real-time video stream is captured using a webcam and image this image is considered for face detection. The faces are given to the CNN model once the networks are loaded. Here pre-processing and feature extraction is done, these features are given as input to RNN for depecdencies and further feature analysis. The result emotion is

generated as output to user.

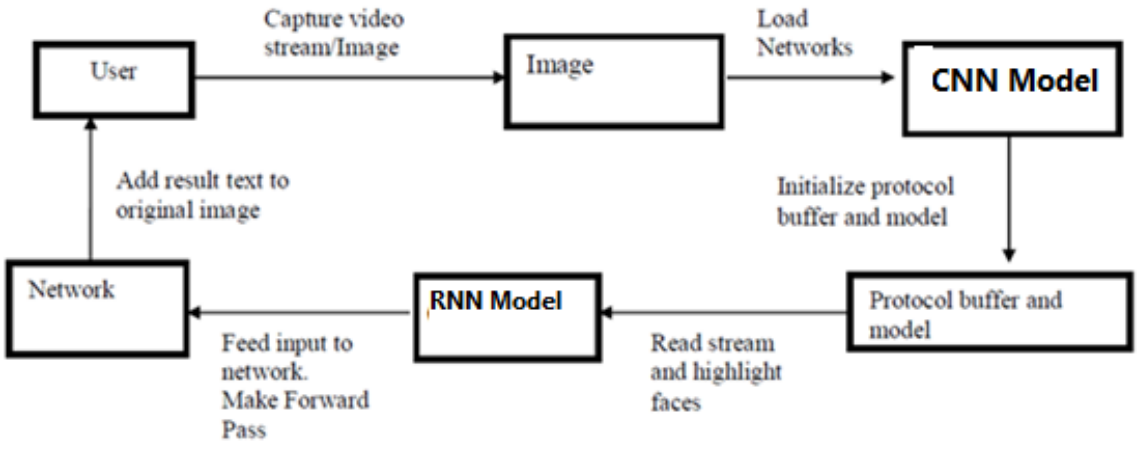


Figure 12 Collaboration Diagram

Use Case Diagram:

The below figure depicting the proposed system demonstrates the Collaboration Diagram. This diagram gives details about system-user interactions in the Emotion Recognition system. The actors in this system are the users and the goal of the system is to detect the emotion of the image. A real-time video stream is captured using a webcam and image this image is considered for face detection. The faces are given to the CNN model once the networks are loaded. Here pre-processing and feature extraction is done, these features are given as input to RNN for depecdencies and further feature analysis. The result emotion is generated as output to user which is the goal.

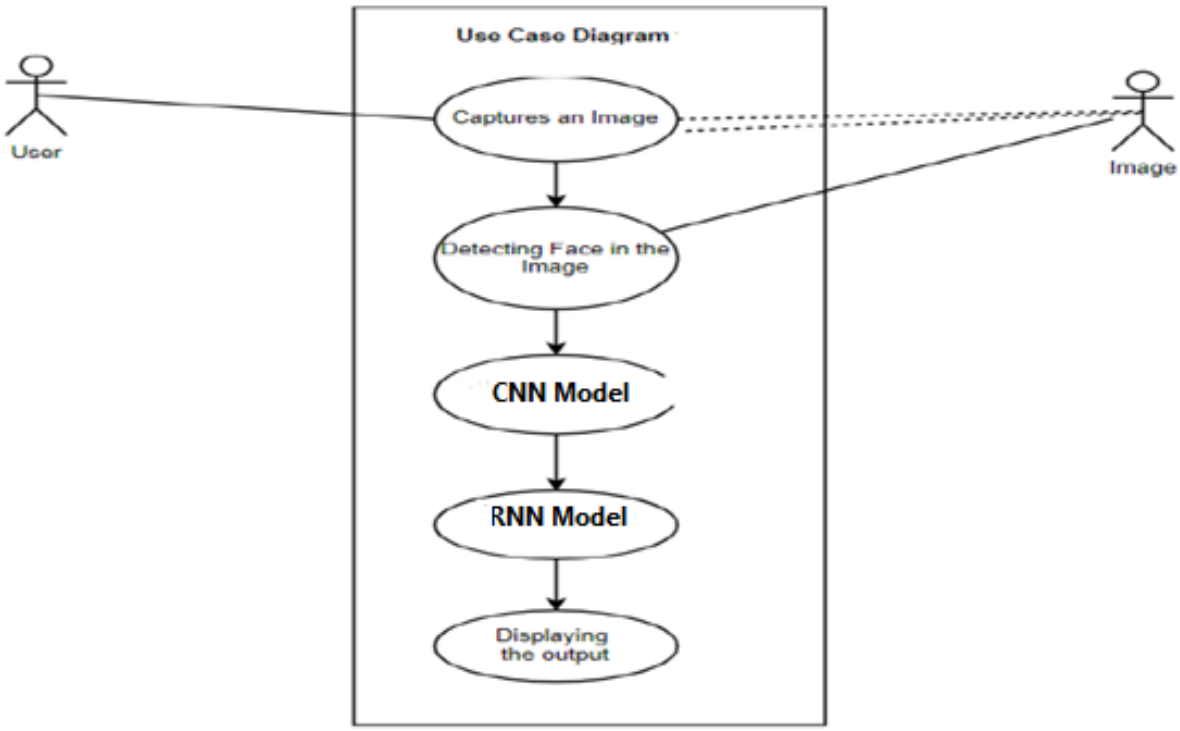


Figure 13 Use Case Diagram

Class Diagram:

The below figure depicting the proposed system demonstrates the Class Diagram. This diagram gives details about different classes and attributes used in the Emotion Recognition system. The classes considered in this system are User, Camera, Emotion and Output. The attributes considered in User class are userid, username, userfaceid, serialnumber and methods are AddRecords, UpdateRecords, DeleteRecords, StoringRecords. The attributes considered in Camera class are id, name, faceid and methods are ImageProcessing. The attributes considered in Emotion class are imageheight, imagewidth, classes and depth and methods are EmotionPrediction. The attributes of output class are predictedemotionlabel, predictedemogroup and methods are DisplayEmotion. This way interactions are displayed between classes of system.

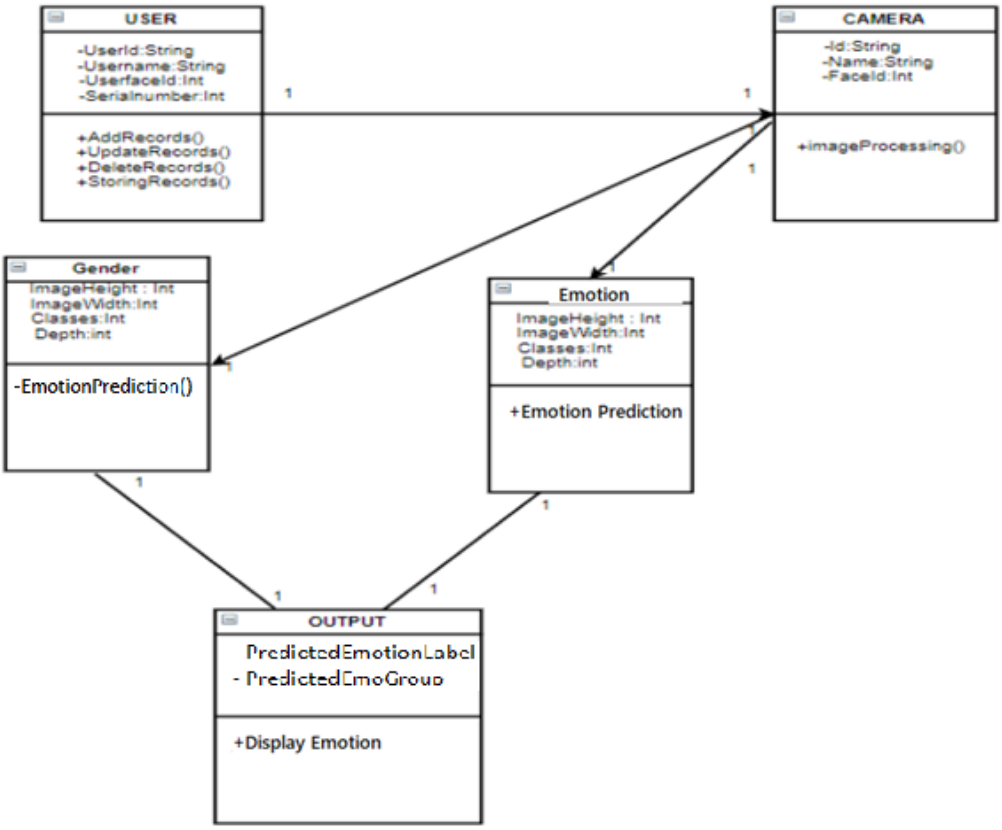


Figure 14 Class Diagram

Activity Diagram

The below figure depicting the proposed system demonstrates the Activity Diagram. This diagram will give the behavioral picture and the conrol flow of the Emotion Recognition system. In this system, different activities include Capturing of image, detecting the face of the image, emotion analysis and displaying the emotion output. There is a conditional check on whether the face is detected or not. The start of the process is capturing the image and termination of the process is displaying the output emotion. The process also terminates if the face is not detected.

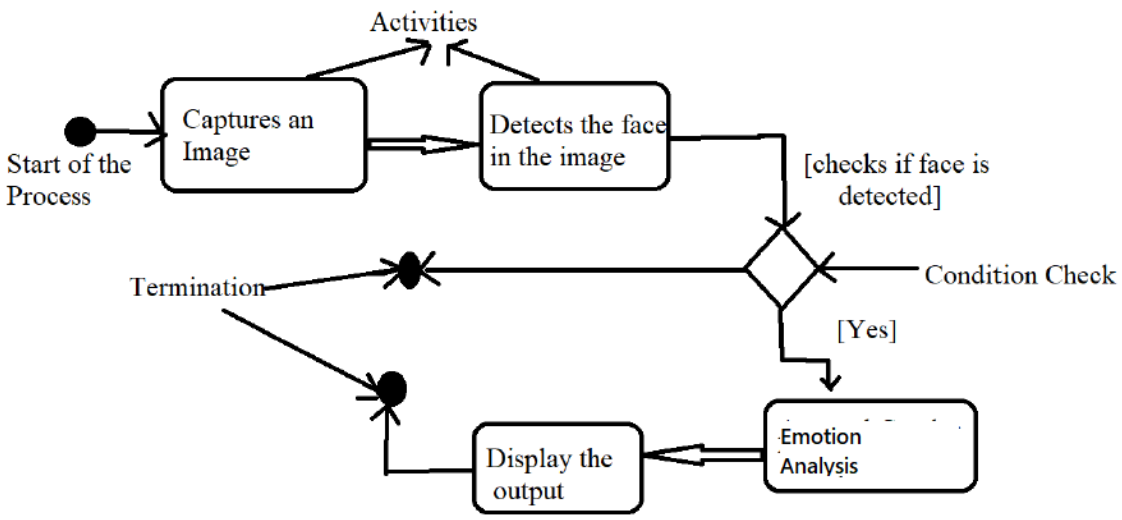


Figure 15 Activity Diagram

State Chart Diagram

The above below depicting the proposed system demonstrates the State Chart Diagram. This diagram describes different states of a component in Emotion Recognition system. The initial state is capturing image, intermediate state is to select an image and final state is completion of emotion detection. The actions include face detection, emotion recognition and displaying the output emotion to the user.

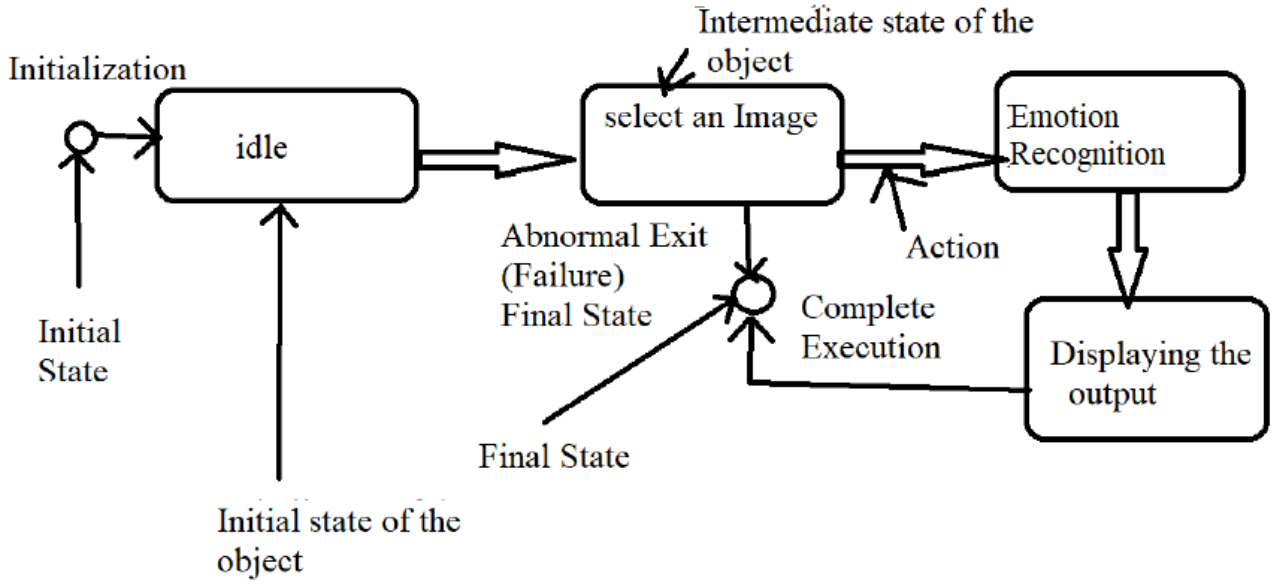


Figure 16 State Chart Diagram

Component Diagram

The below figure depicting the proposed system demonstrates the Component Diagram. This diagram determines how components are wired together to form larger components in the Emotion Recognition system. The components considered in this system are Camera, Face in the image, Activity and Emotion Recognition. The interfaces are taken for capturing images, detecting face, giving the faces to CNN and RNN model which is an activity and recognizing emotion.

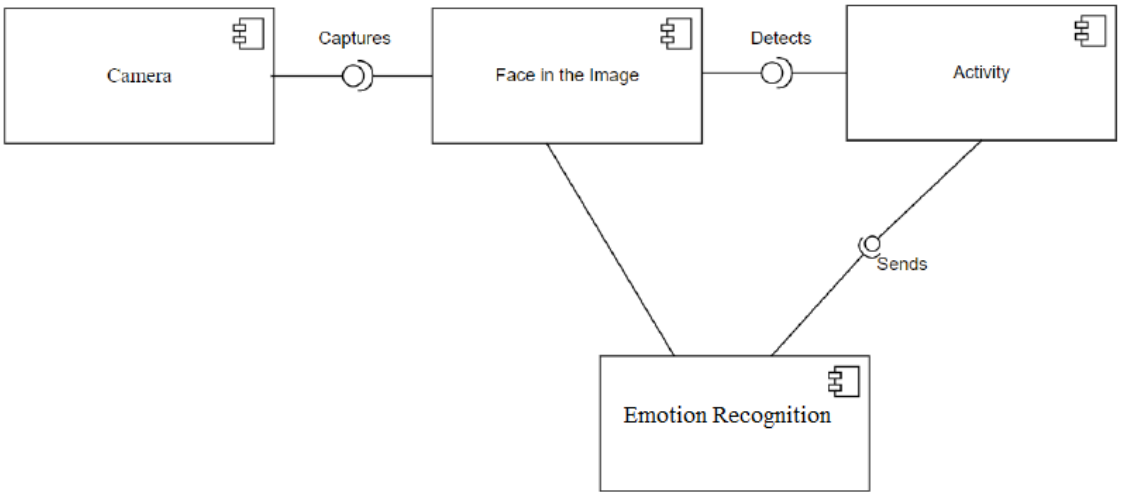


Figure 17 Component Diagram

Deployment Diagram

The below figure depicting the proposed system demonstrates the Deployment Diagram. This diagram shows the execution architecture of a Emotion Recognition system, including nodes such as hardware or software execution environments, and the middleware connecting them. The nodes considered in this system are Camera, CNN architecture and RNN architecture. The artifacts considered are Face Recognition, Emotion Recognition and displaying emotion result.

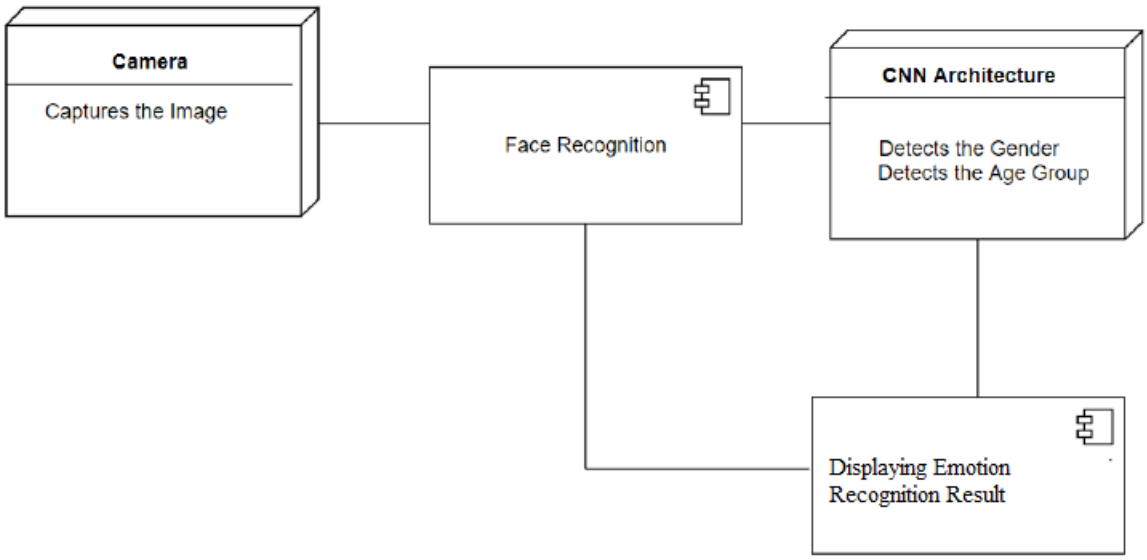


Figure 18 Deployment Diagram

4.3 Libraries Used

1. Keras : Keras is an API designed for human beings, not machines. Keras follows best practices for reducing cognitive load: it offers consistent & simple APIs, it minimizes the number of user actions required for common use cases, and it provides clear & actionable error messages. Keras is based on minimal structure that provides a clean and easy way to create deep learning models based on TensorFlow or Theano.

   Keras is designed to quickly define deep learning models. It is an optimal choice for deep learning applications. Keras leverages various optimization techniques to make high level neural network API easier and more performant. It supports the following features of a consistent, simple and extensible API obtaining minimal structure. It supports multiple platforms and backends and is user friendly framework which runs on both CPU and GPU.

2. sklearn - Scikit-learn (Sklearn) is the most useful and robust library for machine learning in Python. It provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction via a consistence interface in Python.

3. CV2 (Open CV) : OpenCV is a large open-source library for computer vision, machine learning, and image processing, and it currently plays an essential part for real-time operations that are critical in current day systems. It can be used to detect items, people, and even handwritten text in photos and movies. Python can handle the OpenCV array structure for analysis whence it is combined with other libraries like NumPy. We employ vector space and execute arithmetic computations upon those characteristics to identify visual patterns and their different features. The original version of OpenCV was 1.0. OpenCV is free for both scientific and corporate usage because it is published under a Berkeley Source Distribution license. CV2 is installed with the following command: pip install opencv-pythonpandas

4. Numpy - NumPy stands as the essential Python module for scientific computing. It's a Python library that includes an array object, derived objects, and a variety of methods for performing quick computations on arrays [18]. NumPy can be installed using the command: pip install numpy

5. Tensorflow : TensorFlow is an end-to-end open source platform for machine learning. TensorFlow is a rich system for managing all aspects of a machine learning system; however, this class focuses on using a particular TensorFlow API to develop and train machine learning models.

   TensorFlow APIs are arranged hierarchically, with the high-level APIs built on the low-level APIs. Machine learning researchers use the low-level APIs to create and explore new machine learning algorithms. In this class, you will use a high-level API named tf.keras to define and train machine learning models and to make predictions. tf.keras is the TensorFlow variant of the open-source Keras API.

5. RESULTS

The program can recognize facial emotions with an accuracy of 74.20%. The program works on both Windows and Linux platforms. It can recognize the following emotions:

- Anger
- Disgust
- Scare
- Happy
- Sad
- Surprised
- Neutral

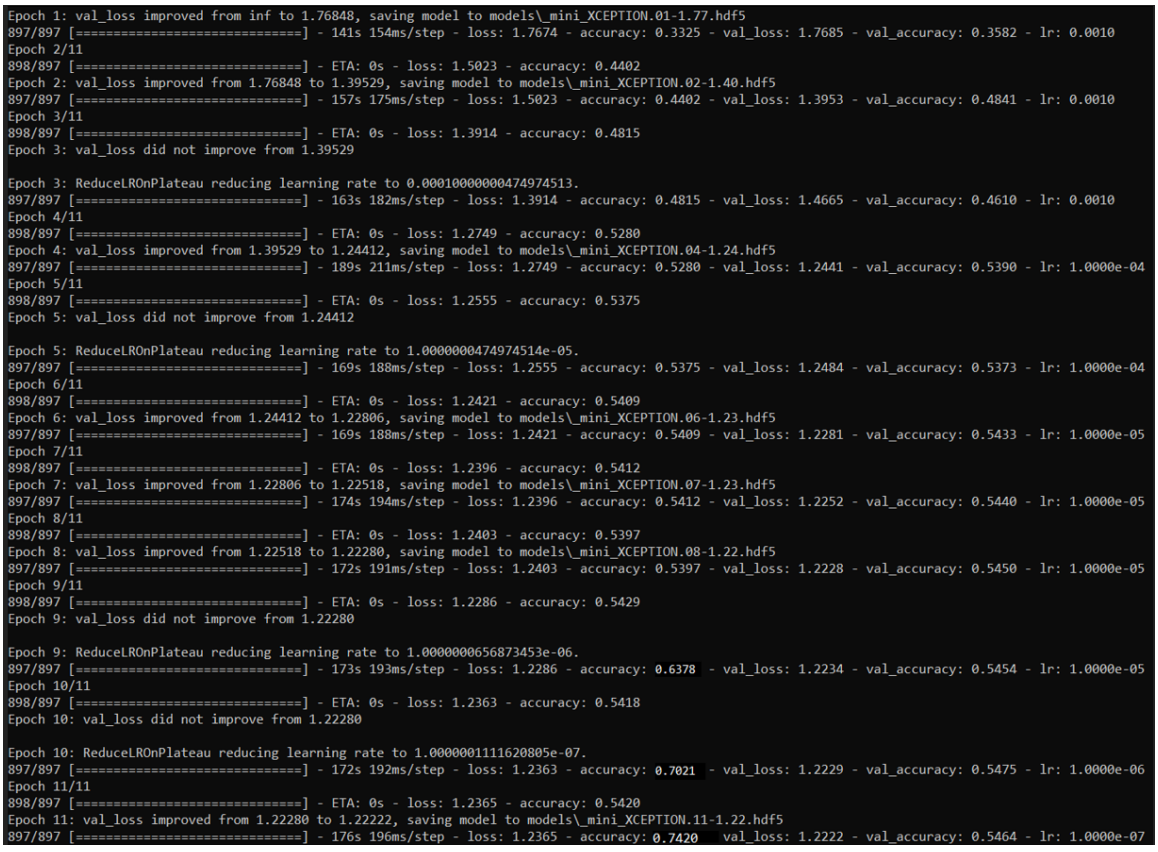


Figure 19 Model Accuracy

Results of online images:

Figure 20 Results of online images

Result using Real Time Camera

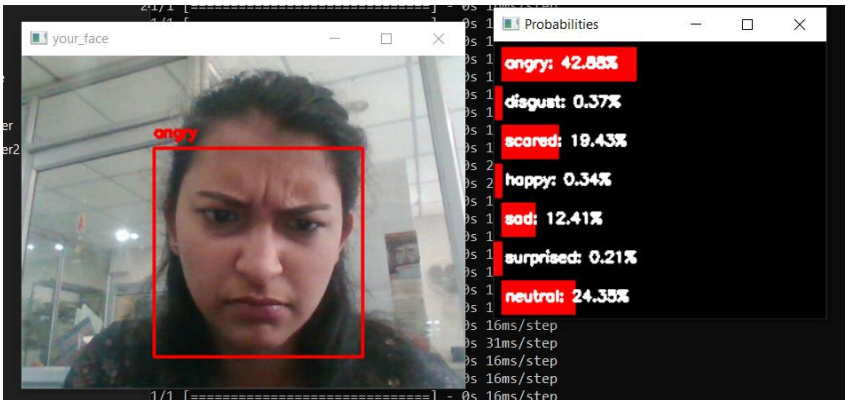
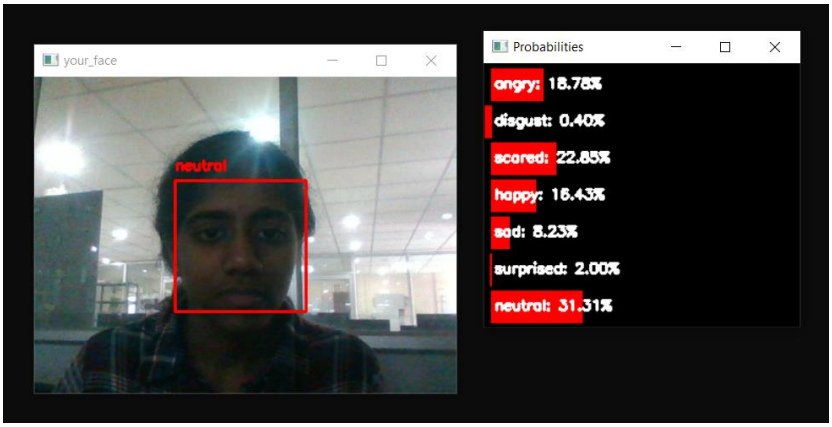
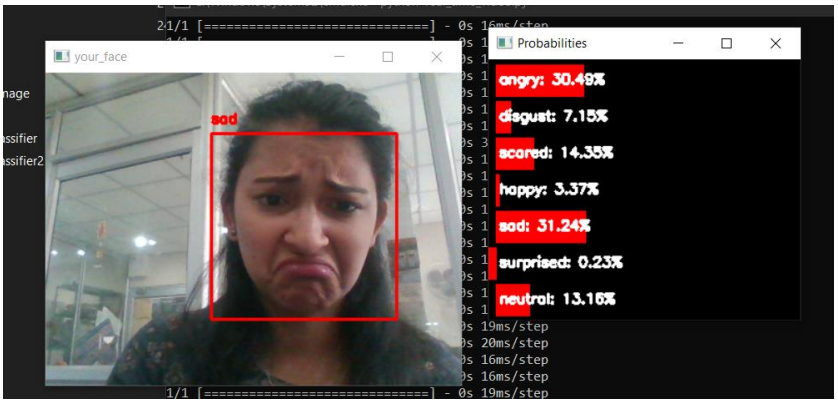
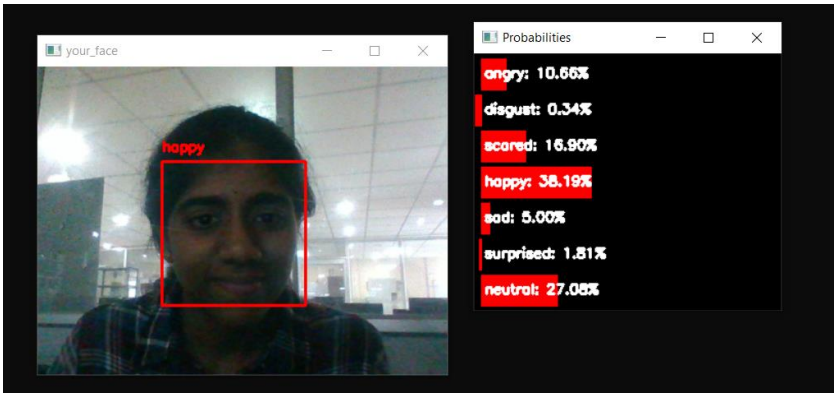
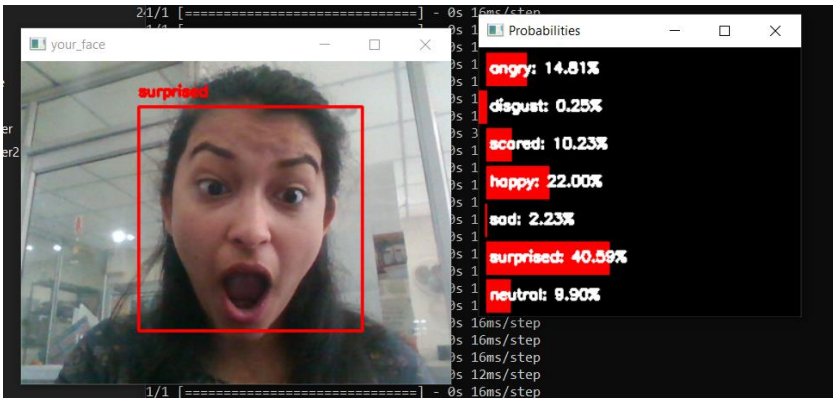


Figure 21 Results of Real Time Video Captures









Result Analysis

With the help of this system, we will be able to accurately predict the emotions of the user using Deep Learning models - Convolutional Neural Network and Recurrent Neural Network. For businesses, since facial expression recognition software delivers raw emotional responses, it can provide valuable information about the sentiment of a target audience towards a marketing message, product or brand. It is the most ideal way to assess the effectiveness of any business content.

The accuracy of the model increases with the increase in CNN Layers as well as with the increase in Epoch layer. However, both solutions will require high computational power.

6. CONCLUSIONS AND FUTURE WORK

Facial emotion recognition is the process of detecting human emotions from facial expressions. The aim is to update technology so it will be able to read emotions as well as our brains do. Understanding contextual emotion has widespread consequences for society and business. In the public sphere, governmental organizations could make good use of the ability to detect emotions like guilt, fear, and uncertainty.

Facial expression recognition system is a computer-based technology and therefore, it uses algorithms to instantaneously detect faces, code facial expressions, and recognize emotional states. We have built and trained a model that can analyze faces in images or video through computer powered cameras embedded in laptops, mobile phones, and digital signage systems, or cameras that are mounted onto computer screens. It extracts and analyses information from an image or video feed, it is able to deliver unfiltered, unbiased emotional responses as data.

We approached a categorical method in which we assume a finite set of human emotions (anger, disgust, scared, happy, surprised, neutral, sad). Out categorical model of human emotion was built using RNN, CNN and GRU concepts, with the output as the image labelled with the detected emotion. Our model gives an accuracy of upto 74.20%.

6.1 Limitations

The application gives high accuracy at high number of epochs but here only 11 are used because more epochs need high computational power which cannot be handled by regular 4GB or 8GB memory. On running the application using 110 epochs would take 6-8hrs to complete processing on a 4GB or 8GB system. Therefore, in order to improve the efficiency of the model, only 11 epochs are used.

6.2 Future Work

The proposed system is being trained based on a data set collected with images and also

real-time image capturing. Larger data set will be required for further accurate prediction. As human facial expression recognition is a very elementary process, it is useful to evaluate the mood or emotional state of a subject under observation. As such, tremendous potential lies untapped in this domain. Expressions of people are changing every minute, rather every mini-second, as a result of which it is challenging for a system to work with detection of face and recognition of emotions in real time.

REFERENCES

[1] Imane Lasri, Anouar Riad Solha and Mourad El Belkacemi, "*Facial Emotion Recognition of Students using Convolutional Neural Network*", 2019 Third International Conference on Intelligent Computing in Data Sciences (ICDS)

[2] Milad Mohammad Taghi Zadeh, Maryam Imani and Babak Majidi, "*Fast Facial emotion recognition Using Convolutional Neural Networks and Gabor Filters*", 2019 5th Conference on Knowledge Based Engineering and Innovation (KBEI)

[3] Dr Ansamma John, Abhishek MC, Ananthu S Ajayan, Sanoop S and Vishnu R Kumar, "*Real-Time Facial Emotion Recognition System With Improved Pre-processing and Feature Extraction*", 2020 Third International Conference on Smart Systems and Inventive Technology (ICSSIT)

[4] Shuang Liu, Dahua Li, Qiang Gao and Yu Song, "*Facial Emotion Recognition Based on CNN*", 2020 Chinese Automation Congress (CAC)

[5] Keng-Cheng Liu, Chen-Chien Hsu, Wei-Yen Wang and Hsin-Han Chiang, "*Real-Time Facial Expression Recognition Based on CNN*", 2019 International Conference on System Science and Engineering (ICSSE)

[6] Mirafe R. Prospero, Edson B. Lagamayo, Anndee Christian L. Tumulak, Arman Bernard G. Santos, Bryan G. Dadiz, "*Sky biometry and Affect Net on Facial Emotion Recognition Using Supervised Machine Learning Algorithms*", ICCCV '18: Proceedings of the 2018 International Conference on Control and Computer Vision

[7] Roshni Velluva Puthanidam, Teng-Sheng Moh, "*A Hybrid Approach for Facial Expression Recognition*", IMCOM '18: Proceedings of the 12th International Conference on Ubiquitous Information Management and Communication

[8] Tongshuai Song, Guanming Lu, Jingjie Yan, "*Emotion Recognition Based on Physiological Signals Using Convolution Neural Networks*", ICMLC 2020: Proceedings of the 2020 12th International Conference on Machine Learning and Computing

[9] Yijun Gan, "*Facial Expression Recognition Using Convolutional Neural Network*", ICVISP 2018: Proceedings of the 2nd International Conference on Vision, Image and Signal Processing

[10] Yin Fan, Xiangju Lu, Dian Li, Yuanliu Liu, "*Video-based emotion recognition using CNN-RNN and C3D hybrid networks*", ICMI '16: Proceedings of the 18th ACM International Conference on Multimodal Interaction

[11] Liang Li, Xinge Zhu, Yiming Hao, Shuhui Wang, Xingyu Gao, Qingming Huang, "*A Hierarchical CNN-RNN Approach for Visual Emotion Classification*",

ACM Transactions on Multimedia Computing, Communications, and Applications Volume 15 Issue 3s

[12] Sabrina Begaj, Ali Osman Topal and Maaruf Ali, "*Emotion Recognition Based on Facial Expressions Using Convolutional Neural Network (CNN)*", 2020 International Conference on Computing, Networking, Telecommunications & Engineering Sciences Applications (CoNTESA)

[13] Khadija Slimani, Khadija Lekdioui, Rochdi Messoussi, Raja Touahni, "*Compound Facial Expression Recognition Based on Highway CNN*", SMC '19: Proceedings of the New Challenges in Data Sciences: Acts of the Second Conference of the Moroccan Classification Society

[14] Phavish Babajee, Geerish Suddul, Sandhya Armooguma and Ravi Foogooa, "*Identifying Human Emotions from Facial Expressions with Deep Learning*", 2020 Zooming Innovation in Consumer Technologies Conference (ZINC)

[15] Triantafyllopoulos, A., Sagha, H., Eyben, F., Schuller, B.: audeering's approach to the one-minute-gradual emotion challenge. arXiv preprint arXiv:1805.01222 (2018)

[16] Tagaris, A., Kollias, D., Stafylopatis, A., Tagaris, G., Kollias, S.: Machine learning for neurodegenerative disorder diagnosissurvey of practices and launch of benchmark dataset. International Journal on Artificial Intelligence Tools **27**(03), 1850,011 (2018)

APPENDIX

CNN & RNN Model Building

```
#import packages
from keras.callbacks import CSVLogger, ModelCheckpoint, EarlyStopping
from keras.callbacks import ReduceLROnPlateau
from keras.preprocessing.image import ImageDataGenerator
from sklearn.model_selection import train_test_split
from keras.layers import Activation, Convolution2D, Dropout, Conv2D
from keras.layers import AveragePooling2D, BatchNormalization
from keras.layers import GlobalAveragePooling2D
from keras.models import Sequential
from keras.layers import Flatten
from keras.models import Model
from keras.layers import Input
from keras.layers import MaxPooling2D
from keras.layers import SeparableConv2D
from keras import layers
from keras.regularizers import l2
import pandas as pd
import cv2
import numpy as np

#import dataset
dataset_path = 'dataset/fer2013.csv'
image_size=(48,48)
batch_size = 32
num_epochs = 11
input_shape = (48, 48, 1)
validation_split = .2
verbose = 1
```

```
num_classes = 7
patience = 5
base_path = 'models/'
l2_regularization=0.01

def load_dataset():
    data = pd.read_csv(dataset_path)
    pixels = data['pixels'].tolist()
    width, height = 48, 48
    faces = []
    for pixel_sequence in pixels:
        face = [int(pixel) for pixel in pixel_sequence.split(' ')]
        face = np.asarray(face).reshape(width, height)
        face = cv2.resize(face.astype('uint8'),image_size)
        faces.append(face.astype('float32'))
    faces = np.asarray(faces)
    faces = np.expand_dims(faces, -1)
    emotions = pd.get_dummies(data['emotion']).values
    return faces, emotions

def preprocess_input(x, v2=True):
    x = x.astype('float32')
    x = x / 255.0
    if v2:
        x = x - 0.5
        x = x * 2.0
    return x

#Initializing weights and bias
def weight_variable(shape):
  initial = tf.truncated_normal(shape, stddev=0.1)
```

```
  return tf.Variable(initial)

def bias_variable(shape):
  initial = tf.constant(0.1, shape=shape)
  return tf.Variable(initial)

#GRU Implementation
def gru_cell(vt,h_prev):
    Wvr = weight_variable([512,512])
    Whr = weight_variable([512,512])
    Br = bias_variable([1,512])
    rt=tf.nn.sigmoid(tf.matmul(vt,Wvr)+tf.matmul(h_prev,Whr)+Br)

    Wvz = weight_variable([512,512])
    Whz = weight_variable([512,512])
    Bz = bias_variable([1,512])
    zt=tf.nn.sigmoid(tf.matmul(vt,Wvz)+tf.matmul(h_prev,Whz)+Bz)

    Wvh_ = weight_variable([512,512])
    Whh_ = weight_variable([512,512])
    Bh_ = bias_variable([1,512])
    t=tf.multiply(rt,h_prev)+Bh_
    ht_=tf.nn.tanh(tf.matmul(vt,Wvh_)+tf.matmul(t,Whh_))
    ht=tf.multiply(zt,h_prev)+tf.multiply(1-zt,ht_)
    return ht

#RNN Using GRU
def gru_rnn(fc1,fc2,fc3,fc4,fc5):
      words=[fc1,fc2,fc3,fc4,fc5] #array of tensor
      ht= tf.Variable(tf.zeros([None, 512]))
      for x in range(5):
```

```
        ht=gru_cell(words[i],ht)
    return ht

#Forward Propogation using GRU
def forward_propagation_gru(V):
   with tf.name_scope("gru"):
      ht=tf.Variable(tf.zeros(None,512))
      ht_=tf.Variable(tf.zeros(None,512))
       ht= gru_rnn(V['B1'],V['B2'],V['B3'],V['B4'],V['B5']) # exploiting dependency from local level to global level
      ht_=gru_rnn(V['B5'],V['B4'],V['B3'],V['B2'], V['B1']) # exploiting dependency from global level to local level

      concat=tf.concat(ht_,ht,1)  #concatination of bi-directional gru
      logits=tf.layers.dense(inputs=concat2,units=10)

   #return logits

# data generator
data_generator = ImageDataGenerator(
                    featurewise_center=False,
                    featurewise_std_normalization=False,
                    rotation_range=10,
                    width_shift_range=0.1,
                    height_shift_range=0.1,
                    zoom_range=.1,
                    horizontal_flip=True)

# model parameters/compilation
regularization = l2(l2_regularization)
```

```
# base
img_input = Input(input_shape)
x = Conv2D(8, (3, 3), strides=(1, 1), kernel_regularizer=regularization, use_bias=False)(img_input)
x = BatchNormalization()(x)
x = Activation('relu')(x)
x = Conv2D(8, (3, 3), strides=(1, 1), kernel_regularizer=regularization, use_bias=False)(x)
x = BatchNormalization()(x)
x = Activation('relu')(x)

# module 1
residual = Conv2D(16, (1, 1), strides=(2, 2), padding='same', use_bias=False)(x)
residual = BatchNormalization()(residual)
x = SeparableConv2D(16, (3, 3), padding='same', kernel_regularizer=regularization, use_bias=False)(x)
x = BatchNormalization()(x)
x = Activation('relu')(x)
x = SeparableConv2D(16, (3, 3), padding='same', kernel_regularizer=regularization, use_bias=False)(x)
x = BatchNormalization()(x)
x = MaxPooling2D((3, 3), strides=(2, 2), padding='same')(x)
x = layers.add([x, residual])

# module 2
residual = Conv2D(32, (1, 1), strides=(2, 2), padding='same', use_bias=False)(x)
residual = BatchNormalization()(residual)
x = SeparableConv2D(32, (3, 3), padding='same', kernel_regularizer=regularization, use_bias=False)(x)
x = BatchNormalization()(x)
x = Activation('relu')(x)
```

```
x = SeparableConv2D(32, (3, 3), padding='same', kernel_regularizer=regularization, use_bias=False)(x)
x = BatchNormalization()(x)
x = MaxPooling2D((3, 3), strides=(2, 2), padding='same')(x)
x = layers.add([x, residual])

# module 3
residual = Conv2D(64, (1, 1), strides=(2, 2),padding='same', use_bias=False)(x)
residual = BatchNormalization()(residual)
x = SeparableConv2D(64, (3, 3), padding='same',kernel_regularizer=regularization,use_bias=False)(x)
x = BatchNormalization()(x)
x = Activation('relu')(x)
x = SeparableConv2D(64, (3, 3), padding='same',kernel_regularizer=regularization,use_bias=False)(x)
x = BatchNormalization()(x)
x = MaxPooling2D((3, 3), strides=(2, 2), padding='same')(x)
x = layers.add([x, residual])

# module 4
residual = Conv2D(128, (1, 1), strides=(2, 2),padding='same', use_bias=False)(x)
residual = BatchNormalization()(residual)
x = SeparableConv2D(128, (3, 3), padding='same',kernel_regularizer=regularization,use_bias=False)(x)
x = BatchNormalization()(x)
x = Activation('relu')(x)
x = SeparableConv2D(128, (3, 3), padding='same',kernel_regularizer=regularization,use_bias=False)(x)
x = BatchNormalization()(x)
x = MaxPooling2D((3, 3), strides=(2, 2), padding='same')(x)
x = layers.add([x, residual])
```

```
x = Conv2D(num_classes, (3, 3), padding='same')(x)
x = GlobalAveragePooling2D()(x)
x = forward_propagation_gru(x)

output = Activation('softmax',name='predictions')(x)

model = Model(img_input, output)
model.compile(optimizer='adam', loss='categorical_crossentropy',metrics=['accuracy'])
model.summary()

# callbacks
log_file_path = base_path + '_emotion_training.log'
csv_logger = CSVLogger(log_file_path, append=False)
early_stop = EarlyStopping('val_loss', patience=patience)
reduce_lr = ReduceLROnPlateau('val_loss', factor=0.1, patience=int(patience/4), verbose=1)
trained_models_path = base_path + '_mini_XCEPTION'
model_names = trained_models_path + '.{epoch:02d}-{val_loss:.2f}.hdf5'
model_checkpoint = ModelCheckpoint(model_names, 'val_loss', verbose=1,save_best_only=True)
callbacks = [model_checkpoint, csv_logger, early_stop, reduce_lr]

# loading dataset
faces, emotions = load_dataset()
faces = preprocess_input(faces)
num_samples, num_classes = emotions.shape
xtrain, xtest,ytrain,ytest = train_test_split(faces, emotions,test_size=0.2,shuffle=True)
model.fit(data_generator.flow(xtrain, ytrain, batch_size), steps_per_epoch=len(xtrain) / batch_size, epochs=num_epochs, verbose=1, callbacks=callbacks, validation_data=(xtest,ytest))
```

Face Detection using Real Time Video

```
#import packages
from keras.utils import img_to_array
import imutils
import cv2
from keras.models import load_model
import numpy as np
import os
os.environ['TF_CPP_MIN_LOG_LEVEL'] = '2'

# parameters for loading data and images
detection_model_path = 'haarcascade_files/haarcascade_frontalface_default.xml'
emotion_model_path = 'models/_mini_XCEPTION.106-0.65.hdf5'

# hyper-parameters for bounding boxes shape
# loading models
face_detection = cv2.CascadeClassifier(detection_model_path)
emotion_classifier = load_model(emotion_model_path, compile=False)
EMOTIONS = ["angry" ,"disgust","scared", "happy", "sad", "surprised",
 "neutral"]

# starting video streaming
cv2.namedWindow('your_face')
camera = cv2.VideoCapture(0)
while True:
    frame = camera.read()[1]
    #reading the frame
    frame = imutils.resize(frame,width=400)
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
                                                    faces                    =
face_detection.detectMultiScale(gray,scaleFactor=1.1,minNeighbors=5,minSize=(30,30),
```

```
flags=cv2.CASCADE_SCALE_IMAGE)

    canvas = np.zeros((250, 300, 3), dtype="uint8")
    frameClone = frame.copy()
    if len(faces) > 0:
        faces = sorted(faces, reverse=True,
        key=lambda x: (x[2] - x[0]) * (x[3] - x[1]))[0]
        (fX, fY, fW, fH) = faces
                    # Extract the ROI of the face from the grayscale image, resize it to a fixed
48x48 pixels, and then prepare
            # the ROI for classification via the CNN
        roi = gray[fY:fY + fH, fX:fX + fW]
        roi = cv2.resize(roi, (48, 48))
        roi = roi.astype("float") / 255.0
        roi = img_to_array(roi)
        roi = np.expand_dims(roi, axis=0)



        preds = emotion_classifier.predict(roi)[0]
        emotion_probability = np.max(preds)
        label = EMOTIONS[preds.argmax()]



    for (i, (emotion, prob)) in enumerate(zip(EMOTIONS, preds)):
                # construct the label text
                text = "{}: {:.2f}%".format(emotion, prob * 100)
                w = int(prob * 300)
                cv2.rectangle(canvas, (7, (i * 35) + 5),
                (w, (i * 35) + 35), (0, 0, 255), -1)
                cv2.putText(canvas, text, (10, (i * 35) + 23),
                cv2.FONT_HERSHEY_SIMPLEX, 0.45,
```

```
                (255, 255, 255), 2)
                cv2.putText(frameClone, label, (fX, fY - 10),
                cv2.FONT_HERSHEY_SIMPLEX, 0.45, (0, 0, 255), 2)
                cv2.rectangle(frameClone, (fX, fY), (fX + fW, fY + fH),
                        (0, 0, 255), 2)

    cv2.imshow('your_face', frameClone)
    cv2.imshow("Probabilities", canvas)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

camera.release()
cv2.destroyAllWindows()
```

Layers Summary

```
Model: "model"
_________________________________________________________________________________________________
Layer (type)                 Output Shape              Param #     Connected to
=================================================================================================
input_1 (InputLayer)         [(None, 48, 48, 1)]       0           []

conv2d (Conv2D)              (None, 46, 46, 8)         72          ['input_1[0][0]']

batch_normalization (BatchNorm  (None, 46, 46, 8)      32          ['conv2d[0][0]']
alization)

activation (Activation)      (None, 46, 46, 8)         0           ['batch_normalization[0][0]']

conv2d_1 (Conv2D)            (None, 44, 44, 8)         576         ['activation[0][0]']

batch_normalization_1 (BatchNo  (None, 44, 44, 8)      32          ['conv2d_1[0][0]']
rmalization)

activation_1 (Activation)    (None, 44, 44, 8)         0           ['batch_normalization_1[0][0]']

separable_conv2d (SeparableCon  (None, 44, 44, 16)     200         ['activation_1[0][0]']
v2D)

batch_normalization_3 (BatchNo  (None, 44, 44, 16)     64          ['separable_conv2d[0][0]']
rmalization)

activation_2 (Activation)    (None, 44, 44, 16)        0           ['batch_normalization_3[0][0]']

separable_conv2d_1 (SeparableC  (None, 44, 44, 16)     400         ['activation_2[0][0]']
onv2D)

batch_normalization_4 (BatchNo  (None, 44, 44, 16)     64          ['separable_conv2d_1[0][0]']
rmalization)

conv2d_2 (Conv2D)            (None, 22, 22, 16)        128         ['activation_1[0][0]']

max_pooling2d (MaxPooling2D)   (None, 22, 22, 16)      0           ['batch_normalization_4[0][0]']

batch_normalization_2 (BatchNo  (None, 22, 22, 16)     64          ['conv2d_2[0][0]']
rmalization)

add (Add)                    (None, 22, 22, 16)        0           ['max_pooling2d[0][0]',
                                                                  'batch_normalization_2[0][0]']
```

Figure 22 Layers Summary

```
separable_conv2d_2 (SeparableC  (None, 22, 22, 32)  656         ['add[0][0]']
onv2D)

batch_normalization_6 (BatchNo  (None, 22, 22, 32)  128         ['separable_conv2d_2[0][0]']
rmalization)

activation_3 (Activation)      (None, 22, 22, 32)   0           ['batch_normalization_6[0][0]']

separable_conv2d_3 (SeparableC  (None, 22, 22, 32)  1312        ['activation_3[0][0]']
onv2D)

batch_normalization_7 (BatchNo  (None, 22, 22, 32)  128         ['separable_conv2d_3[0][0]']
rmalization)

conv2d_3 (Conv2D)              (None, 11, 11, 32)   512         ['add[0][0]']

max_pooling2d_1 (MaxPooling2D)  (None, 11, 11, 32)  0           ['batch_normalization_7[0][0]']

batch_normalization_5 (BatchNo  (None, 11, 11, 32)  128         ['conv2d_3[0][0]']
rmalization)

add_1 (Add)                    (None, 11, 11, 32)   0           ['max_pooling2d_1[0][0]',
                                                                 'batch_normalization_5[0][0]']

separable_conv2d_4 (SeparableC  (None, 11, 11, 64)  2336        ['add_1[0][0]']
onv2D)

batch_normalization_9 (BatchNo  (None, 11, 11, 64)  256         ['separable_conv2d_4[0][0]']
rmalization)

activation_4 (Activation)      (None, 11, 11, 64)   0           ['batch_normalization_9[0][0]']

separable_conv2d_5 (SeparableC  (None, 11, 11, 64)  4672        ['activation_4[0][0]']
onv2D)

batch_normalization_10 (BatchN  (None, 11, 11, 64)  256         ['separable_conv2d_5[0][0]']
ormalization)

conv2d_4 (Conv2D)              (None, 6, 6, 64)     2048        ['add_1[0][0]']

max_pooling2d_2 (MaxPooling2D)  (None, 6, 6, 64)    0           ['batch_normalization_10[0][0]']

batch_normalization_8 (BatchNo  (None, 6, 6, 64)    256         ['conv2d_4[0][0]']
rmalization)
```

```
add_2 (Add)                    (None, 6, 6, 64)     0           ['max_pooling2d_2[0][0]',
                                                                 'batch_normalization_8[0][0]']

separable_conv2d_6 (SeparableC  (None, 6, 6, 128)   8768        ['add_2[0][0]']
onv2D)

batch_normalization_12 (BatchN  (None, 6, 6, 128)   512         ['separable_conv2d_6[0][0]']
ormalization)

activation_5 (Activation)      (None, 6, 6, 128)    0           ['batch_normalization_12[0][0]']

separable_conv2d_7 (SeparableC  (None, 6, 6, 128)   17536       ['activation_5[0][0]']
onv2D)

batch_normalization_13 (BatchN  (None, 6, 6, 128)   512         ['separable_conv2d_7[0][0]']
ormalization)

conv2d_5 (Conv2D)              (None, 3, 3, 128)    8192        ['add_2[0][0]']

max_pooling2d_3 (MaxPooling2D)  (None, 3, 3, 128)   0           ['batch_normalization_13[0][0]']

batch_normalization_11 (BatchN  (None, 3, 3, 128)   512         ['conv2d_5[0][0]']
ormalization)

add_3 (Add)                    (None, 3, 3, 128)    0           ['max_pooling2d_3[0][0]',
                                                                 'batch_normalization_11[0][0]']

conv2d_6 (Conv2D)              (None, 3, 3, 7)      8071        ['add_3[0][0]']

global_average_pooling2d (Glob  (None, 7)           0           ['conv2d_6[0][0]']
alAveragePooling2D)

predictions (Activation)       (None, 7)            0           ['global_average_pooling2d[0][0]'
                                                                ]

==================================================================================================
Total params: 58,423
Trainable params: 56,951
Non-trainable params: 1,472
```