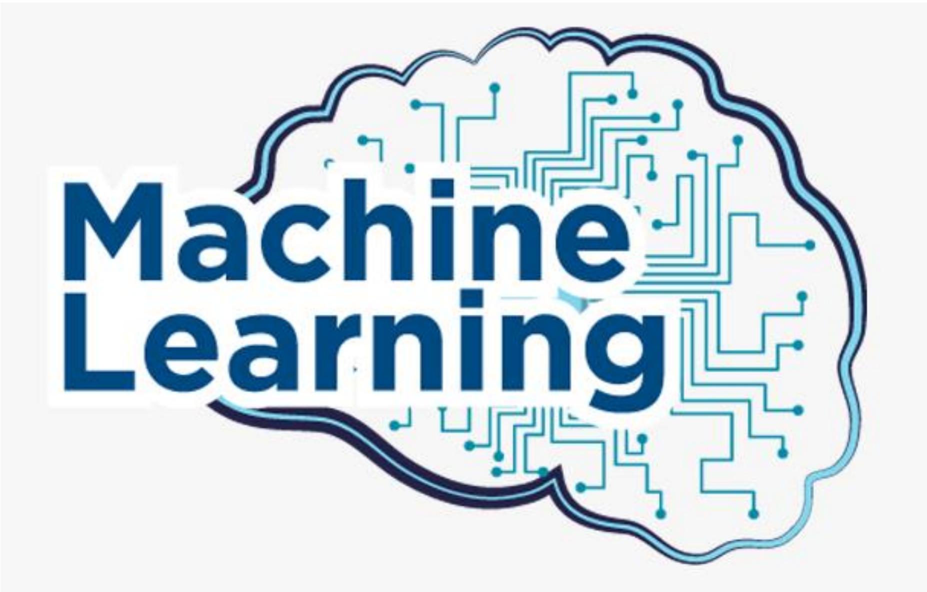
# EMOJIFY

## (PROJECT FILE)




Course Instructor – Pratima Singh

(ECAM – 2)



| NAME OF THE STUDENT | UNIVERSITY ROLL NO. |
|---|---|
| ANSH DHALLA | 2020UEA6581 |
| SHIVAM THAKUR | 2020UEA6589 |
| CHIRAG KAPOOR | 2020UEA6579 |

# ***TABLE OF CONTENTS***

| **TOPIC** | **PAGE NO.** |
|---|---|

# ABSTRACT

Emojis are small images that are commonly included in social media text messages. The combination of visual and textual content in the same message builds up a modern way of communication. Emojis or avatars are ways to indicate nonverbal cues. These cues have become an essential part of online chatting, product review, brand emotion, and many more. It also led to increasing data science research dedicated to emoji-driven storytelling. With advancements in computer vision and deep learning, it is now possible to detect human emotions from images. In this deep learning project, we will classify human facial expressions to filter and map corresponding emojis or avatars. This project is not intended to solve a real-world problem, instead it allows us to see things more colorful in the chatting world. Emojify is a software which deals with the creation of Emoji's or Avatars.

**Key – words:** Convolutional Neural Network (CNN), Deep Learning, Fer2013 Dataset.

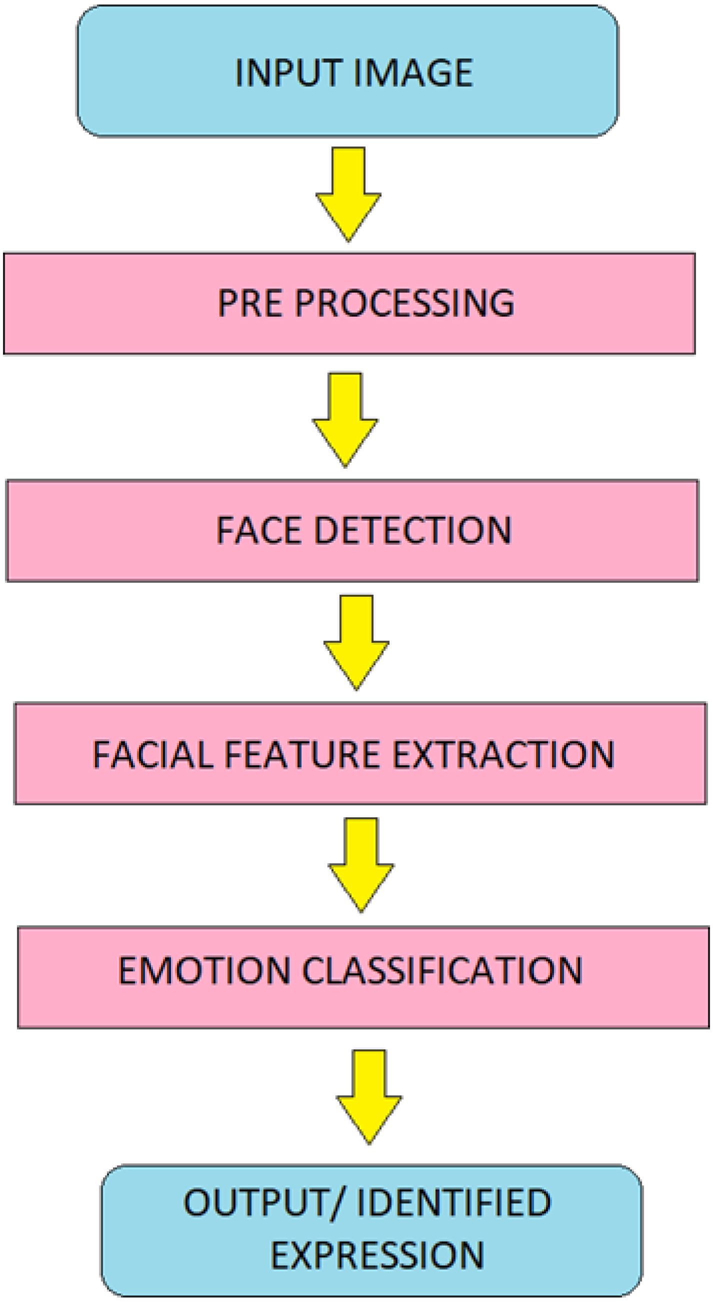
# INTRODUCTION

In this project, we will build and train a convolutional neural network (CNN) in Keras from scratch to recognize facial expressions. The data consists of 48x48 pixel grayscale images of faces. The objective is to classify each face based on the emotion shown in the facial expression into one of seven categories (0 = Angry, 1 = Happy, 2 = Neutral, 3 = Sad, 4 = Surprise). We will use OpenCV to automatically detect faces in images and draw bounding boxes around them. Once we have trained, saved, and exported the CNN, we will directly serve the trained model to a web interface and perform real-time facial expression recognition on video and image data.
The project can be broadly divided into two parts -

1. Build and train a model in Keras for Facial Expression Recognition.

2. Test the model using real time images

# METHODOLOGY




**THE WORK FLOW OF OUR PROJECT**

# DATASET & ITS FEATURES

In this project, the dataset used to train the models is FER-2013. The FER-2013 dataset consists of 35887 images, of which 28709 labelled images belong to the training set and the remaining 7178 images belong to the test set. The images in FER-2013 dataset is labeled as one of the seven universal emotions: Happy, Sad, Angry, Surprise, and Neutral. Among these emotion classifications, the most images belong to 'happy' emotions account to 7215 images of the 28709 training images. The number of images that belong to each emotion is given by returning the length of each directory using the OS module in Python. The images in FER-2013 dataset are in grayscale and is of dimensions 48x48 pixels. This dataset was created by gathering results from Google Image search of each emotions. The number of images of each emotion is given in table 1. The number of images of each emotion type is returned by the functions of 'OS' module in python. To explore the dataset further, and to understand what kind of images lie in the dataset, we plot few example images from the dataset using the 'utils' module in python. The resultant plot of example images obtained is given in figure 1. From various research papers, we have studied that the average attainable accuracy of a training model developed using FER-2013 dataset is 66.7% and our aim is also to design a CNN model with similar or a better accuracy.

Table 1: Number of images of each emotion type

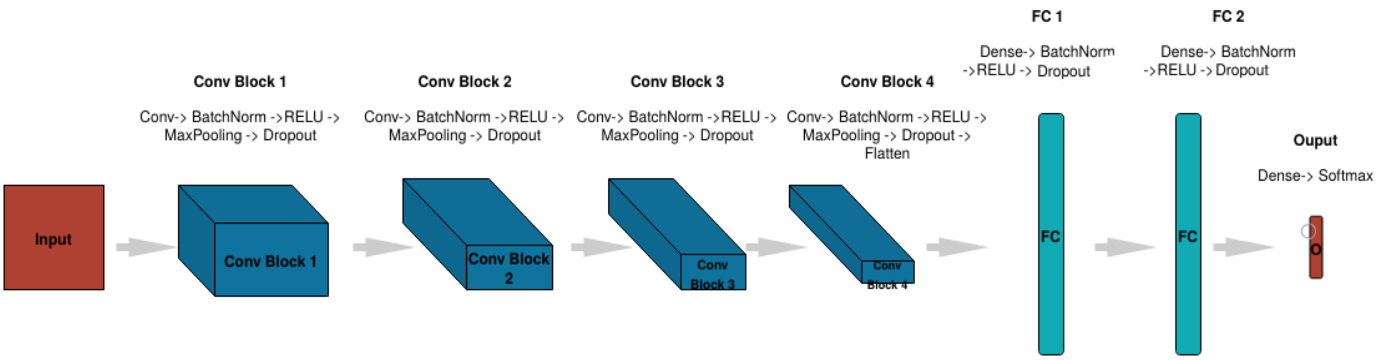
| S.No. | Type of Emotion | No. of images in thedataset |
|---|---|---|
| 1. | Angry | 3995 |
| 2. | Happy | 7215 |
| 3. | Neutral | 4965 |
| 4. | Sad | 4830 |
| 5. | Surprise | 3171 |

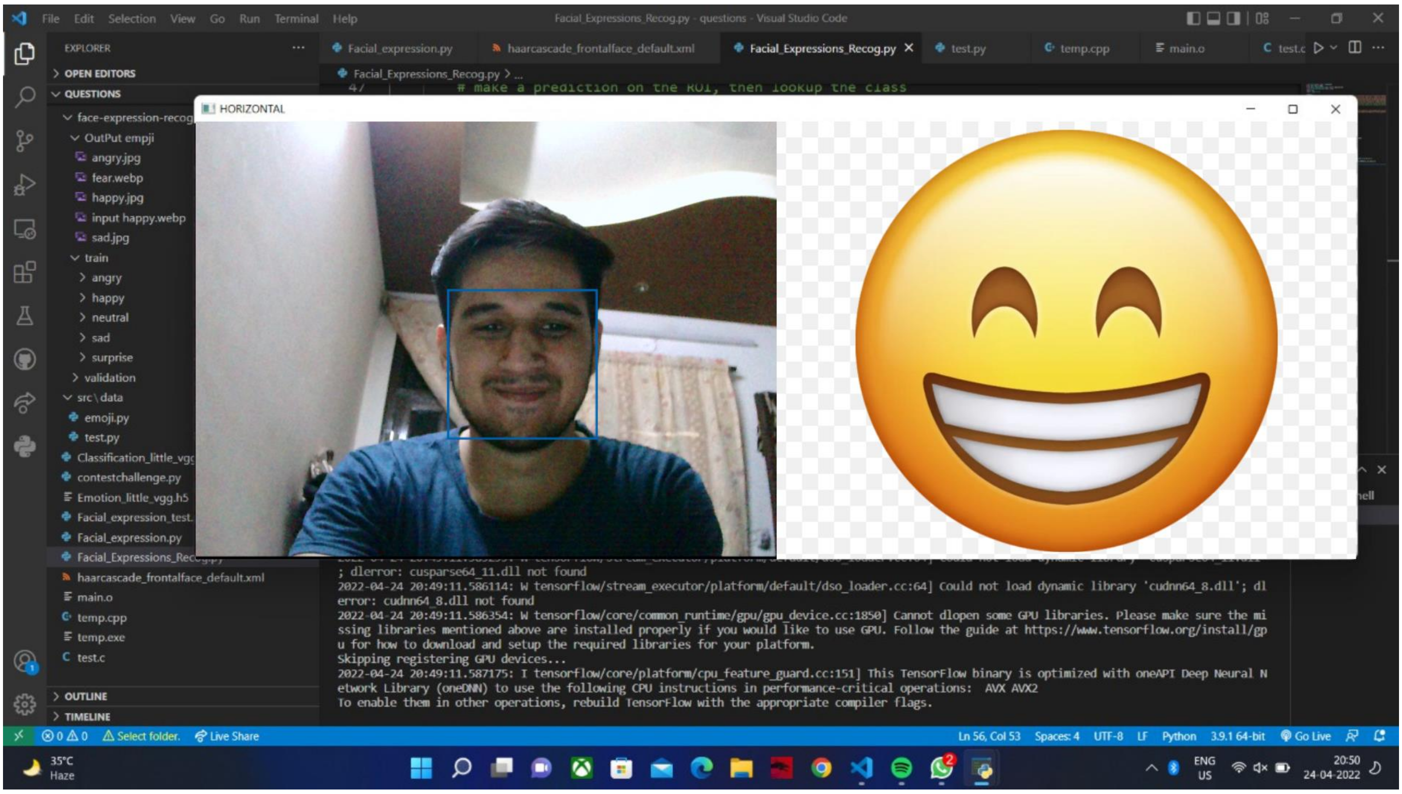
Fig. 1: Plot of Example images

# IMPLEMENTATION & TESTING

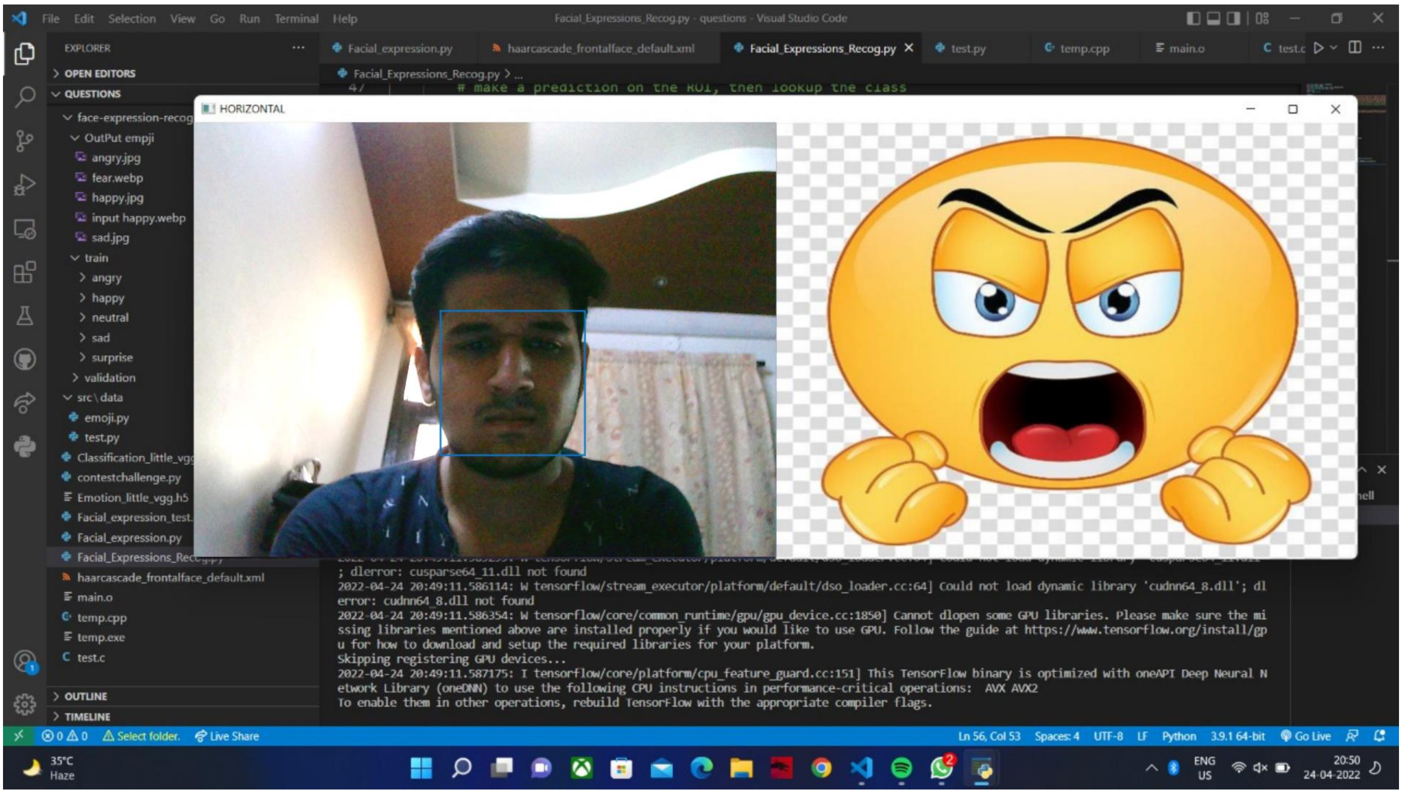
## CNN Model

The CNN designed is based on sequential model and is designed to have six activation layers, of which 4 are convolutional layers and the remaining 2 are fully controlled layers.

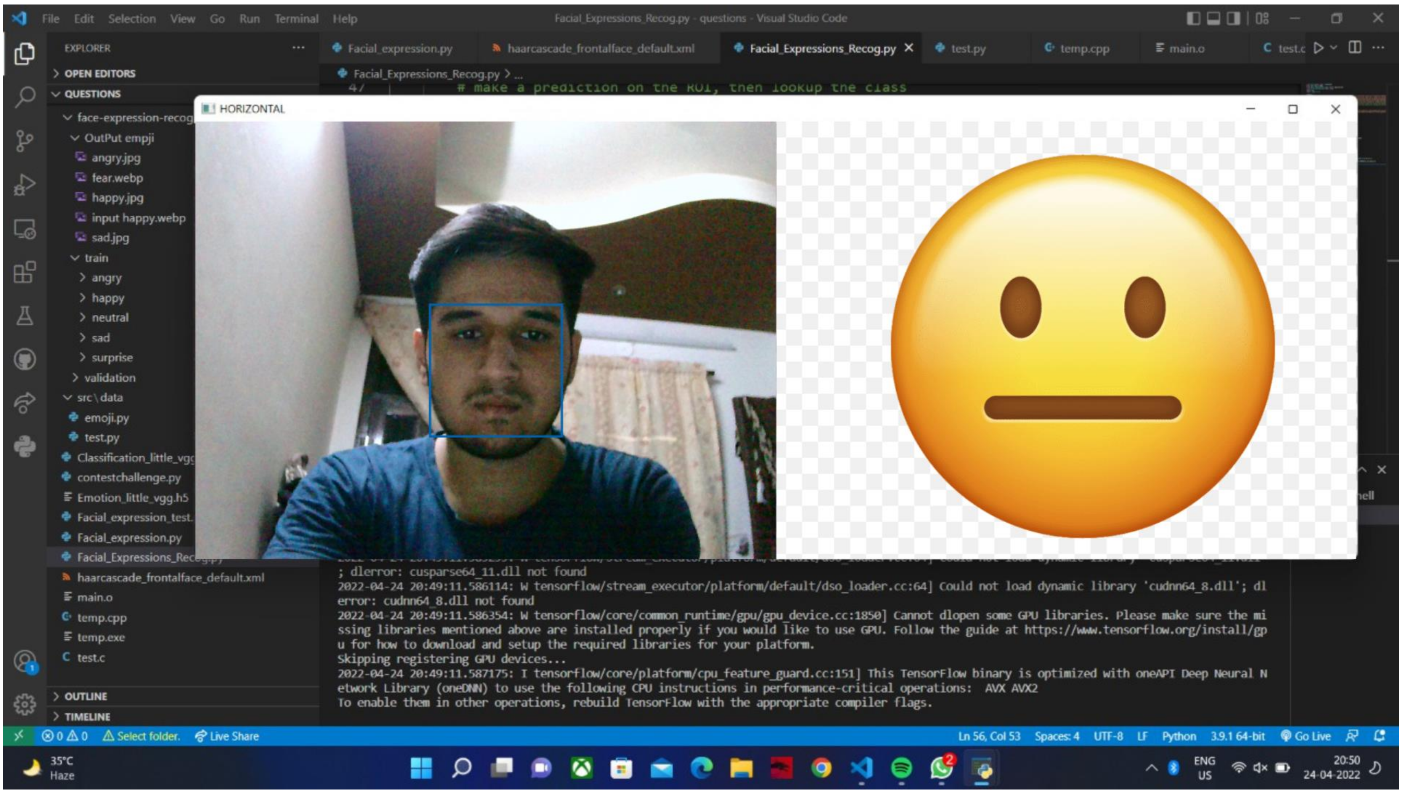



- **Happy Emoji**




- **Angry Emoji**




- **Neutral Emoji**

# RESULT

## ACCURACY

The CNN model designed is set to undergo 25 epochs. When trained gives an accuracy of 49% after the 25th epoch and the maximum efficiency achieved is also 56%.

It has been observed that the CNN model developed can easily or flawlessly detect the following emotion types: Happy, Neutral, Sad and Surprise. The lighting conditions or the Illumination setup, play a major part in image detection and is a very important factor in the model’s performance.

## RELATED WORK

This is the SUMMARY printed. – print(model.summary())

| Layer (Type) | Output Shape | Parameter |
|---|---|---|
| **Conv2d_1 (Conv2D)** | (None, 48, 48, 32) | 320 |
| **Activation_1 (Activation)** | (None, 48, 48, 32) | 0 |
| **Batch_normalization_1 (Batch Normalization)** | (None, 48, 48, 32) | 128 |
| **Conv2d_2 (Conv2D)** | (None, 48, 48, 32) | 9248 |
| **Activation_2 (Activation)** | (None, 48, 48, 32) | 0 |
| **Batch_normalization_2 (Batch Normalization)** | (None, 48, 48, 32) | 128 |
| **Max_pooiling2d_1 (MaxPooling2D)** | (None, 24, 24, 32) | 0 |
| **Dropout_1 (Dropout)** | (None, 24, 24, 32) | 0 |
| **Conv2d_3 (Conv2D)** | (None, 24, 24, 64) | 18496 |
| **Activation_3 (Activation)** | (None, 24, 24, 64) | 0 |
| **Batch_normalization_3 (Batch Normalization)** | (None, 24, 24, 64) | 256 |
| **Conv2d_4 (Conv2D)** | (None, 24, 24, 64) | 36928 |
| **Activation_4 (Activation)** | (None, 24, 24, 64) | 0 |
| **Batch_normalization_4 (Batch Normalization)** | (None, 24, 24, 64) | 256 |
| **Max_pooiling2d_2 (MaxPooling2D)** | (None, 24, 24, 64) | 0 |
| **Dropout_2 (Dropout)** | (None, 12, 12, 64) | 0 |
| **Conv2d_5 (Conv2D)** | (None, 12, 12, 128) | 73856 |
| **Activation_5 (Activation)** | (None, 12, 12, 128) | 0 |
| **Batch_normalization_5 (Batch Normalization)** | (None, 12, 12, 128) | 512 |
| **Conv2d_6 (Conv2D)** | (None, 12, 12, 128) | 147584 |
| **Activation_6 (Activation)** | (None, 12, 12, 128) | 0 |
| **Batch_normalization_6 (Batch Normalization)** | (None, 12, 12, 128) | 512 |
| **Max_pooiling2d_3 (MaxPooling2D)** | (None, 6, 6, 128) | 0 |
| **Dropout_3 (Dropout)** | (None, 6, 6, 128) | 0 |
| **Conv2d_7 (Conv2D)** | (None, 6, 6, 256) | 295168 |
| **Activation_7 (Activation)** | (None, 6, 6, 256) | 0 |
| **Batch_normalization_7 (Batch Normalization)** | (None, 6, 6, 256) | 1024 |
| **Conv2d_8 (Conv2D)** | (None, 6, 6, 256) | 590080 |
| **Activation_8 (Activation)** | (None, 6, 6, 256) | 0 |
| **Batch_normalization_8 (Batch Normalization)** | (None, 6, 6, 256) | 1024 |
| **Max_pooiling2d_4 (MaxPooling2D)** | (None, 3, 3, 256) | 0 |
| **Dropout_4 (Dropout)** | (None, 3, 3, 256) | 0 |
| **Flatten_1 (Flatten)** | (None, 2304) | 0 |
| **Dense_1 (Dense)** | (None, 64) | 147520 |
| **Activation_9 (Activation)** | (None, 64) | 0 |
| **Batch_normalization_9 (Batch Normalization)** | (None, 64) | 256 |
| **Dropout_5 (Dropout)** | (None, 64) | 0 |
| **Dense_2 (Dense)** | (None, 64) | 4160 |
| **Activation_10 (Activation)** | (None, 64) | 0 |
| **Batch_normalization_10 (Batch Normalization)** | (None, 64) | 256 |
| **Dropout_6 (Dropout)** | (None, 64) | 0 |
| **Dense_3 (Dense)** | (None, 5) | 325 |
| **Activation_11 (Activation)** | (None, 5) | 0 |

# CONCLUSION

Using the FER-2013 dataset, a test accuracy of 56% is attained with this designed CNN model. The achieved results are satisfactory as the average accuracies on the FER-2013 dataset is 65% +/- 5% and therefore, this CNN model is nearly accurate. For an improvement in this project and its outcomes, it is recommended to add new parameters wherever useful in the CNN model and removing unwanted and not-so useful parameters. Adjusting the learning rate and adapting with the location might help in improving the model. Accommodating the system to adapt to a low graded illumination setup and nullify noises in the image can also add onto the efforts to develop the CNN model. Increasing the layers in the CNN model might not deviate from the achieved accuracy, but the number of epochs can be set to higher number, to attain a higher accurate output. Though, increasing the number of epochs to a certain limit, will increase the accuracy, but increasing the number of epochs to a higher value will result in over- fitting.

## FUTURE WORK

1. At this level, we have included only 5 emojis to mask our facial expressions, but our future work would include far more emojis so that we can map emojis as per the expressions accurately.

2. To improve the accuracy of our model. In order to match it to the average accuracy of FER dataset.

3. To integrate this model with the application and make a feedback emoji SAS as they are one of the most fast-growing and common ways of gathering feedback that companies of all sizes are now using. For business development, customer feedback is significant. It helps you to substantially enhance the customer experience, which directly affects financial growth.