IMAGE PROCESSING
AND
ONLINE RECOMMENDATION OF PRODUCTS USING ARTIFICIAL INTELLIGENCE TECHNIQUES

Submitted as Partial Fulfillment of Bachelor of Technology in Computer Science & Engineering
Of
Maulana Abul Kalam Azad University of Technology
(Formerly known as West Bengal University of Technology)

Project Report

Submitted by –

Names of Students	**University Roll No.**
Saloni Banerjee	**11600117021**
Sidra Anwar	**11600117015**
Roshan Chongder	**11600217022**
Ranojoy Paul	**11601617041**
Aritra Chakraborty	**11600717041**

Under Supervision of –

Mrs. Soma Bandyopadhyay
Assistant Professor, Computer Science & Engineering Department

Department of Computer Science & Engineering,
MCKV Institute of Engineering
243, G.T. Road(N)
Liluah, Howrah – 711204

Department of Computer Science & Engineering
MCKV Institute of Engineering
243, G. T. Road (N),
Liluah, Howrah-711204

CERTIFICATE OF RECOMMENDATION

I hereby recommend that the thesis prepared under my supervision by Saloni Banerjee, Sidra Anwar, Roshan Chongder, Ranojoy Paul, Aritra Chakraborty entitled Image Processing and Online Recommendation of Products using Artificial Intelligence Techniques, be accepted in partial Fulfillment of the requirements for the degree of Bachelor of Technology in Computer Science & Engineering Department .

Mr. Avijit Bose
Head of the Department
Computer Science & Engineering
Department,
MCKV Institute of Engineering, Howrah

Mrs. Soma Bandyopadhyay
Project Guide & Assistant Professor
Computer Science & Engineering
Department,
MCKV Institute of Engineering, Howrah

MCKV Institute of Engineering
243, G. T. Road (N), Liluah
Howrah-711204

Affiliated to

Maulana Abul Kalam Azad University of Technology
(Formerly known as West Bengal University of Technology)

CERTIFICATE

This is to Certify that the Project titled “Image Processing and Online Recommendation of Products using Artificial Intelligence Techniques”, submitted by -

Name of students	**University Roll No.**
1. Saloni Banerjee	11600117021
2. Sidra Anwar	11600117015
3. Roshan Chongder	11600217022
4. Ranojoy Paul	11601617041
5. Aritra Chakraborty	11600717041

has been carried out under the guidance of myself following the rules and regulations of the degree of Bachelor of Technology in Computer Science & Engineering of **Maulana Abul Kalam Azad University of Technology** (Formerly West Bengal University of Technology).

____

Mrs. Soma Bandyopadhyay
Project Guide & Asst. Professor,
Computer Science & Engineering Dept.

1. Saloni Banerjee
2. Sidra Anwar
3. Roshan Chongder
4. Ranojoy Paul
5. Aritra Chakraborty

MCKV Institute of Engineering
243, G. T. Road (N), Liluah
Howrah-711204
Affiliated to
Maulana Abul Kalam Azad University of Technology
(Formerly known as West Bengal University of Technology)

CERTIFICATE OF APPROVAL
(B. Tech Degree in Computer Science & Engineering)

This project report is hereby approved as a creditable study of an Engineering subject carried out and presented in a manner satisfactory to warrant its acceptance as a prerequisite to the degree for which it has been submitted. It is to be understood that by this approval, the undersigned do not necessarily endorse or approve any statement made, opinion expressed and conclusion drawn therein but approve the project report only for the purpose for which it has been submitted.

COMMITTEE ON FINAL EXAMINATION FOR EVALUATION OF PROJECT REPORT

1. ______________________________
2. ______________________________
3. ______________________________
4. ______________________________
5. ______________________________

ACKNOWLEDGEMENT

We're well in the Era of Technology, but this project is consolidating and polishing our age-old skills of Time Management, Teamwork and Perseverance.

We express our sincere gratitude to Mr. Avijit Bose, Head of Department, Computer Science and Engineering Department, MCKV Institute of Engineering, Liluah. We receive ceaseless enthusiasm, constant guidance and judicious encouragement from our Project Guide Mrs. Soma Bandyopadhyay. They are the constant pillars of support behind our efforts, despite all odds. We thank them for their hindsight every step of the way.

All team members have industriously contributed towards and enjoyed sculpting this enriching Project, taking away motivation to strive for progress for all future endeavors.

CONTENTS

ABSTRACT

Human Expressions are key to analyzing Human Emotions, and in turn are a capacitive tool when utilized to target products and services as a recommendation to improve the standard of living and generate better revenue by advertising a significantly personal and relevant array of customized recommendations, to interest them in an investment.

A major segment of the dynamics of a product or service oriented platform depends on the remuneration incurred through successful sale and greater client satisfaction. This expression-based online recommendation system is an initiative towards creating a hassle-free environment, yet a promising tool to attract and retain potential consumers in the emerging premise of online platforms. Online platforms are known to uphold their reputation with a broader catalog of choices, competitive pricing, and greater albeit more transparent access to information. However, the prime motivation behind an expression-based online recommendation of products is to improve privacy policies by limiting the compromise of customer data in the process of generating targeted advertisements, thus providing a safe shopping experience online.

While basic machine learning models do become progressively better at whatever their function is, they still need some guidance. Beside that if an AI algorithm returns an inaccurate prediction, then we have to step in and make adjustments. With a deep learning model, an algorithm can determine on its own whether a prediction is accurate or not through its own neural network. Although Deep Learning is a subfield of Machine Learning, both fall under the broad category of Artificial Intelligence.

In recent years, deep learning using feature extraction of image data based on Convolutional Neural Networks (CNN) has become more and more popular. Their popularity stems from their ability to extract good features from image data, for CNN's computationally intensive tasks can be run on the GPU to achieve high performance at very low consumption. This algorithm can achieve much higher accuracy than traditional ones, making it possible to commercialize and utilize. Combining these new analytical tools with modern databases and registries promise an entirely new approach towards predicting human emotion based on facial features and ideally, developing ways to predict the reaction of a customer on purchasing a product.

In this project we have tried to design a model for emotion detection using facial expression. For human-computer interaction, facial expression makes a platform for non-verbal communication. The emotions are effectively changeable happenings that are evoked as a result of impelling force. So in real life application, detection of emotion is a very challenging task. Facial expression recognition systems require the human face to have multiple variability such as color, orientation, expression, posture and texture. In our model we have taken a specific dataset with tagged images with their emotions and processed them using convolutional neural networks to train the model about different features of the image with respect to their specific tags. Then frames from live streaming and fed to the trained model for prediction.

1. INTRODUCTION

Facial expression is an important part of non-verbal communication and one common means of human communication. Expression recognition, as one of the important development directions of human-computer interaction, can improve the fluency, accuracy and naturalness of interaction. Research has shown that over 90 percent of our communication can be non-verbal, but technology has struggled to keep up, and traditional code is generally bad at understanding our intonations and intentions[1]. Facial emotion recognition is the process of detecting human emotions from facial expressions. The human brain recognizes emotions automatically, and software has now been developed that can recognize emotions as well. This technology is becoming more accurate all the time, and will eventually be able to read emotions as well as our brains do [2].

AI can detect emotions by learning what each facial expression means and applying that knowledge to the new information presented to it. Emotional artificial intelligence is a technology that is capable of reading, imitating, interpreting, and responding to human facial expressions and emotions[3]. Neural networks are a set of algorithms, modeled loosely after the human brain, that are designed to recognize patterns. They interpret sensory data through a kind of machine perception, labelling or clustering raw input. The patterns they recognize are numerical, contained in vectors, into which all real-world data, be it images, sound, text or time series, must be translated. In a traditional artificially connected neural network, each neuron between each adjacent two layers is connected. When the feature dimension of the input layer becomes very high, the parameters that need to be trained in the fully connected network will increase a lot, and the calculation speed will become very slow. Recently, the introduction of convolutional neural network algorithms (CNNs) has greatly improved the speed, accuracy and robustness of facial expression recognition, making it possible to apply this technology to innovative design and manufacturing. CNNs have good feature extraction capabilities and can run on GPU. It has high efficiency and low power consumption [4]. In the Convolutional Neural Network (CNN), the convolutional layer neurons are only connected to some of the neurons in the previous layer, that is, the connections between their neurons are not fully connected, and in the same layer. The weights and offsets of the connections between certain neurons are shared, thus greatly reducing the number of training parameters required. The convolutional neural network structure generally contains these layers, including Input layer, Convolutional layer, Excitation layer, Pooling layer, Fully connected layer, Output layer, Normal Normalization layer and Fusion layer[5]. Neural networks help us cluster and classify. It can be considered as a clustering and classification layer on top of the data stored and managed. They help to group unlabelled data according to similarities among the example inputs, and they classify data when they have a labelled dataset to train on[6].

2. MOTIVATION OF WORK

In the field of product design, especially in human computer interaction design, recognizing facial expression information and judging people's emotions have become the frontier of research. In the 1970s, psychologists Ekman and Keltner proposed the facial motion coding system (FACES) [7] based on the motion unit (Aus) to describe facial expressions.

Actually, facial muscle generates monetary adaptation in facial appearance which can be recapitulated by incorporating action units. The five common emotions have been considered as globally recognizable as the movements of muscle are very similar to these emotional expressions from the people from various regions and society. Therefore, we have mainly concentrated on the automatic recognition of these five fundamental emotions,i.e. Happy, Sad, Angry, Surprise and Disgust. In general, emotion recognition is a two steps procedure which involves extraction of significant features and classification. Feature extraction determines a set of independent attributes, which together can portray an expression of facial emotion. For classification in emotion recognition the features are mapped into either of various emotion classes like anger, happy, sad, neutral, surprise, etc. For the effectiveness calculation of a facial expression identification model both the group of feature attributes which have been taken for feature extraction[8] and the classifier that is responsible classification are equivalently significant. In the present era of Internet technology E-commerce businesses are growing rapidly. Recommender system plays a significant role to suggest product recommendation to customer/user. Customer’s general buying behavior may not be influenced and controlled by the brand and firm, while placing an order or a request. They are influenced by interactions with search engines, recommendations, online reviews and other information while navigating around the digital environment. Facial expression and emotion analysis can help the online shoppers to capture customer's emotions and help them to choose the right products[9].

CNN based methods are known to achieve state-of-the-art accuracy in various image classification challenges. For ICML 2013 workshop contest named Challenges in Representation Learning later hosted on kaggle, one of the top performing entries used CNN with SVM to achieve state-of-the-art accuracy of 71% on FER-2013 dataset.This work makes an attempt to arrive at a CNN architecture specifically for FER-2013 by making an extensive study of variation of accuracy with CNN[10] parameters.

Now for a badly picked collection of feature attributes, in some cases, even a smart classification mechanism is not able to produce an ideal outcome. Thus, for getting the high classification accuracy and qualitative outcome, picking of superior features will play a major role[8].

3. RELATED WORKS

In emotional recognition of face a notable advancement has been observed in the field of neuroscience, cognitive and computational intelligence [11][12]. It is also proved by Kharat and Dudul that about 55% effect of overall emotion expression is as facial expression which is contributed during social interactions.

Krizhevsky in 2012 trained a CNN architecture, popularly known as AlexNet, on GPU to achieve good accuracy on the ILSVRC-2010 dataset. After that, there has been increasing research to improve the original architecture of Krizhevsky to achieve better accuracies on Imagenet dataset. The original model by Krizhevsky had convolution layers with kernel sizes 11, 5 and 3 followed by fully connected layers. This architecture was a unique combination of convolution layers followed by max-pool operations. AlexNet architecture also involved use of response normalization for the first few layers. Although the network achieved good accuracy, the choice of its parameters was empirical which left a scope for further optimization[10] of the network.

DAGER(Deep Age, Gender and Emotion Recognition Using Convolutional Neural Networks) proposed by Afshin Dehghan, Enrique G. Ortiz, Guang Shu and Syed Zain Masood describe the details of a fully automated age, gender and emotion recognition system [13].

The circumflex model by Russell and recognition of six basic emotions are having remarkable contributions in the field of emotion recognition. Other than this, the work by Kudiri M. Krishna, Said Abas Md, Nayan M Yunus, where they tried to detect emotion by using the concept of sub-image based features through facial expression. Silva C. DE Liyanage, Miyasato Tsutomu, Nakatsu Ryohei, they formed a model for emotion recognition with the help of multimodal information. Maja Pantic, Ioannis Patras, implemented an approach for recognition of facial action and temporal segment from fare profile image sequences[8] by considering the dynamic property of facial action. Li Zhang, Ming Jiang, Dewan Farid, M.A. Hassain, they modelled an intelligent system for automatic emotion recognition. Happy S L, Routray Aurobinda, created an automatic emotion recognition system by using salient features. This article is greatly influenced by all of this contribution in the field of emotion recognition.

4. OVERVIEW OF MACHINE LEARNING

A Machine Learning system learns from historical data, builds the prediction models, and whenever it receives new data, predicts the output for it. The accuracy of predicted output depends upon the amount of data, as the huge amount of data helps to build a better model which predicts the output more accurately [14]. Different kinds of Recommender Systems comprise of Association Rule Mining-Market Basket Analysis, Collaborative Filtering, Clustering, etc.

4.1. WORKING OF NEURAL NETWORK

Deep learning is the name we use for “stacked neural networks”, that is, networks composed of several layers. The layers are made of nodes. A node is just a place where computation happens, loosely patterned on a neuron in the human brain, which fires when it encounters sufficient stimuli. A node combines input from the data with a set of coefficients, or weights, that either amplify or dampen that input, thereby assigning significance to inputs with regard to the task the algorithm is trying to learn[15]. These input-weight products are summed and then the sum is passed through a node’s so-called activation function, to determine whether and to what extent that signal should progress further through the network to affect the ultimate outcome.

If the signal passes through, the neuron has been “activated”. A better representation of a single node is illustrated in Figure 1 :

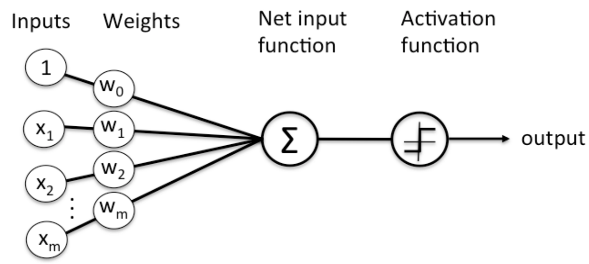

Figure 1:Node of a Neural network

A node layer is a row of those neuron-like switches that turn on or off as the input is fed through the net. Each layer’s output is simultaneously the subsequent layer’s input, starting from an initial input layer receiving the data [16].

4.2. CONVOLUTIONAL NEURAL NETWORK

A Convolutional Neural Network (ConvNet/CNN) is a Deep Learning algorithm which can accept an image as input, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other. The pre-processing required in a ConvNet is much lower as compared to other classification algorithms. While in primitive methods filters are hand-engineered, with enough training [17]. Figure 2 depicts the single layer of a CNN.

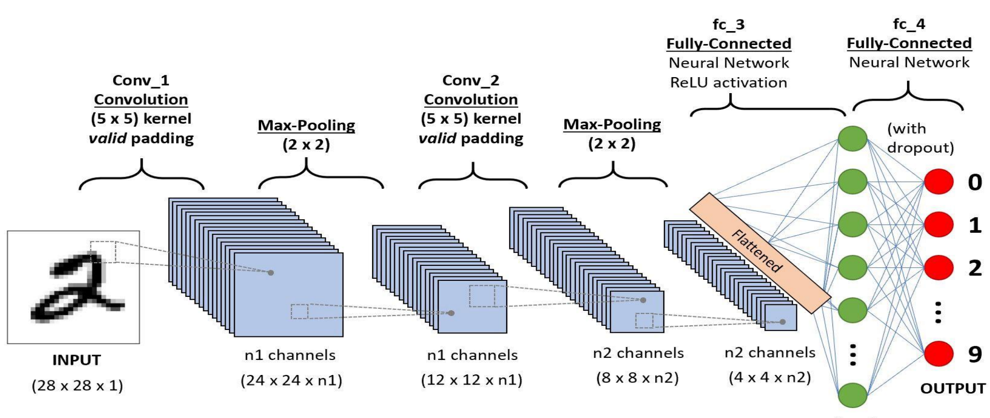

Figure 2:Single layer of a CNN

5. DEPENDENCIES

KERAS

Keras is a powerful and easy-to-use free open source Python library for developing and evaluating deep learning models. It wraps the efficient numerical computation libraries Theano and TensorFlow and allows defining and training neural network models in just a few lines of code. Keras is an API designed for human beings, not machines. This makes Keras easy to learn and easy to use[18].

PANDAS

Pandas has been one of the most popular and favourite Data Science tools used in Python programming language for data wrangling and analysis. Data is unavoidably messy in the real world. It is a powerful tool for cleaning, transforming, manipulating and analyzing data [19].

NUMPY

NumPy is a Python package, which stands for 'Numerical Python', and is a library consisting of multidimensional array objects and a collection of routines for processing of arrays. Numeric, the ancestor of NumPy, was developed by Jim Hugunin[20].

Using NumPy, a developer can perform the following operations −

- Mathematical and logical operations on arrays.
- Fourier transforms and routines for shape manipulation.
- Operations related to linear algebra. NumPy has in-built functions for linear algebra and random number generation.

TENSORFLOW

TensorFlow provides a collection of workflows to develop and train models using Python or JavaScript, and to easily deploy in the cloud, on-prem, in the browser, or on-device no matter what language is used. The TensorFlow data API enables building complex input pipelines from simple, reusable pieces. It allows developers to create large-scale neural networks with many layers. TensorFlow is mainly used for: Classification, Perception, Understanding, Discovering, Prediction and Creation [21].

KERNEL

An image processing kernel is a convolution matrix or mask which can be used for blurring, sharpening, embossing, edge detection, and more by doing a convolution between a kernel and an image [22].

KERAS CONV2D

It is a 2D Convolution Layer, this layer creates a convolution kernel that is wind with layers input which helps in producing a tensor of outputs.

FILTERS

It's an integer that defines the number of features from which the convolution layer will learn.

STRIDES

It's an integer or a set of two integers that specifies the step of convolution along with the height and width of the input.

DROPOUT LAYER

Dropout regularization is a computationally cheap way to regularize a deep neural network. Dropout works by probabilistically removing, or “dropping out,” inputs to a layer, which may be input variables in the data sample or activations from a previous layer. It has the effect of simulating a large number of networks with very different network structure and, in turn, making nodes in the network generally more robust to the inputs [26].

MAXPOOL 2D

Maximum pooling, or max pooling, is a pooling operation that calculates the maximum, or largest, value in each patch of each feature map. The results are down-sampled or pooled feature maps that highlight the most present feature in the patch, not the average presence of the feature in the case of average pooling [23].

MAX POOLING

It is a pooling operation that selects the maximum element from the region of the feature map covered by the filter. Thus, the output after max-pooling layer would be a feature map containing the most prominent features of the previous feature map [24]. The process of Max Pooling is exemplified in Figure 3.

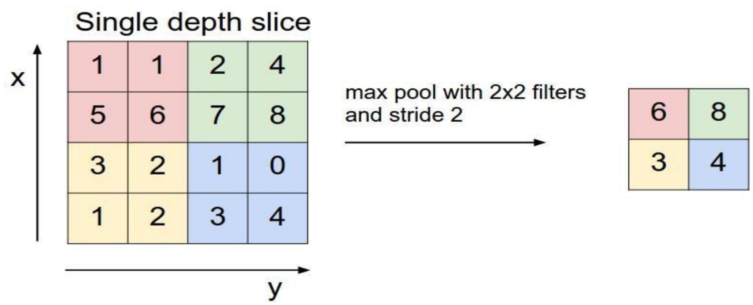

Figure 3: Instance of Pooling

ReLU

It refers to the Rectifier Unit, the most commonly deployed activation function for the outputs of the CNN neurons. Mathematically, it is described as: Unfortunately, the ReLU function is not differentiable at the origin, which makes it hard to use with backpropagation training[25]. Figure 4 illustrates the functionality of ReLU.

ReLU is a half rectifier:

f(y) = 0 , while y < 0;
f(y) = y , while y >=0

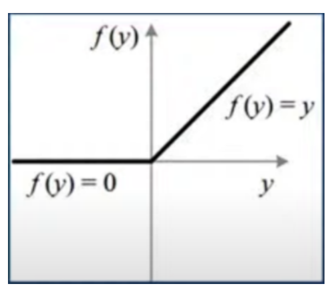

Figure 4: ReLU plot

6. DATA SET REPRESENTATION

6.1. DATA SET DESCRIPTION

The data consists of 48x48 pixel grayscale 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 into one of five categories (Angry, Happy, Sad, Surprise, Neutral) and the train.csv contains two columns, "emotion" and "pixels" respectively. The "pixels" column contains a string surrounded in quotes for each image. The contents of this string are space-separated pixel values in row major order. The file test.csv contains only the "pixels" column and our task is to predict the emotion column [27].The following facial emotions that we have worked with are shown in Table 1 with corresponding tags.

Emotion	Angry	Happy	Sad	Surprise	Neutral
Tag	0	3	4	5	6

Table 1: Tags corresponding to each emotion

The total data used for Training and Testing are 20962 and 6043 respectively. This dataset was prepared by Pierre-Luc Carrier and Aaron Courville.

6.2. DATA SET REPRESENTATION

Figure 5 shows different types of images present in the dataset.

Figure 5 : Images in the Data Set used

Figure 6 shows the method to read data from the csv file as pandas data frame.

```python
# Method to read CSV Files
def read_CSV( path ):
    try:
        file = pd.read_csv( path )
        return file
    except FileNotFoundError:
        print( "CSV File not found at " + path )
        return None
    except Exception:
        print(" Unknown error appeared ")
        return None

# reading the csv file

data_set = read_CSV('drive/MyDrive/EmotionDetection/fer2013/fer2013.csv')
print( data_set.info() ) # checking info of the data set
print( data_set.head())  # showing the first ew line of the data set
```

Figure 6: Method to read CSV

The data set contains about 27005 entries total. There are mainly three columns which are "Emotions", "Pixels" and "Usage". The emotion column contains a tag for the actual emotion, pixels hold 48x48 pixels representation and the usage column holds the usage of that entry, indicating whether it is for training or testing [27]. Figure 7 shows how Data Frames are represented in Pandas.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 35887 entries, 0 to 35886
Data columns (total 3 columns):
 #   Column   Non-Null Count  Dtype
---  ------   --------------  -----
 0   emotion  35887 non-null  int64
 1   pixels   35887 non-null  object
 2   Usage    35887 non-null  object
dtypes: int64(1), object(2)
memory usage: 841.2+ KB
None
   emotion                                             pixels     Usage
0        0  70 80 82 72 58 58 60 63 54 58 60 48 89 115 121...  Training
1        0  151 150 147 155 148 133 111 140 170 174 182 15...  Training
2        2  231 212 156 164 174 138 161 173 182 200 106 38...  Training
3        4  24 32 36 30 32 23 19 20 30 41 21 22 32 34 21 1...  Training
4        6  4 0 0 0 0 0 0 0 0 0 0 0 3 15 23 28 48 50 58 84...  Training
```

Figure 7: Representation of data frames in Pandas

6.3 DATA SANITIZATION AND CLEANING

Depending on the ‘usage’ column, pixels and emotions are extracted from the csv file for training and testing purposes. Cleaning the data is a part of pre-processing the data, but in our case the dataset used from Kaggle was already clean. Here, a dictionary has been used to count the number of images for each tag. Doing so, it was found that ‘Happy’ which has a tag as (3) was having a higher ratio of training data compared to others. So the overall model was going on a tilt towards ‘Happy’. Thus the number of data is restricted to around 4000 to bring equilibrium in the training dataset for all the emotions. As the keras module only takes numpy arrays as input parameters, the lists are required to be converted into numpy arrays. After proper rescalling of x_train and y_train and then reshaping [28] them into a 1-dimensional array, we move forward towards the main attraction of the project i.e. designing the CNN. Table 2 shows the count of training and validation data set and Figure 8 shows the method to extract training and validating data set as a list from pandas data frame.

Emotion	Angry	Happy	Sad	Surprise	Neutral
Training Count	3995	4001	4830	3171	4965
Validation Count	958	1774	1247	831	1233

Table 2 : Training And Validation Data Count

```
def data_Addition() :
    global x_train , y_train , x_test , y_test , data_set
    d = {3:0}
    for row_count,row in data_set.iterrows():
        if row['emotion'] in [1,2] or ( row['emotion'] == 3  and d[ row['emotion'] ] > 4000  and row['Usage'] == 'Training' ) : continue
        value = row['pixels'].split(' ')   # extracting the pixels as a list
        try :
            if 'Training' in row['Usage'] :                   # if the current column is for Training

                x_train.append(np.array(value,'float32'))       # adding the pixels in x axis
                y_train.append(row['emotion'])                  # adding the actual emotion tag in y axis

                if row['emotion'] in d :
                    d[ row['emotion'] ]+=1
                else :
                    d[ row['emotion'] ] = 1
            else :

                x_test.append(np.array( value,'float32'))
                y_test.append( row['emotion'])

        except:
            print(" Error occurred at row number " + row_count)
            print("Data Set in that row is " + row )

    print(d)

# Now we will do training and public testing

x_train , y_train = [] , []   # data the will be used for training will added in this two lists
x_test , y_test = [] , []     # data that will be used for public testing will be added here

data_Addition()    # addition of data in the lists for training and testing

print( "Training : " , len(x_train) , "Testing : " , len(x_test) )
```

Figure 8: Data extraction from CSV File

7. IMPLEMENTATION OF THE PROPOSED WORK

7.1. FLOWCHARTS OF THE PROPOSED WORK

For all the purposes of training, testing and validation, google colaboratory environment is used. At first, the data set which is collected from Kaggle is read from the csv file. After loading the data set, data extraction is done for training and validation depending on the ‘usage’ tag. Then the CNN model is built by adding different convolution and other layers. As per the extracted data set labeled for training and validation, the built model is trained using hardware acceleration as GPU on a certain number of epochs. After the training, training and validation accuracy and loss graphs have been plotted. To compute different scores of the trained model, the confusion matrix is plotted on the prediction of the trained model on some random images collected from google. At last the trained model and its weights are loaded for testing. Then from a live video stream, frames are captured and fed to the model for prediction and the predicted outputs are shown on the stream. For the initial part of the project we are going to do the following:

1. Collecting dataset for various emotions
    i. Classification of different Emotion classes to prepare the dataset.
2. Applying CNN for Training the prediction model
    i. Importing the required libraries
    ii. Adding different layers to the architecture to our model
    iii. Using images and labels for training the model
    iv. Training the model using Keras; Tensor flow based on given parameters
3. Validating the efficiency of our model using Confusion matrix
    i. Building Confusion matrix based on Actual and Predicted emotions
    ii. Computing its accuracy, precision and recall
4. Emotion of the customer is captured from the video clip .
5. Facial emotions are predicted using the trained model for further use.

Figure 9 describes the flow of control for Model Training, Figure 10 illustrates the generation of Confusion Matrix and Performance Measurement, Figure 11 shows the steps leading the model to generate Predictions and Figure 12 shows the Testing of our model with images extracted from live video stream.

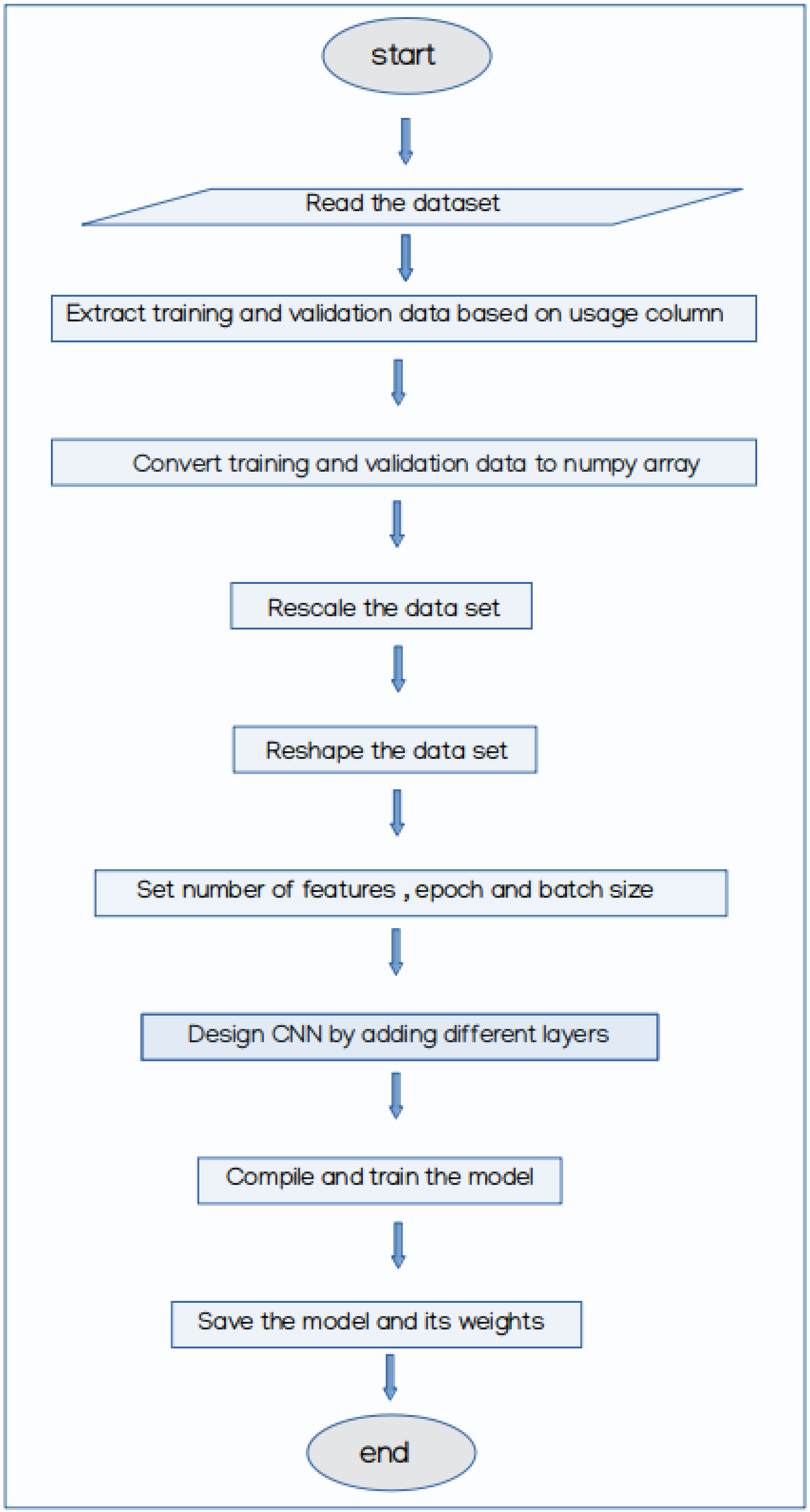

Figure 9: Flow of Training the model

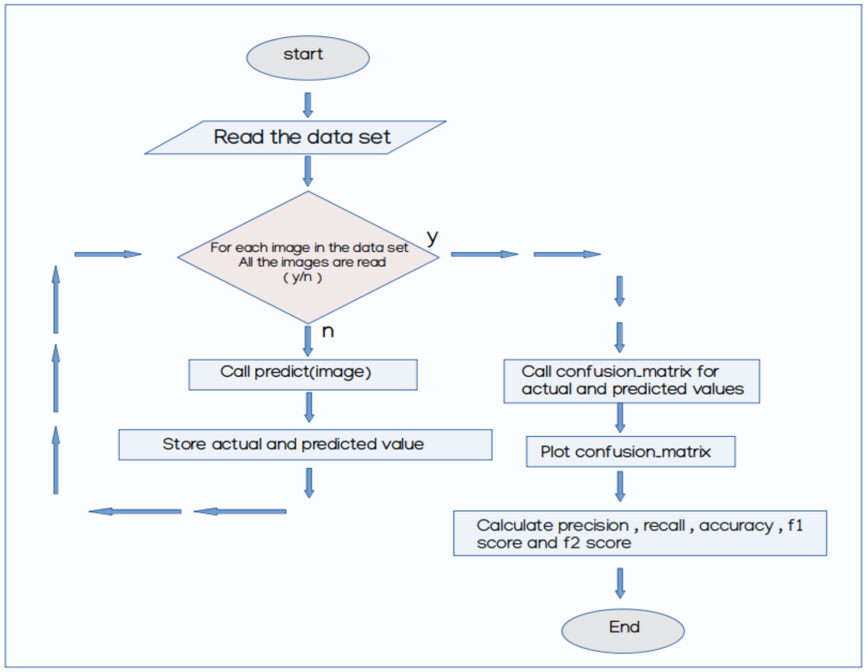

Figure 10: Flow of Confusion Matrix generation and Performance Analysis

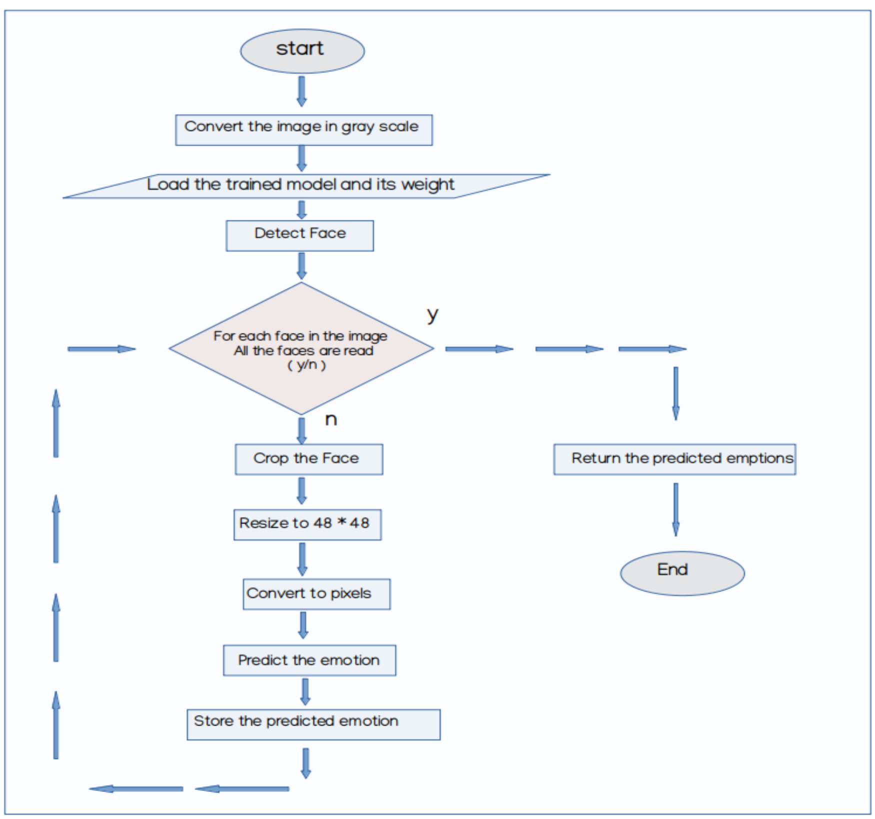

Figure 11: Flow of the prediction for a certain image

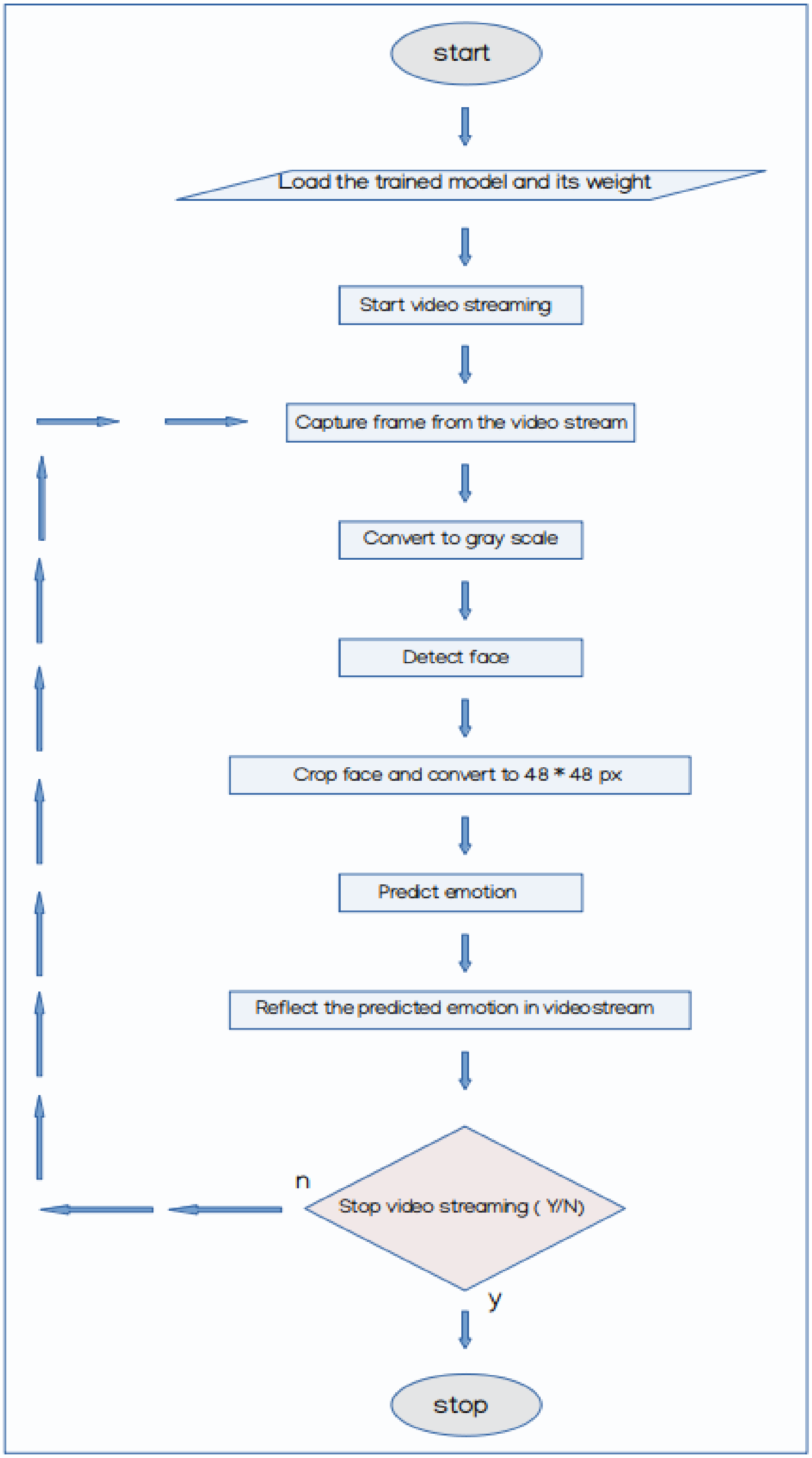

Figure 12: Flow of Testing the model on real time images

7.2 ARCHITECTURE OF A CONVOLUTIONAL NEURAL NETWORK

The architecture of a ConvNet is analogous to that of the connectivity pattern of Neurons in the Human Brain and was inspired by the organization of the Visual Cortex. Individual neurons respond to stimuli only in a restricted region of the visual field known as the Receptive Field. A collection of such fields overlap to cover the entire visual area [29]. Figure 13 shows the architectural diagram of CNN.

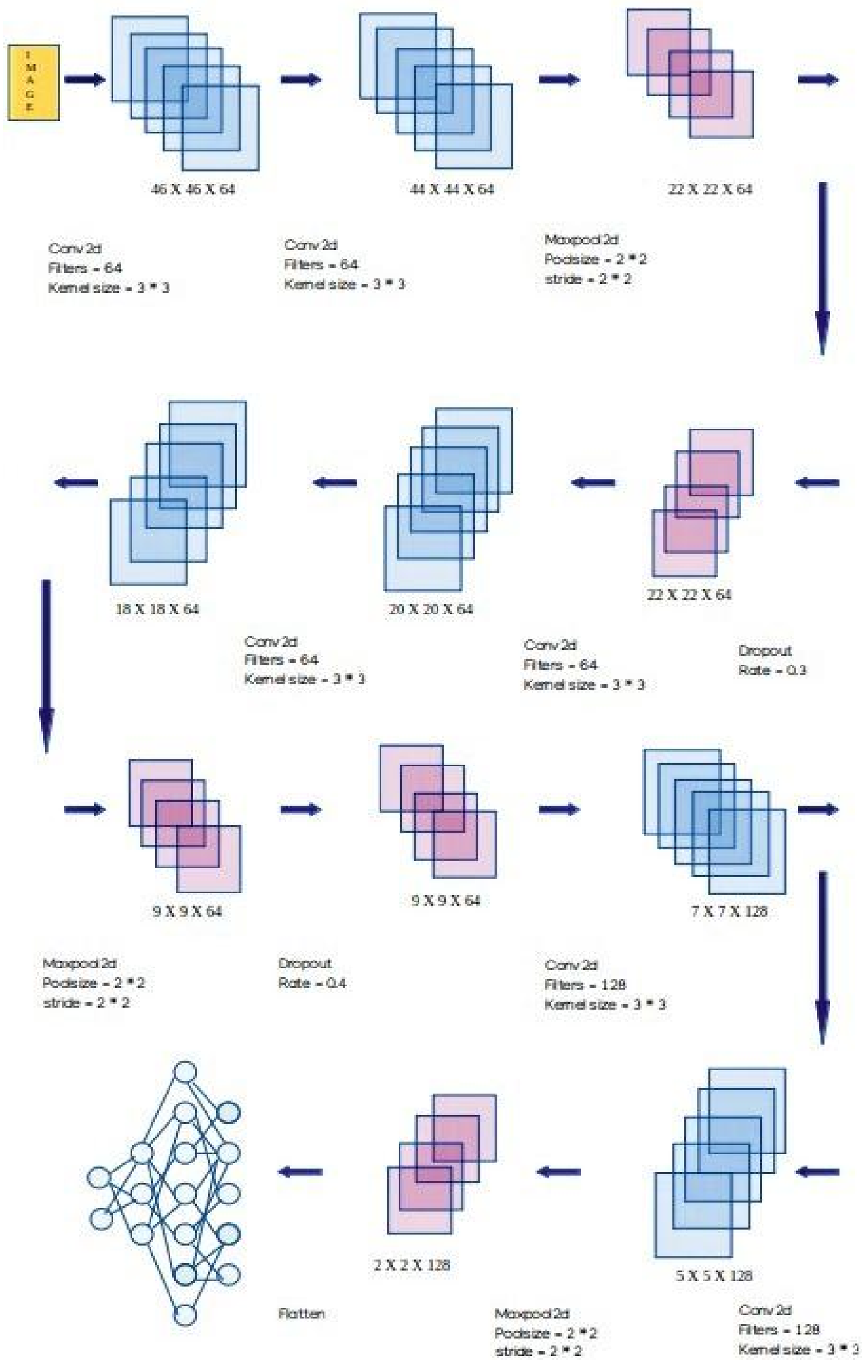

Figure 13:Block diagram of CNN layers

A ConvNet is able to successfully capture the Spatial and Temporal dependencies in an image through the application of relevant filters. So that the architecture performs a better fitting to the image dataset due to the reduction in the number of parameters involved and reusability of weights. In other words, the network can be trained to understand the sophistication of the image better and is represented through Figure 14 and Figure 15.

7.3. SUMMARY OF THE MODEL

```
Model: "sequential_6"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_36 (Conv2D)           (None, 46, 46, 64)        640
_________________________________________________________________
conv2d_37 (Conv2D)           (None, 44, 44, 64)        36928
_________________________________________________________________
max_pooling2d_18 (MaxPooling (None, 22, 22, 64)        0
_________________________________________________________________
dropout_24 (Dropout)         (None, 22, 22, 64)        0
_________________________________________________________________
conv2d_38 (Conv2D)           (None, 20, 20, 64)        36928
_________________________________________________________________
conv2d_39 (Conv2D)           (None, 18, 18, 64)        36928
_________________________________________________________________
max_pooling2d_19 (MaxPooling (None, 9, 9, 64)          0
_________________________________________________________________
dropout_25 (Dropout)         (None, 9, 9, 64)          0
_________________________________________________________________
conv2d_40 (Conv2D)           (None, 7, 7, 128)         73856
_________________________________________________________________
conv2d_41 (Conv2D)           (None, 5, 5, 128)         147584
_________________________________________________________________
max_pooling2d_20 (MaxPooling (None, 2, 2, 128)         0
_________________________________________________________________
flatten_6 (Flatten)          (None, 512)               0
_________________________________________________________________
dense_18 (Dense)             (None, 512)               262656
_________________________________________________________________
dropout_26 (Dropout)         (None, 512)               0
_________________________________________________________________
dense_19 (Dense)             (None, 512)               262656
_________________________________________________________________
dropout_27 (Dropout)         (None, 512)               0
_________________________________________________________________
dense_20 (Dense)             (None, 7)                 3591
=================================================================
Total params: 861,767
Trainable params: 861,767
Non-trainable params: 0
_________________________________________________________________
```

Figure 14:Model summary

```python
def Design_CNN():

    # The number of epochs is a hyperparameter
    # that defines the number times that the learning algorithm will work through the entire training dataset
    features = 64
    Batch_size = 64
    Label = 5
    epoch = 100
    global  x_train, y_train

    model = Sequential()
```

Figure 15:Parameters for designing CNN

EPOCH

An epoch means training the neural network with all the training data for one cycle. In an epoch, we use all of the data exactly once. A forward pass and a backward pass together are counted as one pass. Figure 16 helps in visualizing an Epoch.

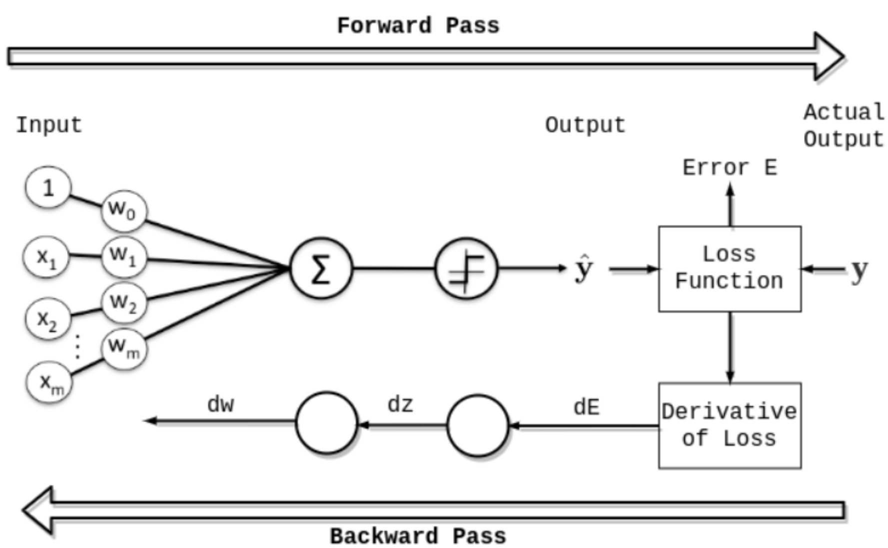

Figure 16:Visualization of an Epoch

BATCH SIZE

The batch size is a hyperparameter that defines the number of samples to work through before updating the internal model parameters.

FEATURES

The feature maps of a CNN capture the result of applying the filters to an input image. i.e at each layer, the feature map is the output of that layer.

LABEL

They are the emotions that we have worked with for Facial Expressions for Happy, Angry, Sad, Surprise and Neutral. The layers of CNN in this model are as shown in Figure:17.

```python
def Design_CNN():

    # The number of epochs is a hyperparameter
    # that defines the number times that the learning algorithm will work through the entire training dataset
    features = 64
    Batch_size = 64
    Label = 7
    epoch = 100
    global  x_train, y_train
    model = Sequential()

    ## Layer 1
    # adding layers
    # Conv2d is used as the image are in 2d format
    # here we are trying extract input
    # Relu is a rectifier

    # Search Kernal size
    model.add(Conv2D(features,kernel_size=(3,3),activation='relu',input_shape=(x_train.shape[1:])))
    model.add(Conv2D(features,kernel_size=(3, 3),activation='relu'))

    # adding a max pooling 2D layer
    # It mainly helps to control over fitting
    # can use average pooling layer also
    model.add( MaxPool2D(pool_size=(2,2),strides=(2,2)) )

    # adding a drop out layer
    model.add(Dropout(0.3))

    ## 2ND layer
    model.add(Conv2D(features, kernel_size=(3, 3), activation='relu'))
    model.add(Conv2D(features, kernel_size=(3, 3), activation='relu'))
    model.add(MaxPool2D(pool_size=(2, 2), strides=(2, 2)))
    model.add(Dropout(0.4))

    ## 3RD Layer
    model.add(Conv2D(2*features, kernel_size=(3, 3), activation='relu'))
    model.add(Conv2D(2*features, kernel_size=(3, 3), activation='relu'))
    model.add(MaxPool2D(pool_size=(2, 2), strides=(2, 2)))

    model.add( Flatten() )

    # adding dense layers
    model.add(Dense(2**3 * features, activation='relu'))
    model.add(Dropout(0.3))
    model.add(Dense(2 ** 3 * features, activation='relu'))
    model.add(Dropout(0.2))

    # Adding the final layers
    model.add(Dense(Label,activation='softmax')) # Activation is softmax as we want to bind in the 7 labels of Options

    model.compile(loss=categorical_crossentropy,optimizer=Adam(),metrics=['accuracy'])
    model.fit(x_train,y_train,batch_size=Batch_size,epochs=epoch,verbose=1,validation_data=(x_test,y_test), shuffle=True )
```

Figure 17: Typical layers of CNN in this model

7.4 MODEL TRAINING AND VALIDATION

Number of Epochs: 100
Batch Size: 64

The model while getting Trained, broken down in each Epoch, appears as follows:

Epoch 95/100
328/328 [==============================] - 7s 22ms/step - loss: 0.2434 - accuracy: 0.9122 - val_loss: 1.3486 - val_accuracy: 0.6618
Epoch 96/100
328/328 [==============================] - 7s 20ms/step - loss: 0.2590 - accuracy: 0.9095 - val_loss: 1.3581 - val_accuracy: 0.6608
Epoch 97/100
328/328 [==============================] - 6s 20ms/step - loss: 0.2718 -accuracy: 0.9071 - val_loss: 1.4403 - val_accuracy: 0.6518
Epoch 98/100
328/328 [==============================] - 6s 20ms/step - loss: 0.2584 - accuracy: 0.9081 - val_loss: 1.2969 - val_accuracy: 0.6528
Epoch 99/100
328/328 [==============================] - 7s 20ms/step - loss: 0.2556 -accuracy: 0.9121 - val_loss: 1.2792 - val_accuracy: 0.6507
Epoch 100/100
328/328 [==============================] - 7s 22ms/step - loss: 0.2638 - accuracy: 0.9065 - val_loss: 1.4305 - val_accuracy: 0.6575

GRAPHICAL REPRESENTATION

The graphical representation of the above output obtained from the model can be explained in two parts, Accuracy and Loss respectively, as shown in Figures 18 and 19 respectively.
Train Accuracy increases exponentially and the Validation Accuracy increases gradually but becomes saturated at around 68%. Thus we can say that the number of epochs can be reduced as after 50 epochs there is not much change in the Validation Accuracy.

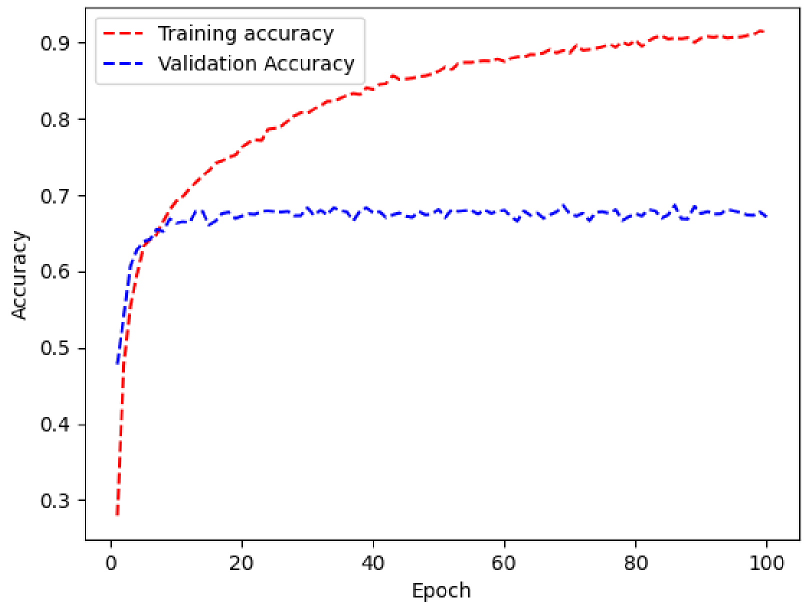

Figure 18:Accuracy plot

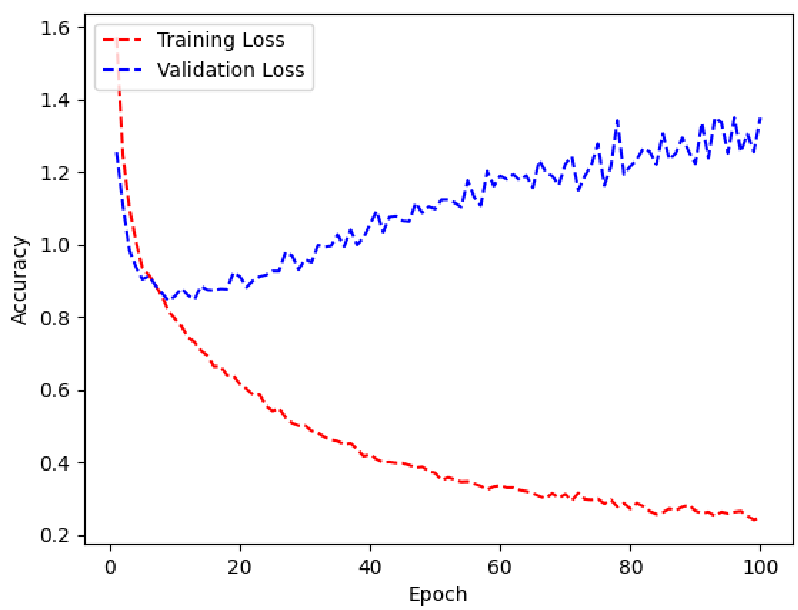

Figure 19:Loss Plot

Here we can see the model is comparable on both Training and Validation datasets (labelled). If these parallel plots start departing consistently, it is a sign to stop training at an earlier epoch.

7.5. TESTING

7.5.1. TEST ON SAVED IMAGES

When we feed a few images into the model, it isolates and marks the faces present in those images and then frames them in a rectangle. It then generates predictions for their emotion, the status of which is printed on the top-left corner of the images. Some predicted examples are as shown in Figure 20.

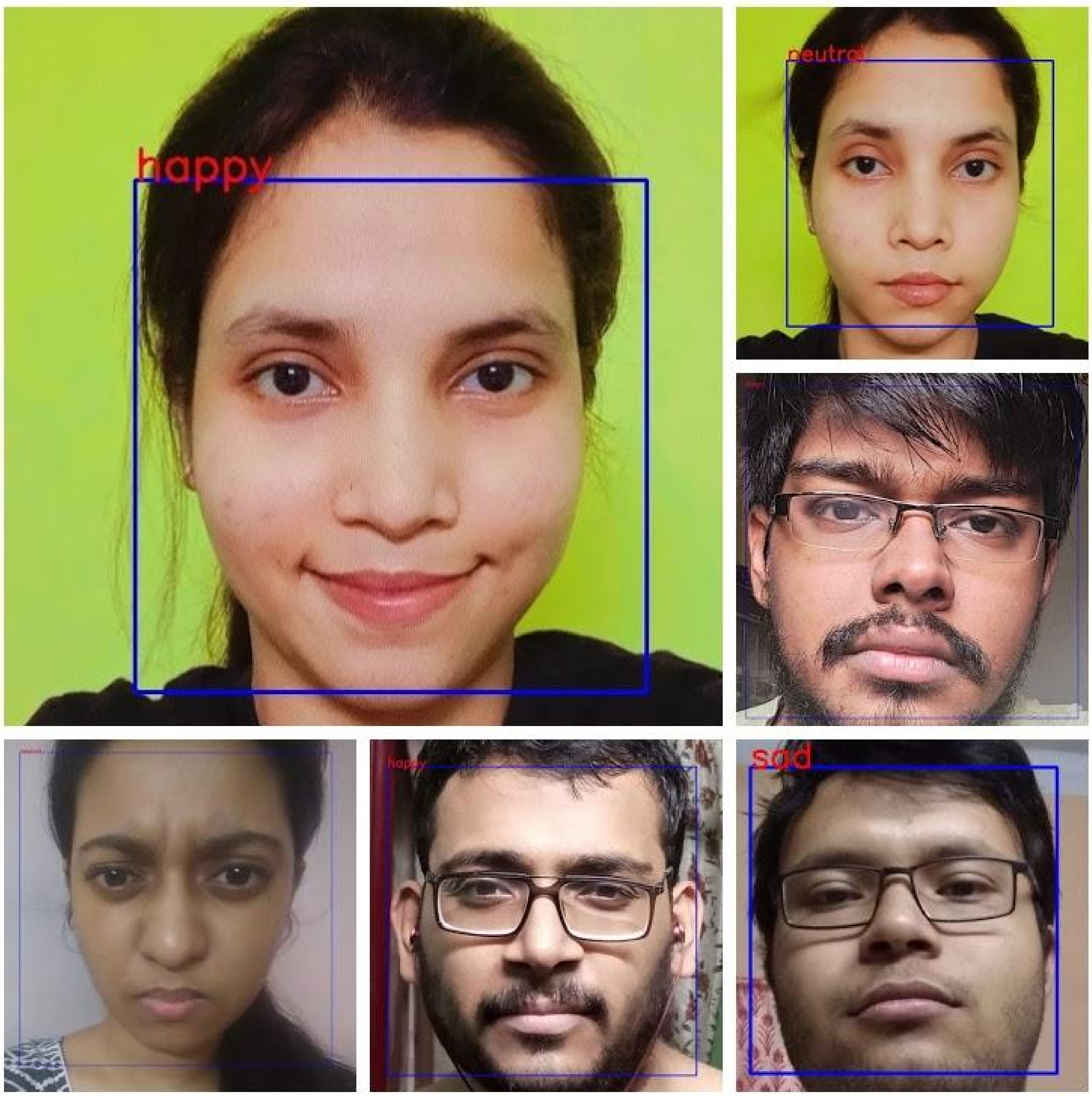

Figure 20: Compilation of some random Images fed to the model

7.5.2. TEST ON LIVE VIDEO STREAM

A live video will be captured by accessing the front camera of the device. It will start capturing frames from that stream and then feed the frames to the model. To start the video stream and capture images from it, we have used the code mentioned below provided by Jack Wotherspoon. Figure 21 nad 22 show how Image frame collection and emotion prediction, respectively.

```
[ ] # start streaming video from webcam
    video_stream()

    # label for video
    label_html = 'Capturing...'

    #face_haar_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
    # initialze bounding box to empty
    bbox = ''
    count = 0
    while True:
        js_reply = video_frame(label_html, bbox)
        if not js_reply:
            break

        # convert JS response to OpenCV Image
        img = js_to_image(js_reply["img"])

        # create transparent overlay for bounding box
        bbox_array = np.zeros([480,640,4], dtype=np.uint8)


        # grayscale image for face detection
        gray_img = cv2.cvtColor(img, cv2.COLOR_RGB2GRAY)
```

Figure 21 : Image frame collection from continuous video stream

This function also converts the JavaScript response to OpenCV image and overlays the bounding box using bbox_array. Then it converts the image to grayscale and checks for the faces detected.

```python
for (x,y,w,h) in faces_detected:
    bbox_array = cv2.rectangle(bbox_array,(x,y),(x+w,y+h),(255,0,0),2)
    cropped_gray=gray_img[y:y+w,x:x+h]  # cropping the image
    cropped_gray=cv2.resize(cropped_gray,(48,48))
    pixels = image.img_to_array(cropped_gray)
    pixels = np.expand_dims(pixels, axis = 0)
    pixels /= 255

    predictions = model.predict(pixels)
    #print( predictions )
    #find max indexed array
    max_index = np.argmax(predictions[0])
    plot.append(max_index)

    emotions = ('angry', 'happy', 'sad', 'surprise', 'neutral')
    predicted_emotion = emotions[max_index]

    cv2.putText(bbox_array, predicted_emotion, (int(x), int(y)), cv2.FONT_HERSHEY_SIMPLEX, 1, (0,0,255), 2)

bbox_array[:,:,3] = (bbox_array.max(axis = 2) > 0 ).astype(int) * 255
# convert overlay of bbox into bytes
bbox_bytes = bbox_to_bytes(bbox_array)
# update bbox so next frame gets new overlay
bbox = bbox_bytes
```

Figure 22 : Code for Emotion prediction

After face detection and converting the coloured image into a gray-scale image we traverse through the list and rescale those resolutions to the 48x48 using cv2.resize() function. After that it feeds those pexels to the predict function for emotion prediction.

According to the emotion tags mentioned in the list it returns those predicted indices and puts that corresponding text to the top left corner of the image.

Figure 23 shows the prediction of emotions on two consecutive image frames extracted from a live video stream.

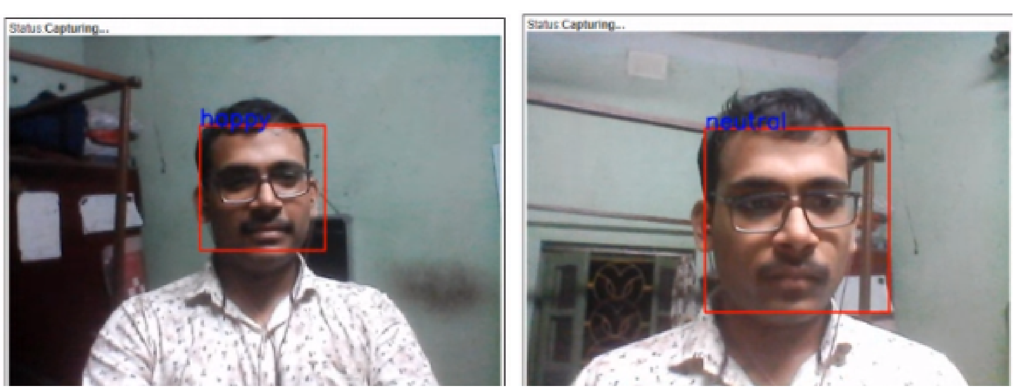

Figure 23: Runtime prediction

8. PERFORMANCE MEASUREMENT

8.1. CONFUSION MATRIX

A Confusion matrix is an N×N matrix used to evaluate the performance of a Multiclass Classification model, where N is the number of target classes. The matrix compares the actual target values with those predicted by the Machine Learning model. This provides a holistic view of how well the Classification model is performing and the kind of errors generated, it also offers us guidance to choose the correct flow [30].

For a binary classification problem, we would have a 2×2 matrix with 4 values as shown in Figure 24.

		Predicted Class	
		POSITIVE	NEGATIVE
Actual Class	POSITIVE	True Positive (TP)	False Negative (FN) Type II Error
	NEGATIVE	False Positive (FP) Type I Error	True Negative (TN)

Figure 24: Diagram of Confusion Matrix for General Binary Classification Problem

8.1.1. CLASSIFICATION OF PREDICTIONS

The Confusion Matrix consists of 4 types of Predictions with respect to the actual Emotions of Images collected from the Internet, they as follows:

True Positive (TP)

- The predicted emotion matches the actual emotion
- The actual emotion was “Anger” and the model predicted “Anger”

True Negative (TN)

- The predicted value matches the actual value
- The actual value was not Anger and the model predicted an emotion other than Anger

False Positive (FP) – Type 1 error

- An emotion was falsely predicted
- The actual value was not Anger but the model predicted Anger

False Negative (FN) – Type 2 error

- An emotion was falsely predicted
- The actual value was Anger but the model predicted an emotion other than Anger

8.2. ACCURACY CHECKPOINTS

Our dataset is an example of an **Imbalanced Dataset**, since the number of data points for Angry, Happy, Sad, Surprise and Neutral expressions are unequal. Any Accuracy Calculation would be an inefficient metric for the model owing to its multiclass nature. There must be a measure of how many positive cases the model can predict correctly to give the most accurate results. A check on the number of True Positive cases out of the correctly predicted cases, will prove the reliability of the model. This shines light on the importance of the dual concept of **Precision** and **Recall.**

8.2.1. PRECISION

It indicates how many of the correctly predicted cases actually turned out to be positive, and is a useful metric in cases where False Positive is a higher concern than False Negatives.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \dots\dots (1)$$

8.2.2. RECALL

It indicates how many of the actual positive cases the model could predict correctly and is a useful metric in cases where False Negative overshadows False Positive.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \quad \dots\dots (2)$$

In practice, when we try to increase the precision of our model, the recall goes down, and vice-versa. The F1-score captures both the trends in a single value:

8.2.3. FBETA-MEASURE

It is an abstraction of the F-measure where the balance of precision and recall in the calculation of their harmonic mean is controlled by a coefficient called “Beta”.

$$\text{Fbeta} - \text{Measure} = \frac{(1 + \text{beta}^2) \times \text{Precision} \times \text{Recall}}{\text{beta}^2 \times \text{Precision} + \text{Recall}} \quad \ldots\ldots (3)$$

The choice of the beta parameter will be used in the name of the F Beta-measure. Plugging the values of beta as 1 and 2 respectively, F1 and F2 Measures are derived.

F1-Measure: It is a Harmonic Mean of Precision and Recall, thus giving a combined idea about both of the metrics. It is maximum when Precision is equal to Recall.

$$\text{F1} - \text{Measure} = \frac{2}{\frac{1}{\text{Recall}} + \frac{1}{Precision}} \quad \ldots\ldots (4)$$

F2-Measure: It is an example of the F Beta-measure, and has the effect of lowering the importance of Precision and increasing the importance of Recall. If maximizing Precision minimizes false positives, and maximizing Recall minimizes false negatives, then the F2-measure puts more attention on minimizing false negatives than minimizing false positives. The F2-measure is calculated as follows:

$$\text{F2} - \text{Measure} = \frac{(1 + 2^2) \times \text{Precision} \times \text{Recall}}{2^2 \times \text{Precision} + \text{Recall}} \quad \ldots\ldots (5)$$

9. RESULT AND DISCUSSION

On a classification dataset with 74 data points consisting of random Facial Expression Images obtained from the Internet, a Classifier was applied on it which generated the Confusion Matrix for the Model as shown in Figure 25.

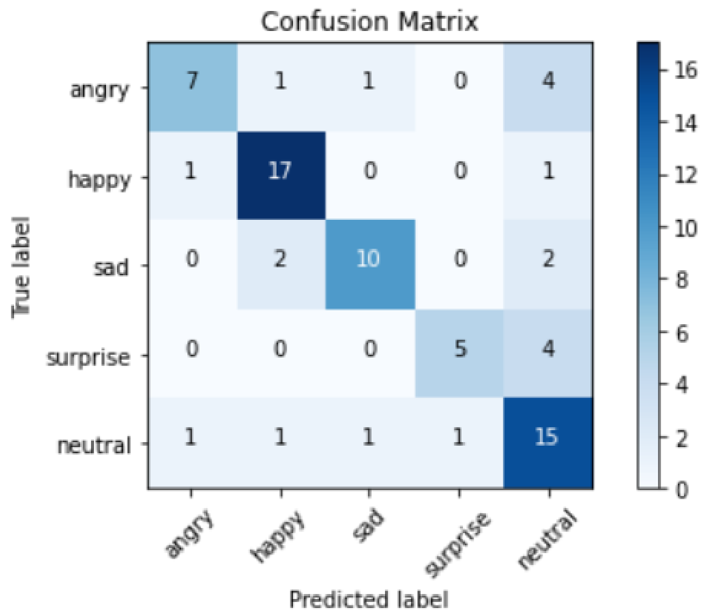

Figure 25: Confusion Matrix generated for this Model.

9.1. DATA OBTAINED FROM CONFUSION MATRIX

Sample Space : 74 Predictions
Principal Diagonal : 54 Predictions

The Principal Diagonal of the Confusion Matrix consists of all such cases where the Predicted Emotion matches the Actual Emotion, and hence it is safe to say that the Principal Diagonal of the Confusion Matrix consists of True Positive cases only. The percentage correctness of each such Prediction is evident from the Normalized form of the above Confusion Matrix, illustrated in Figure 26.

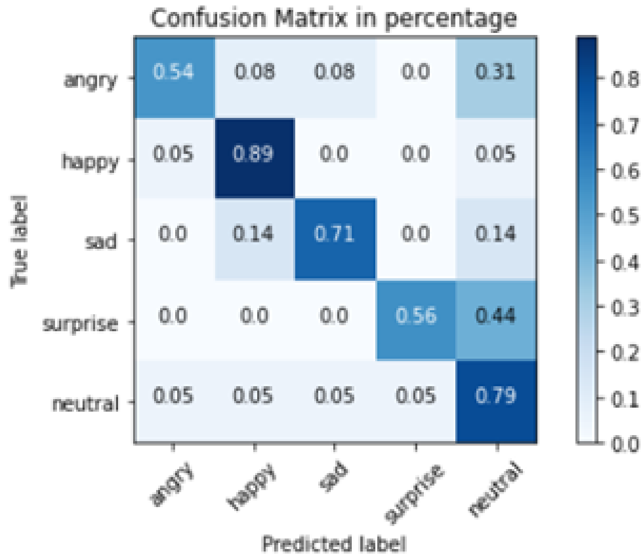

Figure 26: Normalized Confusion Matrix

Calculating the values of Precision, Recall and F1-Score from the data obtained from the Confusion Matrix:

```
import pandas as pd

# Calculating of precision , recall and f1 score for each emotion
TP = lambda cm , i : cm[i][i] # returning only for the ith emotion
FP = lambda cm , i : sum([ cm[j][i] if j!=i else 0 for j in range(len(cm)) ])
FN = lambda cm , i : sum([ cm[i][j] if j!=i else 0 for j in range(len(cm)) ])
f1 = lambda p,r : 2*(p*r)/(p+r)
TotalSample = lambda cm : sum( [ cm[i][j] for i in range(len(cm))  for j in range(len(cm)) ] )

data = [ ]
for i in range(len(plot_labels)) :
  temp =[]
  # traversing through each emotion on index
  temp.append(plot_labels[i])

  precision = TP(c_matrix , i ) / ( TP(c_matrix , i ) + FP(c_matrix , i ) )
  temp.append(round(precision,2))

  recall = TP(c_matrix , i ) / ( TP(c_matrix , i ) + FN(c_matrix , i ) )
  temp.append(round(recall,2))

  temp.append(round( f1(precision,recall) , 2 ))
  data.append(temp)

# accuracy
accuracy = sum( [ TP(c_matrix,i) for i in range(len(plot_labels)) ] ) / TotalSample(c_matrix)
print('Accuracy : ' , round(accuracy,2) )

df = pd.DataFrame( data , columns = [ 'Emotions' , 'Precision' , 'Recall' , 'F1 Score'] )
df
```

Figure 27: Code to generate precision, Recall and F1- score for individual emotions

The predictions obtained for images corresponding to each emotion are recorded and the Precision and Recall values from their respective Confusion Matrices are noted and used in calculating the F1-Score of the model. The results are shown in Figure 28.

Accuracy : 0.73

	Emotions	Precision	Recall	F1 Score
0	angry	0.78	0.54	0.64
1	happy	0.81	0.89	0.85
2	sad	0.83	0.71	0.77
3	surprise	0.83	0.56	0.67
4	neutral	0.58	0.79	0.67

Figure 28: Performance of each emotion when fed to the model

The interpretability of the F1-Score is poor since it does not clarify whether the classifier is maximizing Precision or Recall, so, it is used in combination with other evaluation metrics which gives a complete picture of the result. The values of Precision and Recall for this Model are exactly same since the number of False Positive Predictions is exactly same as that of False Negative Predictions, thus the F2-measure is exactly same as F1-Measure, since both the weightage of Precision and Recall are equal. The Accuracy of the model turns out to be approximately 73%.

10. TEST PERFORMED ON REAL TIME DATA

The idea is to extract multiple frames captured from a real time video stream of the user's webcam after taking necessary permissions, and feeding the frames as input to the model to derive Emotion Predictions.

10.1. GRAPH OF PREDICTIONS

A series of images were captured from a webcam's real time video stream and fed to the model as input to get Predictions. A plot of all results predicted against multiple "Happy" Expressions is shown in Figure 29.

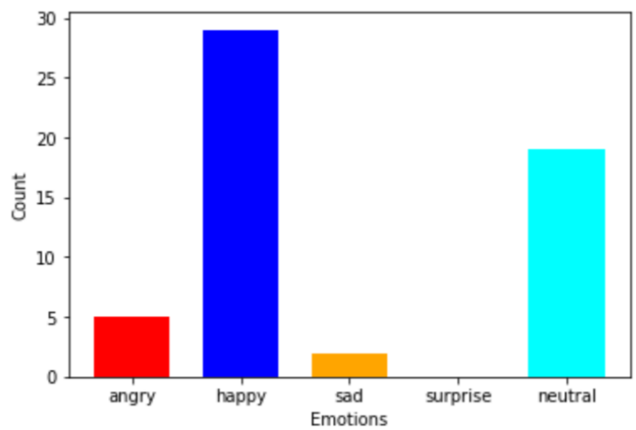

Figure 29: Count of all Predictions for a test with only "Happy" images

The count of correct emotions detected by the model for a consistently "Happy" expression from the user in front of the webcam, with 29 out of the 56 recorded data points predicted accurately as shown in the above graph, when the model is tested with frames extracted from live or real time webcam video streaming.

11. FUTURE SCOPE

This project is aimed at detecting the prototype facial expressions users trigger upon coming across a product within a website that excites them, either positively or negatively, but enough to cause a change in their facial muscles as a direct reaction of fluctuations in their emotion upon seeing the product.

The study of their expressions lead the model to form informed decisions as to what other products might pique their interest, and which category of products to be avoided. The model can be improved to detect the time duration for which a positive or negative expression sustains to calculate the likelihood of them buying the product and the genre or category of their choice.

Tracking the changes in their expressions are indicative of their emotions and hence may be stored as short-spanned review for the product base, which when grossly estimated might lead to changes in the pricing of the product. The main challenge is keeping the customers unbiased- since if they know their expressions might affect the cost of the product later on, they might feel tempted to bend their actual expression towards the direction in which they want the price to go in.

We found it difficult to draw conclusions from some of our submission results. This might be caused by the large number of ambiguous cases that exist in this domain. We found that a fairly large number of training videos could be argued to show a mixture of two or more basic emotions (such as a mixture of surprise with fear or happiness). This suggests that exploring the use of more than a single label for emotion recognition might be a useful direction for future research. This will push researchers in the future to build larger databases and create powerful deep learning architectures to recognize all basic and secondary emotions.

REFERENCES

1. Byron K., Terranova S., Nowicki S. Nonverbal Emotion Recognition and Salespersons : Linking Ability to Perceived and Actual Success. J. Appl. Soc. Psychol. 2007;37:2600–2619. doi: .1559-1816.2007.00272.x

2. Paweł Tarnowski, Marcin Kołodziej,Andrzej Majkowski and Remigiusz J.Rak .“Emotion recognition using facial expressions”. International Conference on Computational Science(ICCS), 12-14 June, 2017.

3. Y. Li, Y. Jiang, D. Tian, L. Hu, H. Lu and Z. Yuan, "AI-Enabled Emotion Communication," in IEEE Network, vol. 33, no. 6, pp. 15-21, Nov.-Dec. 2019, doi: 10.1109/MNET.001.1900070.

4. Mayya V, Pai R M, Pai M M M. Automatic Facial Expression Recognition Using CNN. Procedia Computer Science, 2016, 93:453-461.

5. S. Lawrence, C. L. Giles, Ah Chung Tsoi and A. D. Back, "Face recognition: a convolutional neural-network approach," in IEEE Transactions on Neural Networks, vol. 8, no. 1, pp. 98-113, Jan. 1997, doi: 10.1109/72.554195.

6. Ahmed J. Khalil, Alaa M. Barhoom, Bassem S. Abu-Nasser, Musleh M. Musleh & Samy S. Abu-Naser International Journal of Academic Pedagogical Research (IJAPR) 3 (9):1-7 (2019).

7. Ekman P, Friesen W V. Facial Action Coding System: A Technique for the Measurement of Facial Movement. Palo Alto, CA: Consulting Psychologists Press, 1978.

8. D. Dagar, A. Hudait, H. K. Tripathy and M. N. Das, "Automatic emotion detection model from facial expression," 2016 International Conference on Advanced Communication Control and Computing Technologies (ICACCCT), 2016, pp. 77-85, doi: 10.1109/ICACCCT.2016.7831605.

9. Zeithaml VA, Parasuraman A, Malhotra A. Service Quality Delivery through Web Sites: A Critical Review of Extant Knowledge. Journal of the Academy of Marketing Science. 2002;30(4):362-375. doi:10.1177/009207002236911.

10. Agrawal, A., Mittal, N. Using CNN for facial expression recognition: a study of the effects of kernel size and number of filters on accuracy. Vis Comput 36, 405–412 (2020).

11. D. Anurag,S. Ashim, ‘A Comparative Study on different approaches of Real Time Human Emotion Recognition based on Facial Expression Detection’, International Conference on Advances in Computer Engineering and Applications, 978- 1-4673-6911-4/15/$31.00©2015 IEEE.

12. Maja Pantic, Ioannis Patras. ‘Dynamics of Facial Expression : Recognition of Facial Actions and their Temporal Segments from face profile Image Sequences’, IEEE Transactions on System, Man and Cybernetics.1083-4419/$20.00©2006 IEEE.

13. Ayesha Gurnani, Vandit Gajjar, Viraj Mavani, Yash Khandhedia Computer Science ArXiv 2018.

14. Durga L. Shrestha, Dimitri P. Solomatine, Machine learning approaches for estimation of prediction intervals for the model output, Neural Networks,Volume 19, Issue 2,2006.

15. D. Hammerstrom, "Working with neural networks," in IEEE Spectrum, vol. 30, no. 7, pp. 46-53, July 1993, doi: 10.1109/6.222230.

16. Christ Natalis, A. M. Husein, M. Harahap, A. Dharma and A. M. Simarmata, "Hybrid-AES-Blowfish algorithm: key exchange using neural network," 2019 International Conference of Computer Science and Information Technology (ICoSNIKOM), 2019, pp. 1-4, doi: 10.1109/ICoSNIKOM48755.2019.9111500.

17. S. Albawi, T. A. Mohammed and S. Al-Zawi, "Understanding of a convolutional neural network," 2017 International Conference on Engineering and Technology (ICET), 2017, pp. 1-6, doi: 10.1109/ICEngTechnol.2017.8308186.

18. Ketkar N. (2017) Introduction to Keras. In: Deep Learning with Python. Apress, Berkeley, CA.

19. Snider, L., Swedo, S. PANDAS: current status and directions for research. Mol Psychiatry 9, 900–907 (2004).

20. S. van der Walt, S. C. Colbert and G. Varoquaux, "The NumPy Array: A Structure for Efficient Numerical Computation," in Computing in Science & Engineering, vol. 13, no. 2, pp. 22-30, March-April 2011, doi: 10.1109/MCSE.2011.37.

21. ICFP 2016: Proceedings of the 21st ACM SIGPLAN International Conference on Functional ProgrammingSeptember 2016.

22. J. H. Jung, Y. Shin and Y. Kwon, "Extension of Convolutional Neural Network with General Image Processing Kernels," TENCON 2018 - 2018 IEEE Region 10 Conference, 2018, pp. 1436-1439, doi: 10.1109/TENCON.2018.8650542.

23. Xing Wei, Yue Zhang, Yihong Gong, Nanning Zheng; Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018, pp. 1867-1875.

24. Akhtar, N., Ragavendran, U. Interpretation of intelligence in CNN-pooling processes: a methodological survey. Neural Comput & Applic 32, 879–898 (2020).

25. Deep learning using rectified linear units (relu) AF Agarap - arXiv preprint arXiv:1803.08375, 2018.

26. I. Jindal, M. Nokleby and X. Chen, "Learning Deep Networks from Noisy Labels with Dropout Regularization," 2016 IEEE 16th International Conference on Data Mining (ICDM), 2016, pp. 967-972, doi: 10.1109/ICDM.2016.0121.

27. Kaggle (2019, March 29). Challenges in Representation Learning: Facial Expression Recognition Challenge.

28. DEEM 19: Proceedings of the 3rd International Workshop on Data Management for End-to-End Machine LearningJune 2019.

29. Jared Hamwood, David Alonso-Caneiro, Scott A. Read, Stephen J. Vincent, and Michael J. Collins, "Effect of patch size and network architecture on a convolutional neural network approach for automatic segmentation of OCT retinal layers," Biomed. Opt. Express 9, 3049-3066 (2018).

30. Townsend, J.T. Theoretical analysis of an alphabetic confusion matrix. Perception & Psychophysics 9, 40–50 (1971).