FACIAL EXPRESSION RECOGNITION USING DEEP CONVOLUTIONAL NEURAL NETWORKS

An Industry Oriented Mini Project report submitted in partial fulfilment of requirements for the award of degree of

Bachelor of Technology
In
Computer Science and Engineering

By

K. KALYAN **(Reg No: 17131A0597)**
K. SAI TEJA KUMAR **(Reg No: 17131A0598)**
K. THARUN **(Reg No: 17131A0599)**
K. LALITHA PRANAVI **(Reg No: 17131A05A1)**

GAYATRI VIDYA PARISHAD COLLEGE OF ENGINEERING (AUTONOMOUS)
Approved by AICTE, New Delhi and Affiliated to JNTU-Kakinada
Re-accredited by NAAC with "A" Grade with CGPA of 3.47/4.00
Madhurawada,Visakhapatnam-530 048

Gayatri Vidya Parishad College of Engineering (Autonomous)

CERTIFICATE

This is to certify that the project thesis entitled **"FACIAL EXPRESSION RECOGNITION USING DEEP CONVOLUTIONAL NEURAL NETWORKS "** being submitted by

K. KALYAN	**(Reg No: 17131A0597)**
K. SAI TEJA KUMAR	**(Reg No: 17131A0598)**
K. THARUN	**(Reg No: 17131A0599)**
K. LALITHA PRANAVI	**(Reg No: 17131A05A1)**

in partial fulfilment for the award of the Degree of Bachelor of Technology in Computer Science and Engineering to the Jawaharlal Nehru Technological University Kakinada, Kakinada is a record of bonafied work carried out under my guidance and supervision.

The results embodied in this project thesis have not been submitted to any other University or Institute for the award of any Degree or Diploma.

G. Vani	Dr. P. Krishna Subba Rao
Assistant Professor	Head of the Department
Department of Computer Science	Department of Computer Science

DECLARATION

We hereby declare that this industry oriented main project entitled **"FACIAL EXPRESSION RECOGNITION"** is a bonafide work done by us and submitted to **Department of Computer Science and Engineering, Gayatri Vidya Parishad College of Engineering (Autonomous), Visakhapatnam,** in partial fulfilment for the award of the degree of B.Tech is of our own and it is not submitted to any other university or has been published any time before.

PLACE: VISAKHAPATNAM

DATE: 21-07-2020

K.KALYAN (Reg No:17131A0597)

K.SAI TEJA KUMAR (Reg No:17131A0598)

K.THARUN (Reg No:17131A0599)

K.LALITHA PRANAVI (Reg No:17131A05A1)

ACKNOWLEDEGEMENT

We would like to take this opportunity to extend our hearty gratitude to our esteemed institute **"Gayatri Vidya Parishad College of Engineering (Autonomous)"** where we got the platform to fulfill our cherished desire.

We express our sincere thanks to **Dr. A.B. KOTESWARA RAO,** principal, Gayatri Vidya Parishad College of Engineering (Autonomous), for his support and encouragement during the course of this project.

We express our deep gratitude to Dr. P. Krishna Subba Rao**,** Head of the Department of Computer Science and Engineering ,Gayatri Vidya Parishad College of Engineering (Autonomous) for his constant support and encouragement.

We are obliged to **G.VANI,** Assistant Professor, Department of Computer Science and Engineering, who has been our guide, whose valuable suggestions, guidance and comprehensive assistance helped us a lot in realizing the project.

We also thank **Dr.OMPRAKASH THEMBURNE,** Associate Professor**,** Project Coordinator, Department of Computer Science and Engineering, for guiding us throughout the project and helping us in completing the project efficiently.

Lastly, we are grateful to all our friends, for their relentless support in augmenting the value of work, our family, for being considerate and appreciative throughout.

K.KALYAN (Reg No:17131A0597)

K.SAI TEJA KUMAR (Reg No:17131A0598)

K.THARUN (Reg No:17131A0599)

K.LALITHA PRANAVI (Reg No:17131A05A1)

ABSTRACT

Facial expression recognition has been a potential research area in recent years due to the advancement of its related research area especially machine learning, computer vision, image processing, and cognitive science.

These Human facial expressions convey a lot of information visually rather than articulately. Facial expression recognition plays a crucial role in the area of human-machine interaction. Automatic facial expression recognition system has many applications including, but not limited to, human behavior understanding, detection of mental disorders, and synthetic human expressions. Recognition of facial expression by computer with high recognition rate is still a challenging task. Two popular methods utilized mostly in the literature for the automatic FER systems are based on geometry and appearance.

Kaggle facial expression dataset FER2013 with four facial expression labels as happy, sad, surprise, and neutral is used in this project.In this project we applied various deep learning methods (convolutional neural networks) to identify the some basic human emotions: happiness, sadness, surprise and neutrality.

INDEX

Contents

1.INTRODUCTION

The recognition of facial expressions is not an easy problem for machine learning methods, since people can vary significantly in the way they show their expressions. Even images of the same person in the same facial expression can vary in brightness, background and pose, and these variations are emphasized if considering different subjects (because of variations in shape, ethnicity among others). Hence, facial expression recognition is still a challenging problem in computer vision. In this work, we propose a simple solution for facial expression recognition that uses a combination of Convolutional Neural Network and specific image pre-processing steps.

1.1 OBJECTIVE

The objective of face expression recognition (FER) is identifying emotions of a human. The emotion can be captured either from face or from verbal communication. Psychological characteristics such as heartbeat and blood Pressure, speech, hand gestures, body movements, Facial expressions identify emotions of a person. Facial emotion recognition is one of the useful tasks and can be used as a base for many real-time applications. It can be used as a part of many interesting and useful applications like Monitoring security, treating patients in medical field, marketing research, eLearning etc;.

1.2 ABOUT THE PROJECT

In this project, we have developed convolutional neural networks (CNN) for a facial expression recognition task. The goal is to classify each facial image into one of the four facial emotion categories. We trained CNN models using gray-scale images from the Kaggle FER2013 dataset. We developed our models using Anaconda and used Google-Colab's graphics processing Unit (GPU) computation in order to expedite the training process.

.

1.3 PROCEDURE AND PURPOSE

This software first detects and reads human face, then the system computes various parameters. Upon detecting and registering these parameters, the system compares these parameters with default expressions for human sadness, smile and human expressions. Based on these statistics the system concludes the person's emotional state.

For real-time purposes, facial emotion recognition has a number of applications. Facial emotion recognition could be used in conjunction with other systems to provide a form of safety. For instance, ATMs could be set up such that they won't dispense money when the user is scared. In the gaming industry, emotion-aware games can be developed which could vary the difficulty of a level depending on the player's emotions. It also has uses in video game testing. At present, players usually give some form of verbal or written feedback. Using facial emotion recognition, the number of testers can be increased to accommodate people who use different languages or people who are unable to cohesively state their opinions on the game.

2.LITERATURE SURVEY

Two different approaches are used for facial expression recognition, both of which include two different methodologies, exist. Dividing the face into separate action units or keeping it as a whole for further processing appears to be the first and the primary distinction between the main approaches. In both of these approaches, two different methodologies, namely the 'Geometric based' and the 'Appearance-based' parameterizations, can be used.

Making use of the whole frontal face image and processing it in order to end up with the classifications of 4 universal facial expression prototypes: surprise, sad ness,happy and neutral; outlines the first approach. Here, it is assumed that each of the above mentioned emotions have characteristic expressions on face and that's why recognition of them is necessary and sufficient. Instead of using the face images as a whole, dividing them into some sub-sections for further processing forms up the main idea of the second approach for facial expression analysis. As expression is more related with subtle changes of some discrete features such as eyes, eyebrows and lip corners; these fine-grained changes are used for analyzing automated recognition.

3. SOFTWARE REQUIREMENT ANALYSIS

3.1 PROBLEM DEFINITION

Human emotions and intentions are expressed through facial expressions and deriving an efficient and effective feature is the fundamental component of facial expression system. Facial expressions convey non-verbal cues, which play an important role in interpersonal relations. Automatic recognition of facial expressions can be an important component of natural humanmachine interfaces; it may also be used in behavioral science and in clinical practice. The objective is to develop a reliable system for facial expression recognition using the Sequential CNN.

3.2 SRS DOCUMENT

3.2.1 FUNCTIONAL REQUIREMENTS

A functional requirement defines a function of a system or its component. A function described as a set of inputs, the behavior, and outputs.

SOFTWARE REQUIREMENTS

- Operating system: Windows
- Programming Language: Python
- Packages required:

  TensorFlow

  Keras

  Matplotlib

  livelossplot

  NumPy

HARDWARE REQUIREMENTS

- Processor: intel/ARM processor
- RAM: 8GB
- VRAM: 4GB
- Disc space: 1TB

3.2.2 NON FUNCTIONAL REQUIREMENTS

- The image input size must be minimum of 48*48 pixels.

3.3 SOFTWARE DESCRIPTION

3.3.1. JUPYTER NOTEBOOK

The Jupyter Notebook is an **interactive computing environment** that enables users to author notebook documents that include: - Live code - Interactive widgets - Plots - Narrative text - Equations - Images – Video

Components

The Jupyter Notebook combines three components:

- **The notebook web application**: An interactive web application for writing and running code interactively and authoring notebook documents.
- **Kernels**: Separate processes started by the notebook web application that runs users' code in a given language and returns output back to the notebook web application. The kernel also handles things like computations for interactive widgets, tab completion and introspection.
- **Notebook documents**: Self-contained documents that contain a representation of all content visible in the notebook web application, including inputs and outputs of the computations, narrative text, equations, images, and rich media representations of objects. Each notebook document has its own kernel.

Notebook documents contain the **inputs and outputs** of an interactive session as well as **narrative text** that accompanies the code but is not meant for execution. **Rich output** generated by running code, including HTML, images, video, and plots, is embedded in the notebook, which makes it a complete and self-contained record of a computation. When you run the notebook web application on your computer, notebook documents are just **files on your local filesystem with a ``.ipynb`` extension**. Notebooks consist of a **linear sequence of cells**. There are three basic cell types:

- **Code cells:** Input and output of live code that is run in the kernel
- **Markdown cells:** Narrative text with embedded LaTeX equations
- **Raw cells:** Unformatted text that is included, without modification, when notebooks are converted to different formats using nbconvert

3.3.2 TENSORFLOW

TensorFlow is a free and open source software library for dataflow and differential programming across a range of tasks. It is a symbolic math library and is used for machine learning applications such as neural networks. TensorFlow was developed by Google Brain team for internal Google use.

A tensor is a **vector** or **matrix** of n-dimensions that represents all types of data. All values in a tensor hold identical data type with a known (or partially known) **shape**. The shape of the data is the dimensionality of the matrix or array.

A tensor can be originated from the input data or the result of a computation. In Tensor Flow, all the operations are conducted inside a **graph**. The graph is a set of computation that takes place successively. Each operation is called an **op node** and is connected to each other.

The graph outlines the ops and connections between the nodes. However, it does not display the values. The edge of the nodes is the tensor, i.e., a way to populate the operation with data.

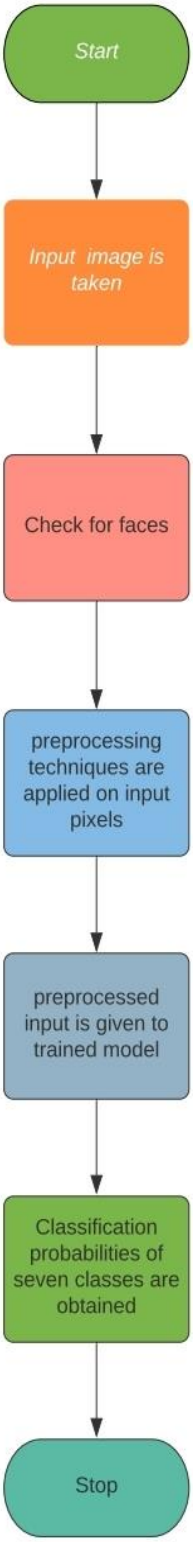
In Machine Learning, models are feed with a list of objects called **feature vectors**.

A feature vector can be of any data type. The feature vector will usually be the primary input to populate a tensor.

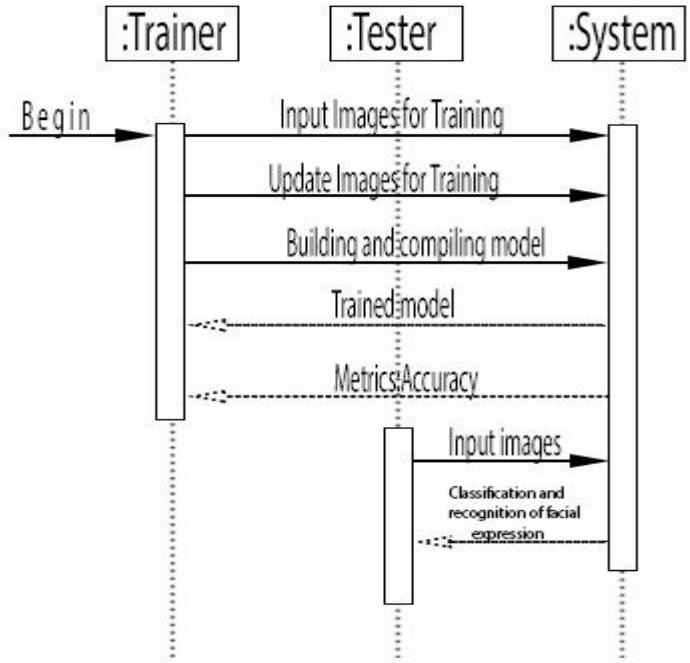
These values will flow into an op node through the tensor and the result of this operation/computation will create a new tensor which in turn will be used in a new operation. All these operations can be viewed in the graph.

4.SOFTWARE DESIGN

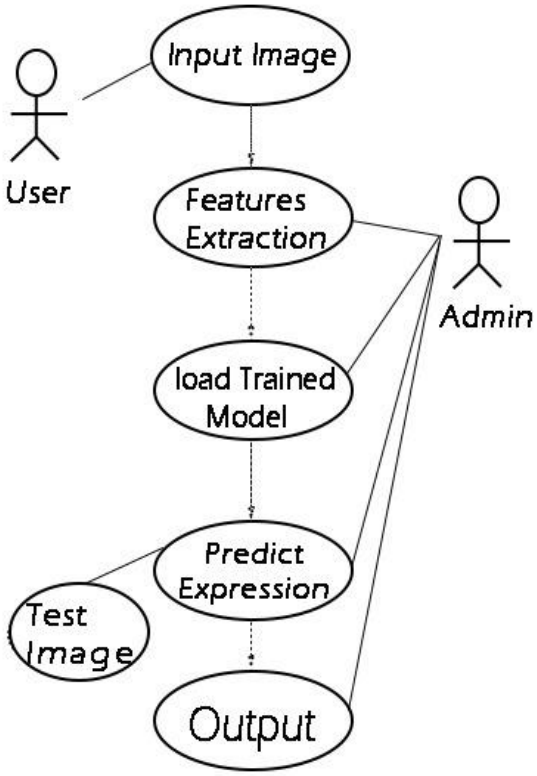
4.1 CONTROL FLOW DIAGRAM



4.2 SEQUENCE DIAGRAM



4.3 USECASE DIAGRAM



5. CODE TEMPLATE

5.1 TRAINING CODE

```
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import utils
import os
%matplotlib inline

from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.layers import Dense, Input, Dropout,Flatten, Conv2D
from tensorflow.keras.layers import BatchNormalization, Activation, MaxPooling2D
from tensorflow.keras.models import Model, Sequential
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.callbacks import ModelCheckpoint, ReduceLROnPlateau
from tensorflow.keras.utils import plot_model

from IPython.display import SVG, Image
from livelossplot import PlotLossesTensorFlowKeras
import tensorflow as tf
print("Tensorflow version:", tf.__version__)
for expression in os.listdir("train/"):
    print(str(len(os.listdir("train/" + expression))) + " " + expression + " images")
img_size = 48
batch_size = 64

datagen_train = ImageDataGenerator(horizontal_flip=True)
train_generator = datagen_train. flow_from_directory("train/",target_size=(img_size,img_size),
                                                     color_mode="grayscale",
```

```
                                        batch_size=batch_size,
                                        class_mode='categorical',
                                        shuffle=True)

datagen_validation = ImageDataGenerator(horizontal_flip=True)
validation_generator = datagen_validation.flow_from_directory("test/",
                                        target_size=(img_size,img_size),
                                        color_mode="grayscale",
                                        batch_size=batch_size,
                                        class_mode='categorical',
                                        shuffle=False)
# Initialising the CNN
model = Sequential()

# 1 - Convolution
model.add(Conv2D(64,(3,3), padding='same', input_shape=(48, 48,1)))
model.add(BatchNormalization())
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))

# 2nd Convolution layer
model.add(Conv2D(128,(5,5), padding='same'))
model.add(BatchNormalization())
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))

# 3rd Convolution layer
```

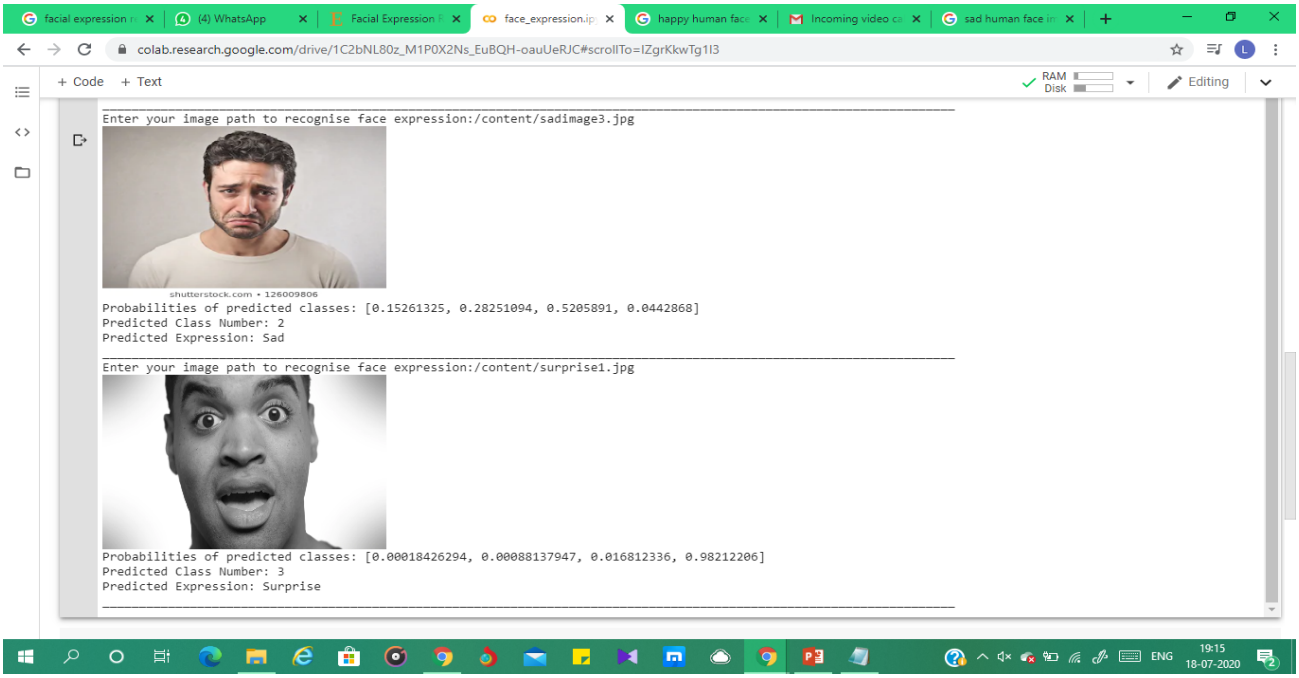
```
model.add(Conv2D(512,(3,3), padding='same'))
model.add(BatchNormalization())
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))

# 4th Convolution layer
model.add(Conv2D(512,(3,3), padding='same'))
model.add(BatchNormalization())
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))

# Flattening
model.add(Flatten())

# Fully connected layer 1st layer
model.add(Dense(256))
model.add(BatchNormalization())
model.add(Activation('relu'))
model.add(Dropout(0.25))

# Fully connected layer 2nd layer
model.add(Dense(512))
model.add(BatchNormalization())
model.add(Activation('relu'))
model.add(Dropout(0.25))

model.add(Dense(4, activation='softmax'))
opt = Adam(lr=0.0003)
```

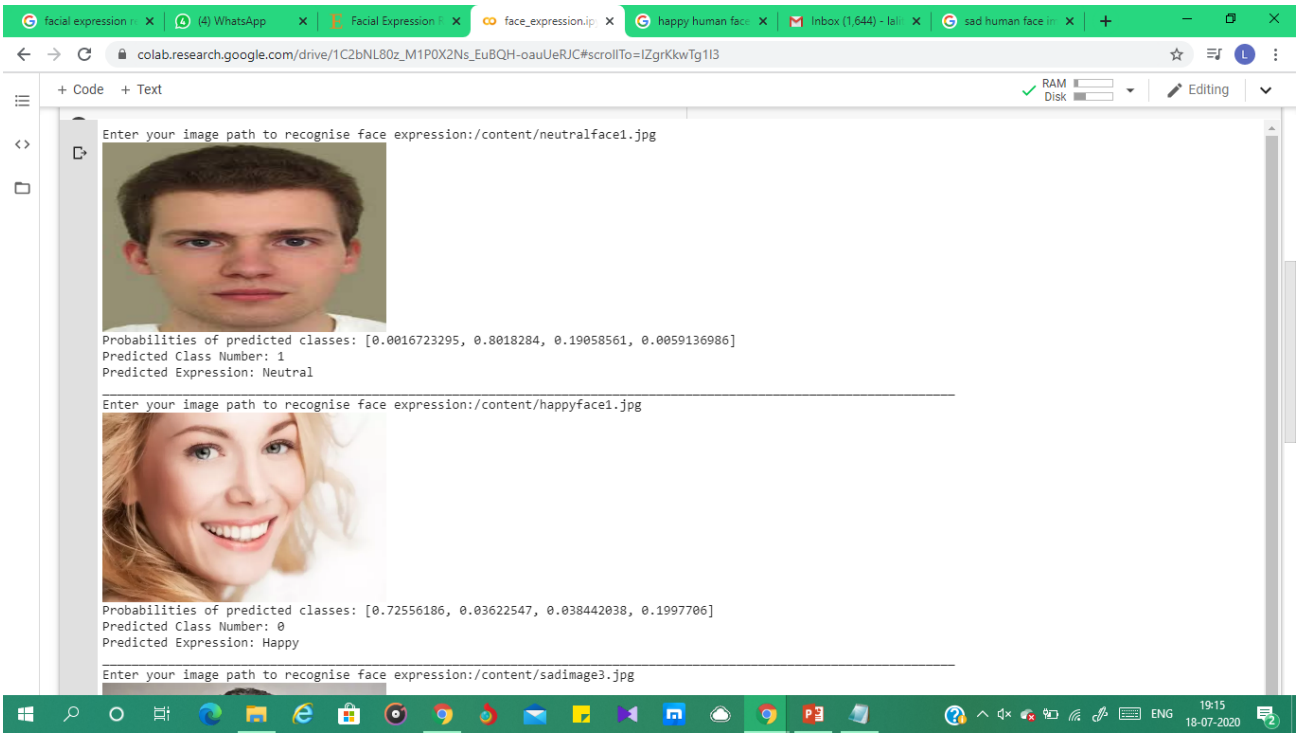
```
model.compile(optimizer=opt, loss='categorical_crossentropy', metrics=['accuracy'])
model.summary()

%%time

epochs = 40
steps_per_epoch = train_generator.n//train_generator.batch_size
validation_steps = validation_generator.n//validation_generator.batch_size

reduce_lr = ReduceLROnPlateau(monitor='val_loss', factor=0.1,
                              patience=2, min_lr=0.00001, mode='auto')
checkpoint = ModelCheckpoint("model_weights.h5", monitor='val_accuracy',
                             save_weights_only=True, mode='max', verbose=1)
callbacks = [PlotLossesTensorFlowKeras(), checkpoint, reduce_lr]

history = model.fit(
    x=train_generator,
    steps_per_epoch=steps_per_epoch,
    epochs=epochs,
    validation_data = validation_generator,
    validation_steps = validation_steps,
    callbacks=callbacks
)
model.save("trainedmodel_epoch40.h5")
```

5.2 TESTING CODE

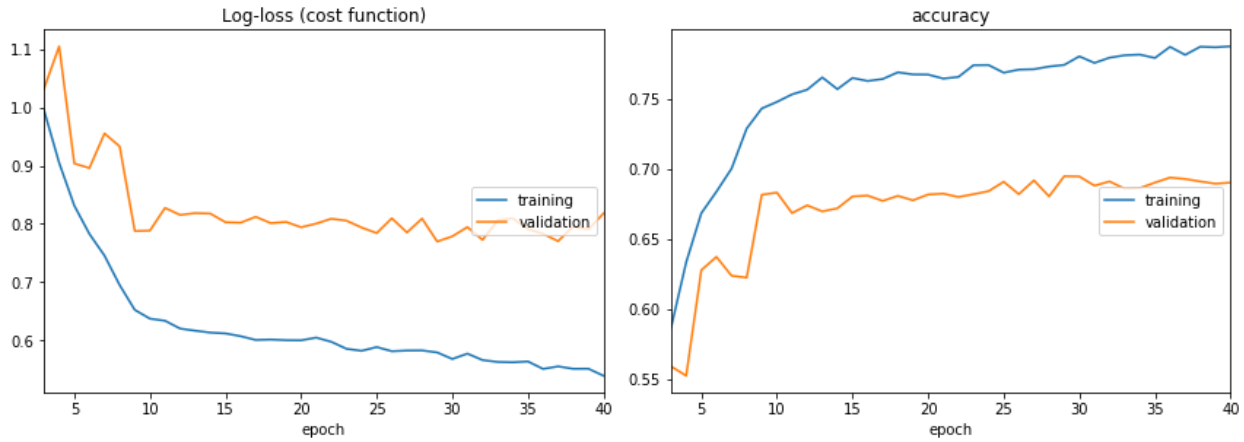
```
import tensorflow as tf
import keras
import numpy as np

from google.colab.patches import cv2_imshow
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing.image import img_to_array


expression = ["Happy", "Neutral", "Sad", "Surprise"]
image = cv2.imread('sadimg2.jpg')
cv2_imshow(image)
image = cv2.resize(image, (48,48))
model = load_model("trainedmodel.h5")
image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
imageArray = img_to_array(image)
imageArray = np.expand_dims(imageArray, axis = 0)
print(imageArray.shape)
y = model.predict(x = imageArray)
l = y[0]
# print(l)
l=list(l)
print(l)
temp = l.index(max(l))
print(temp)
print(expression[temp])
```

6.TESTING AND OUTPUT SCREENS





7. GRAPHS



8.CONCLUSION

In this project, we implemented sequential convolution neural network to classify four human facial expressions i.e. happy, sad, surprise, and neutral. The behavioural aspect of this system relates the attitude behind different expressions as property base. Our classifier achieved accuracy of 78 % on FER2013. We saved our custom trained model and it can make analysis on expressions of people.

9. FURTHER ENHANCEMENTS

While we achieved a successful implementation, significant improvements can be made by addressing several key issues. First, a much larger dataset should be designed to improve the model's generality. This project can be extended for real-time face scanners, videos and also for group photos to assess both people's mental health and integrity.

10. BIBLIOGRAPHY

1. **StackOverflow -** https://stackoverflow.com/

2. **Towards Data Science -** https://towardsdatascience.com/

3. **Kaggle -** https://www.kaggle.com/

4. **Deeplearning.ai -** **1.**https://www.youtube.com./channel/UCcIXc5mJsHVYTZR1maL5l9w

   2. https://www.coursera.org/learn/convolutional-neural-networks