FACIAL EMOTION DETECTION USING DEEP LEARNING

A minor project report submitted

In partial Fulfillment of the Requirements

For the award of the degree of

BACHELOR OF TECHNOLOGY

In

Electronics and Communication Engineering

By

Pranav Batra (08314802819)

Devanshu Agrawal (09614802819)

Rishu Raj (21314802819)

Under the Supervision of

Ms. Kanika Agarwal, Assistant Professor

MAHARAJA AGRASEN INSTITUTE OF TECHNOLOGY

SECTOR 22, ROHINI, DELHI

Affiliated to

GGSIP University, Dwarka, Delhi

DEC,2022

CERTIFICATE

Certified that Pranav Batra (08314802819), Devanshu Agarwal (09614802819) and Rishu Raj(21314802819) have carried out the minor project work presented in this report entitled **"Facial Emotion Detection using Deep Learning"** for the award of **Bachelor of Technology in Electronics and Communication Engineering** from Maharaja Agrasen Institute of Technology affiliated to GGSIP University , Delhi under my supervision .The report embodies results of Original work and studies as carried out by the students himself.

Dr . Sunil Mathur	Ms. Kanika Agarwal
Professor & HOD, ECE	Assistant Professor, ECE

DATE: -

FACIAL EMOTION DETECTION USING DEEP LEARNING

ABSTRACT

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. Facial Expression Recognition usually performed in four-stages consisting of pre-processing, face detection, feature extraction, and expression classification.

In this project we applied various deep learning methods (convolutional neural networks) to identify the key seven human emotions: anger, disgust, fear, happiness, sadness, surprise and neutrality

ACKNOWLEDGEMENT

First and foremost, I thank the Almighty God for sustaining the enthusiasm with which I plunged into this endeavor. I avail this opportunity to express my profound sense of sincere and deep gratitude to all the people who are responsible for the knowledge and experience I have gained during this Project Work.

I have great pleasure in expressing my deep sense of gratitude to our Mentor **Ms. Kanika Agarwal** for her guidance. During this project I could build upon the understanding and knowledge of the subject, acquired new competencies and skills.

Last but not the least I extend my gratitude towards my parents, faculties and friends who extended their wholehearted support towards the successful completion of this Project Work.

Thank you.

Pranav Batra	Devanshu Agarwal	Rishu Raj
08314802819	09614802819	21314802819

Table Of Contents

List Of Images

List Of Tables

1.INTRODUCTION

"2018 is the year when machines learn to grasp human emotions" --Andrew Moore, the dean of computer science at Carnegie Mellon.

With the advent of modern technology our desires went high and it binds no bounds. In the present era a huge research work is going on in the field of digital image and image processing. The way of progression has been exponential and it is ever increasing. Image Processing is a vast area of research in present day world and its applications are very widespread.

Image processing is the field of signal processing where both the input and output signals are images. One of the most important application of Image processing is Facial expression recognition. Our emotion is revealed by the expressions in our face. Facial Expressions plays an important role in interpersonal communication. Facial expression is a non-verbal scientific gesture which gets expressed in our face as per our emotions. Automatic recognition of facial expression plays an important role in artificial intelligence and robotics and thus it is a need of the generation. Some application related to this include Personal identification and Access control, Videophone and Teleconferencing, Forensic application, Human-Computer Interaction, Automated Surveillance, Cosmetology and so on.

The objective of this project is to develop Automatic Facial Expression Recognition System which can take human facial images containing some expression as input and recognize and classify it into seven different expression class such as :-

I. **Neutral**
II. **Angry**
III. **Disgust**
IV. **Fear**
V. **Happy**
VI. **Sadness**
VII. **Surprise**

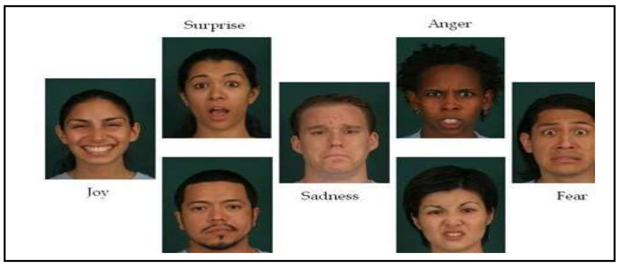


Fig.1.1.

2.MOTIVATION

Significant debate has risen in past regarding the emotions portrayed in the world famous masterpiece of Mona Lisa. British Weekly „New Scientist" has stated that she is in fact a blend of many different emotions, 83%happy, 9% disgusted, 6% fearful, 2% angry.

Fig.2.1.

We have also been motivated observing the benefits of physically handicapped people like deaf and dumb. But if any normal human being or an automated system can understand their needs by observing their facial expression then it becomes a lot easier for them to make the fellow human or automated system understand their needs

Fig.2.2.

3.PROBLEM DEFINITION

Human facial expressions can be easily classified into 7 basic emotions: happy, sad, surprise, fear, anger, disgust, and neutral. Our facial emotions are expressed through activation of specific sets of facial muscles. These sometimes subtle, yet complex, signals in an expression often contain an abundant amount of information about our state of mind. Through facial emotion recognition, we are able to measure the effects that content and services have on the audience/users through an easy and low-cost procedure. For example, retailers may use these metrics to evaluate customer interest. Healthcare providers can provide better service by using additional information about patients' emotional state during treatment. Entertainment producers can monitor audience engagement in events to consistently create desired content.

Humans are well-trained in reading the emotions of others, in fact, at just 14 months old, babies can already tell the difference between happy and sad. **But can computers do a better job than us in accessing emotional states?** To answer the question, We designed a deep learning neural network that gives machines the ability to make inferences about our emotional states. In other words, we give them eyes to see what we can see.

Problem formulation of our project: -

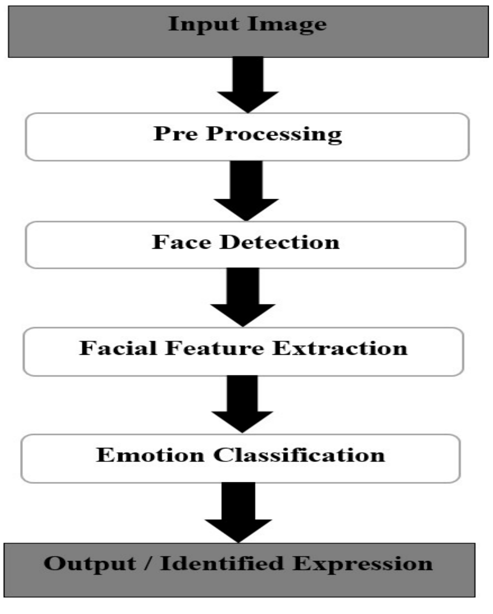


Fig.3.1.

4.LITERATURE STUDY

As per various literature surveys it is found that for implementing this project four basic steps are required to be performed.

i. Preprocessing
ii. Face registration
iii. Facial feature extraction
iv. Emotion classification

Description about all these processes are given below-

- **Pre-processing :**

Preprocessing is a common name for operations with images at the lowest level of abstraction both input and output are intensity images. Most preprocessing steps that are implemented are –

a. Reduce the noise

b. Convert The Image to Binary/Grayscale.

c. Pixel Brightness Transformation.

d. Geometric Transformation

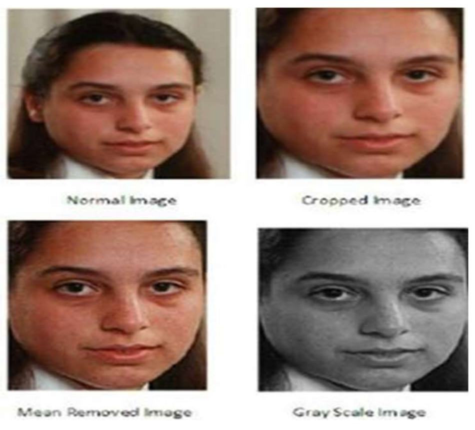


Fig.4.1.

❖ Face Registration :

Face Registration is a computer technology being used in a variety of applications that identifies human faces in digital images. In this face registration step, faces are first located in the image using some set of landmark points called "face localization" or "face detection". These detected faces are then geometrically normalized to match some template image in a process called "face registration".

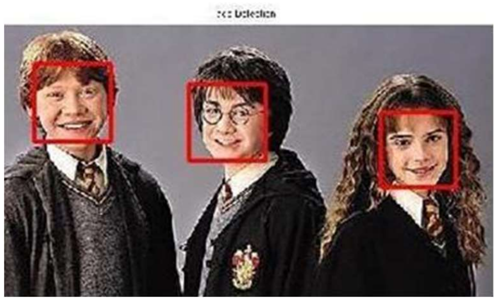


Fig.4.2.

❖ Facial Feature Extraction :

Facial Features extraction is an important step in face recognition and is defined as the process of locating specific regions, points, landmarks, or curves/contours in a given 2-D image or a 3D range image. In this feature extraction step, a numerical feature vector is generated from the resulting registered image. Common features that can be extracted are-

a. Lips

b. Eyes

c. Eyebrows

d. Nose tip

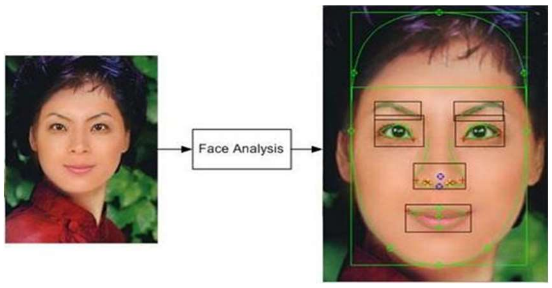


Fig.4.3.

Emotion Classification :

In the third step, of classification, the algorithm attempts to classify the given faces portraying one of the seven basic emotions.

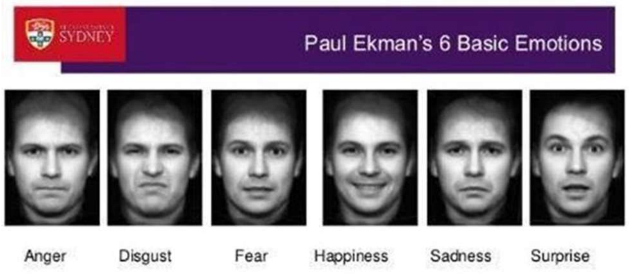


Fig.4.4.

Paul Ekman (born February 15, 1934) is an American psychologist and professor emeritus at the University of California, San Francisco who is a pioneer in the study of emotions and their relation to facial expressions. He has created an "atlas of emotions" with more than ten thousand facial expression.

Different approaches which are followed for Facial Expression Recognition:

- **Neural Network Approach :**

The neural network contained a hidden layer with neurons. The approach is based on the assumption that a neutral face image corresponding to each image is available to the system. Each neural network is trained independently with the use of on-line back propagation.

Neural Network will be discussed later.

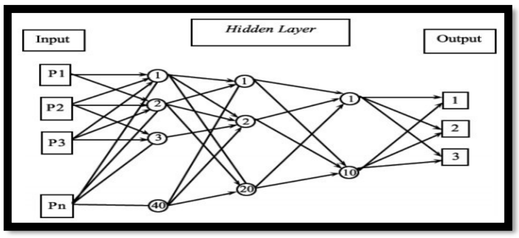


Fig.4.5.

- **Gabon Filter**

In image processing, a **Gabor filter**, named after Dennis Gabor, is a linear filter used for texture analysis, which means that it basically analyses whether there are any specific frequency content in the image in specific directions in a localized region around the point or region of analysis. Frequency and orientation representations of Gabor filters are claimed by many contemporary vision scientists to be similar to those of the human visual systemm, though there is no empirical evidence and no functional rationale to support the idea. They have been found to be particularly appropriate for texture representation and discrimination. In the spatial domain, a 2D Gabor filter is a Gaussiann kernel functionn modulated by a sinusoidal plane wave.

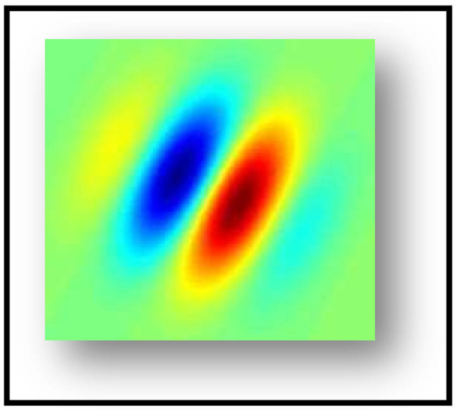


Fig.4.6.

Support Vector Machine :

In machine learning, **support vector machines** (**SVMs**, also **support vector networks**) are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary model (although methods such as Platt scaling exist to use SVM in a probabilistic classification setting). An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall.

- **Training & Testing Database:**

In machine learning, the study and construction of algorithms that can learn from and make predictions on data is a common task. Such algorithms work by making data-driven predictions or decisions, through building a mathematical model from input data.

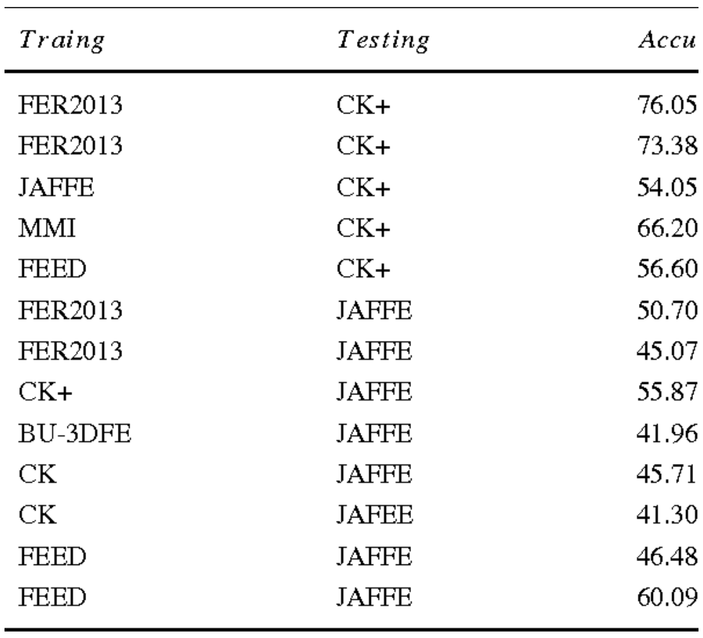
The data used to build the final model usually comes from multiple datasets. In particular, three data sets are commonly used in different stages of the creation of the model.

The model is initially fit on a **training dataset**, that is a set of examples used to fit the parameters (e.g. weights of connections between neurons in artificial neural networks) of the model. The model (e.g. a neural net or a naive Bayes classifier) is trained on the training dataset using a supervised learning method (e.g. gradient descent or stochastic gradient descent).

- **Various facial datasets available online are:**

1. Japanese Female Facial Expression (JAFFE)
2. FER
3. CMU MultiPIE
4. Lifespan
5. MMI
6. FEED
7. CK

- **Accuracy of various databases:**



Traing	*Testing*	*Accu*
FER2013	CK+	76.05
FER2013	CK+	73.38
JAFFE	CK+	54.05
MMI	CK+	66.20
FEED	CK+	56.60
FER2013	JAFFE	50.70
FER2013	JAFFE	45.07
CK+	JAFFE	55.87
BU-3DFE	JAFFE	41.96
CK	JAFFE	45.71
CK	JAFEE	41.30
FEED	JAFFE	46.48
FEED	JAFFE	60.09

Table.4.1

Accuracy of various approaches are stated as follows:

Sr. No	**Method/ Technique(s)/ (Database)**	**Result/Accur a cy**	**Conclusion**	**Future work**
1	Neural Network + Rough Contour Estimation Routine (RCER) [15] (Own Database)	92.1% recognition Rate	In this paper, they describe radial basis function network (RBFN) and a multilayer perception (MLP) network.	-
2	Principal Component Analysis [18] (FACE94)	35% less computatio n time and 100% Recognition	Useful where larger database and less computational time	They want to repeat their experiment on larger and different databases.
		83% Surpris e in CK,	Compared with the facia l	Future work is to develop

3	PCA + Eigenfaces [19] (CK, JAFFE)	83% Happiness in JAFFE, Fear was the most Confused Expression	expression recognition method based on the video sequence, the one based on the static image is more difficult due to the lack of temporal information.	a facial expression recognition system, which combines body gestures of the user with user facial expressions.
4	2D Gabor filter [22] (Random Images)	12 Gabor Filter bank used to locate Edge	Multichannel Gabor filtration scheme used for the detection of salient points and the extraction of texture features for image retrieval applications.	They work on adding global and local colour histograms and parameters connected with the shapes of objects within images.
5	Local Gabor Filter + PCA + LDA [23] (JAFFE)	Obtained 97.33% recognition rate with the help of PCA+LDA Features	They conclude that PCA+LDA features partially eliminate sensitivity of illumination.	-

6	PCA + AAM [24] (Image sequences from FG-NET consortium)	The performanc e ratios are 100 % for Expression recognition from extracted faces,	The computational time and complexity was also very small. Improve the Efficiency	Extend the work to identify the face and it‟s expressions from 3D images.

Table.4.2.

5.SOFTWARE REQUIREMENT

As the project is developed in python, we have used Anaconda for Python 3.6.5 and Jupyter Notebook.

- **Anaconda**

  Itis a free and open source distribution of the Python and R programming languages for data science and machine learning related applications (large-scale data processing, predictive analytics, scientific computing), that aims to simplify package management and deployment. Package versions are managed by the package management system *conda*. The Anaconda distribution is used by over 6 million users, and it includes more than 250 popular data science packages suitable for Windows, Linux, and MacOS.

- **Jupyter Notebook**

  Jupyter Notebook is an open source cross-platform integrated development environment (IDE) for scientific programming in the Python language. Spyder integrates NumPy, SciPy, Matplotlib and IPython, as well as other open source software. It is released under the MIT license.

❖ Hardware Interfaces

1. **Processor :** Intel CORE i5 processor with minimum 2.9 GHz speed.
2. **RAM :** Minimum 4 GB.
3. **Hard Disk :** Minimum 500 GB

❖ Software Interfaces

1. Microsoft Word 2003
2. **Database Storage :** Microsoft Excel
3. **Operating System :** Windows 11

6.PLANNING

The steps we followed while developing this project are-:

1. Analysis of the problem statement.
2. Gathering of the requirement specification
3. Analysation of the feasibility of the project.
4. Development of a general layout.
5. Going by the journals regarding the previous related works on this field.
6. Choosing the method for developing the algorithm.
7. Analyzing the various pros and cons.
8. Starting the development of the project
9. Installation of software like ANACONDA.
10. Developing an algorithm.
11. Analysation of algorithm by guide.
12. Coding as per the developed algorithm in PYTHON.

We developed this project as per the iterative waterfall model:

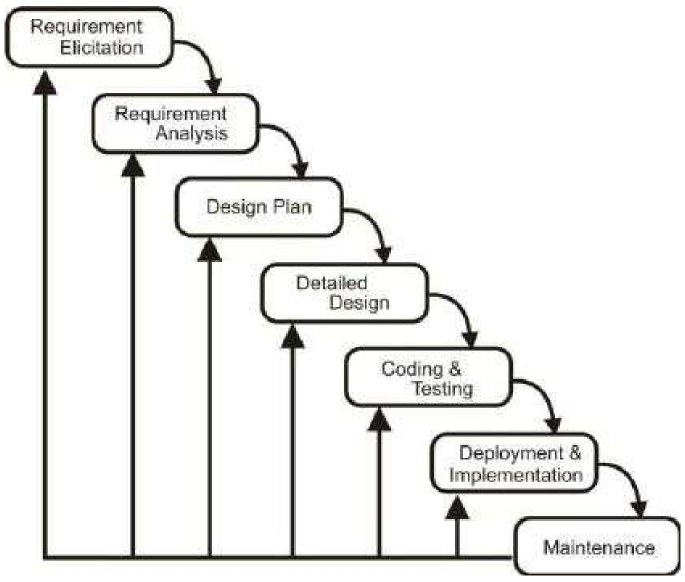


Fig.6.1.

7. ALGORITHM

Step 1 : Collection of a data set of images. (In this case we are using FER2013 database of **35887 pre-cropped, 48-by-48-pixel grayscale images** of faces each labeled with one of the 7 emotion classes: anger, disgust, fear, happiness, sadness, surprise, and neutral.

Step 2 : Pre-processing of images.

Step 3 : Detection of a face from each image.

Step 4 : The cropped face is converted into grayscale images.

Step 5 : The pipeline ensures every image can be fed into the input layer as a (1, 48, 48) numpy array.

Step 5 : The numpy array gets passed into the Convolution2D layer.

Step 6 :Convolution generates feature maps.

Step 7 : Pooling method called MaxPooling2D that uses (2, 2) windows across the feature map only keeping the maximum pixel value.

Step 8 : During training, Neural network Forward propagation and Backward propagation performed on the pixel values.

Step 9 : The Softmax function presents itself as a probability for each emotion class. The model is able to show the detail probability composition of the emotions in the face.

Step 10: The model presents the feature value in live time on screen highlighting the grey Scale part

8. IMPLEMENTATION DETAILS

- **The Database :**

The dataset, used for training the model is from a Kaggle Facial Expression Recognition Challenge a few years back (FER2013). 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 in to one of seven categories (0=Angry, 1=Disgust, 2=Fear, 3=Happy, 4=Sad, 5=Surprise, 6=Neutral).

The training set consists of 28,709 examples. The public test set used for the leaderboard consists of 3,589 examples. The final test set, which was used to determine the winner of the competition, consists of another 3,589 examples.

Emotion labels in the dataset:

0: -4593 images- ***Angry***

1: -547 images- ***Disgust***

2: -5121 images- ***Fear***

3: -8989 images- ***Happy***

4: -6077 images- ***Sad***

5: -4002 images- ***Surprise***

6: -6198 images- ***Neutral***

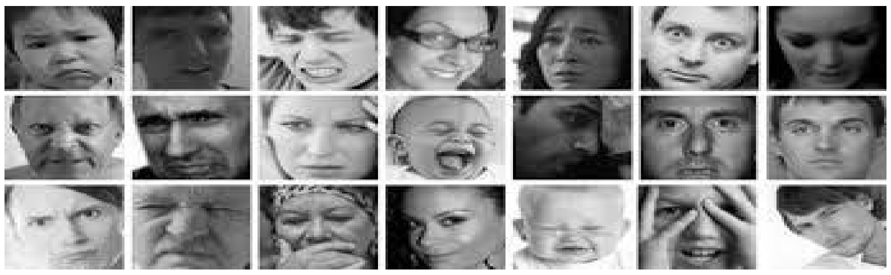


Fig.8.1.

The Library & Packages :

- **OpenCV :**

  OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. Being a BSD-licensed product, OpenCV makes it easy for businesses to utilize and modify the code.

- **Numpy :**

  NumPy is an acronym for "Numeric Python" or "Numerical Python". It is an open source extension module for Python, which provides fast precompiled functions for mathematical and numerical routines. Furthermore, NumPy enriches the programming language Python with powerful data structures for efficient computation of multi-dimensional arrays and matrices. The implementation is even aiming at huge matrices and arrays. Besides that the module supplies a large library of high-level mathematical functions to operate on these matrices and arrays.

  It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

  - A powerful N-dimensional array object
  - Sophisticated (broadcasting) functions
  - Tools for integrating C/C++ and Fortran code
  - Useful linear algebra, Fourier Transform, and random number capabilities.

- **SciPy :**

  SciPy (Scientific Python) is often mentioned in the same breath with NumPy. SciPy extends the capabilities of NumPy with further useful functions for minimization, regression, Fourier-transformation and many others.

  NumPy is based on two earlier Python modules dealing with arrays. One of these is Numeric. Numeric is like NumPy a Python module for high-performance, numeric computing, but it is obsolete nowadays. Another predecessor of NumPy is Num array, which is a complete rewrite of Numeric but is deprecated as well.

- **The Python Alternative To Matlab :**

  Python in combination with Numpy, Scipy and Matplotlib can be used as a replacement for MATLAB. The combination of NumPy, SciPy and Matplotlib is a free (meaning both "free" as in "free beer" and "free" as in "freedom") alternative to MATLAB. Even though MATLAB has a huge number of additional toolboxes available, NumPy has the advantage that Python is a more modern and complete programming language and - as we have said already before - is open source. SciPy adds even more MATLAB-like functionalities to Python. Python is rounded out in the direction of MATLAB with the module Matplotlib, which provides MATLAB-like plotting functionality.

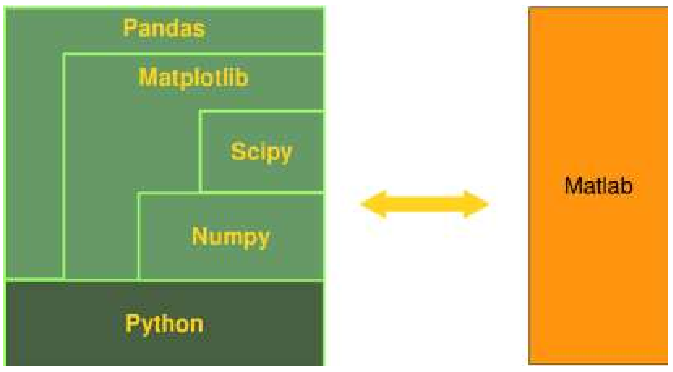


Fig.8.2.

- **Keras :**

  Keras is a high-level neural networks API, written in Python and capable of running on top of TensorFlow, CNTK, or Theano. It was developed with a focus

  on enabling fast experimentation.

  Keras contains numerous implementations of commonly used neural network building blocks suchas layers, objectives, activation functions, optimizers, and a host of tools to make working with image and text data easier. The code is hosted on GitHub, and community support forums include the GitHub issues page, and a Slack channel.

- **TensorFlow :**

  TensorFlow is a Python library for fast numerical computing created and released by Google. It is a foundation library that can be used to create Deep Learning models directly or by using wrapper libraries that simplify the process built on top ofTensorFlow.

ConvolutionalNeuralNetworks(CNN):

Convolutional Neural Networks are very similar to ordinary Neural Networks from the previous chapter: they are made up of neurons that have learnable weights and biases. Each neuron receives some inputs, performs a dot product and optionally follows it with a non-linearity.

o **<u>Overview of CNN architecture :</u>**

DCNNs are feedforward networks in that information flow takes place in one direction only, from their inputs to their outputs. Just as artificial neural networks (ANN) are biologically inspired, so are CNNs. The visual cortex in the brain, which consists of alternating layers of simple and complex cells (Hubel & Wiesel, 1959, 1962), motivates their architecture. CNN architectures come in several variations; however, in general, they consist of convolutional and pooling (or subsampling) layers, which are grouped into modules. Either one or more fully connected layers, as in a standard feedforward neural network, follow these modules. Modules are often stacked on top of each other to form a deep model. It illustrates typical CNN architecture for a toy image classification task. An image is input directly to the network, and this is followed by several stages of convolution and pooling. Thereafter, representations from these operations feed one or more fully connected layers. Finally, the last fully connected layer outputs the class label. Despite this being the most popular base architecture found in the literature, several architecture changes have been proposed in recent years with the objective of improving image classification accuracy or reducing computation costs. Although for the remainder of this section, we merely fleetingly introduce standard CNN architecture.

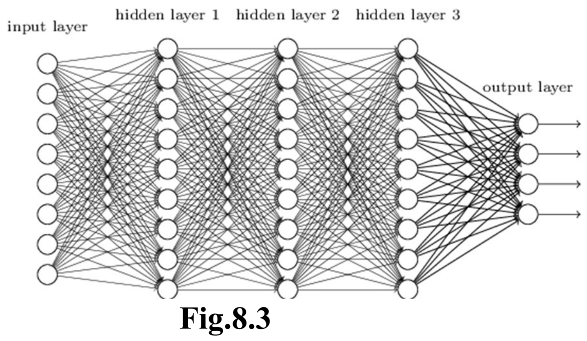


Fig.8.3

- **Pooling Layers :**

  The purpose of the pooling layers is to reduce the spatial resolution of the feature maps and thus achieve spatial invariance to input distortions and translations. Initially, it was common practice to use average pooling aggregation layers to propagate the average of all the input values, of a small neighborhood of an image to the next layer. However, in more recent models, , max pooling aggregation layers propagate the maximum value within a receptive field to the next layer.

- **Fully Connected Layers :**

  Several convolutional and pooling layers are usually stacked on top of each other to extract more abstract feature representations in moving through the network. The fully connected layers that follow these layers interpret these feature representations and perform the function of high-level reasoning. . For classification problems, it is standard to use the softmax operator on top of a DCNN. While early success was enjoyed by using radial basis functions (RBFs), as the classifier on top of the convolutional towers found that replacing the softmax operator with a support vector machine (SVM) leads to improved classification accuracy.

- **Training :**

  CNNs and ANN in general use learning algorithms to adjust their free parameters in order to attain the desired network output. The most common algorithm used for this purpose is backpropagation.

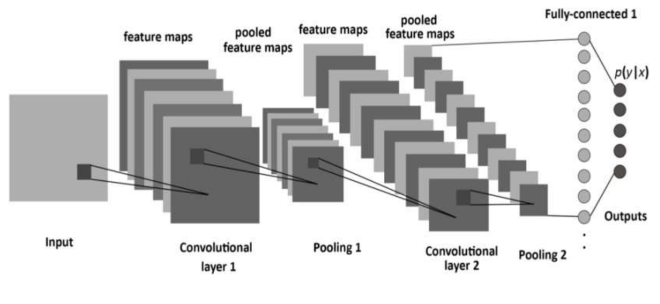


Fig.8.4

9. IMPLEMENTATION OF PROBLEM

- **The Database :**

  The dataset we used for training the model is from a Kaggle Facial Expression Recognition Challenge a few years back (FER2013). It comprises a total of **35887 pre-cropped, 48-by-48-pixel grayscale images** of faces each labeled with one of the emotion classes: **anger, disgust, fear, happiness, sadness, surprise, and neutral**.

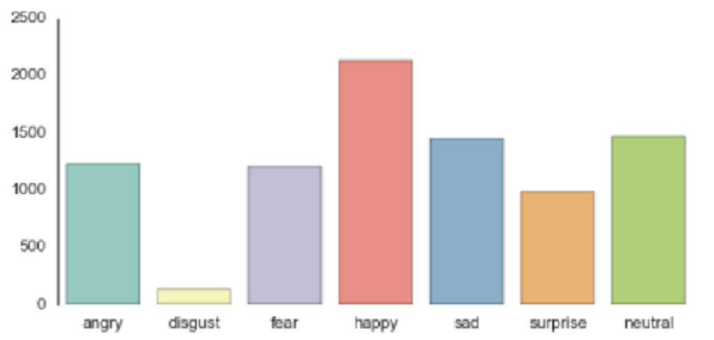


Fig.9.1.

As we were exploring the dataset, we discovered an imbalance of the "disgust" class compared to many samples of other classes. We decided to merge disgust into anger given that they both represent similar sentiment. To prevent data leakage, We built a data generator **fer2013datagen.**py that can easily separate training and hold-out set to different files. We used 28709 labeled faces as the training set and held out the remaining two test sets (3589/set) for after-training validation. The resulting is a **6-class, balanced dataset**, that contains angry, fear, happy, sad, surprise, and neutral. Now we‟re ready to train.

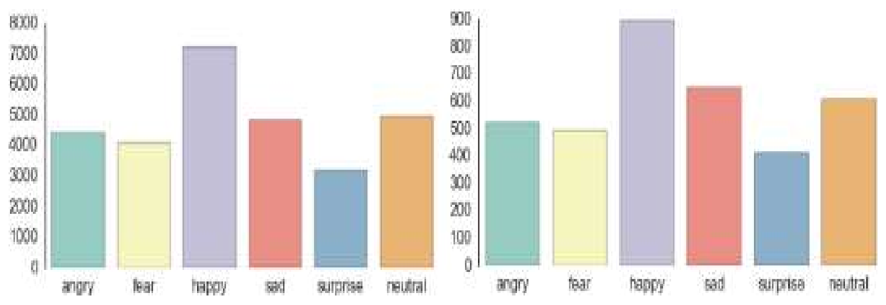


Fig.9.2.

- **Input Layer :**

  The input layer has pre-determined, fixed dimensions, so the image must be **pre-processed** before it can be fed into the layer. We used OpenCV, a computer vision library, for face detection in the image. The haar-cascade_frontalface_default.xml in OpenCV contains pre-trained filters and uses Adaboost to quickly find and crop the face.

- **Convolutional Layers :**

  The numpy array gets passed into the Convolution2D layer where we specify the number of filters as one of the hyperparameters. The set of filters(aka. kernel) are unique with randomly generated Weights.

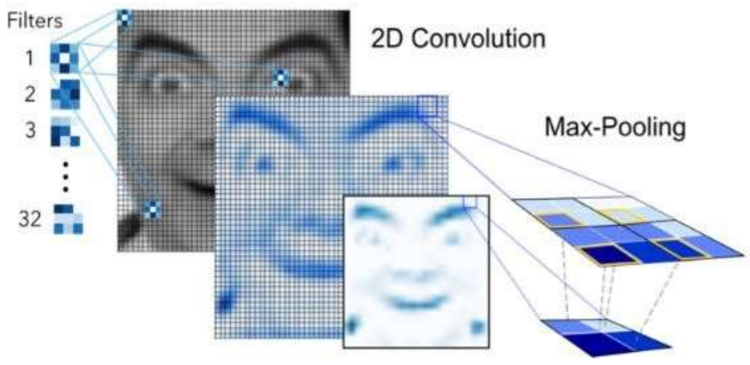


Fig.9.3.

o **Output Layer: -**

Deep Learning we built a simple CNN with an input, three convolution layers, one dense layer, and an output layer to start with. As it turned out, the simple model performed poorly. The low accuracy of 0.1500 showed that it was merely random guessing one of the six emotions. The simple net architecture failed to pick up the subtle details in facial expressions. This could only mean one thing...

This is where deep learning comes in. Given the pattern complexity of facial expressions, it is necessary to build with a deeper architecture in order to identify subtle signals. So we fiddled combinations of three components to increase model complexity:

Models with various combinations were trained and evaluated using GPU computing g2.2xlarge on Amazon Web Services (AWS). This greatly reduced training time and increased efficiency in tuning the model. In the end, our final net architecture was 9 layers deep in convolution with one max-pooling after every three convolution layers as seen below.

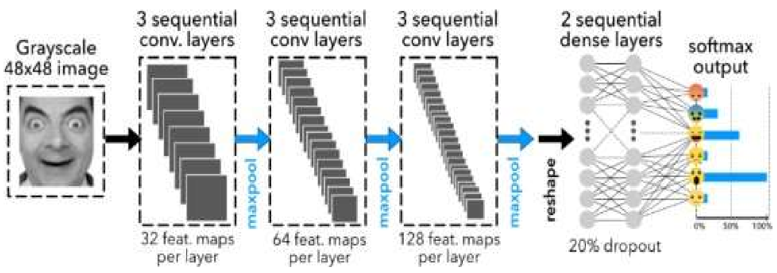


Fig.9.4.

 Model Validation :

Performance As it turns out, the final CNN had a **validation accuracy of 58%**. This actually makes a lot of sense. Because our expressions usually consist a combination of emotions, and *only* using one label to represent an expression can be hard. **In this case, when the model predicts incorrectly, the correct label is often the second most likely emotion as seen in figure below.**

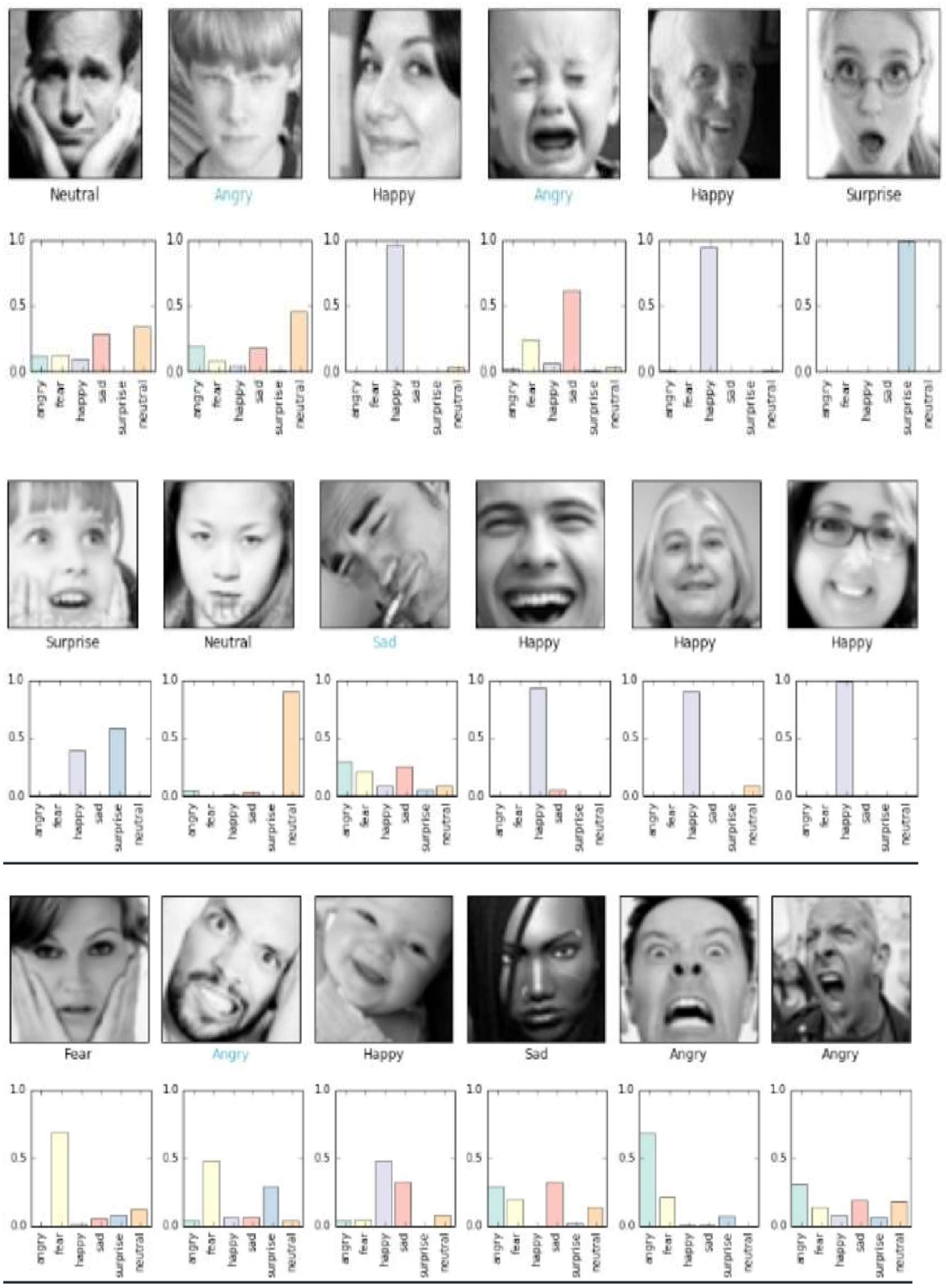
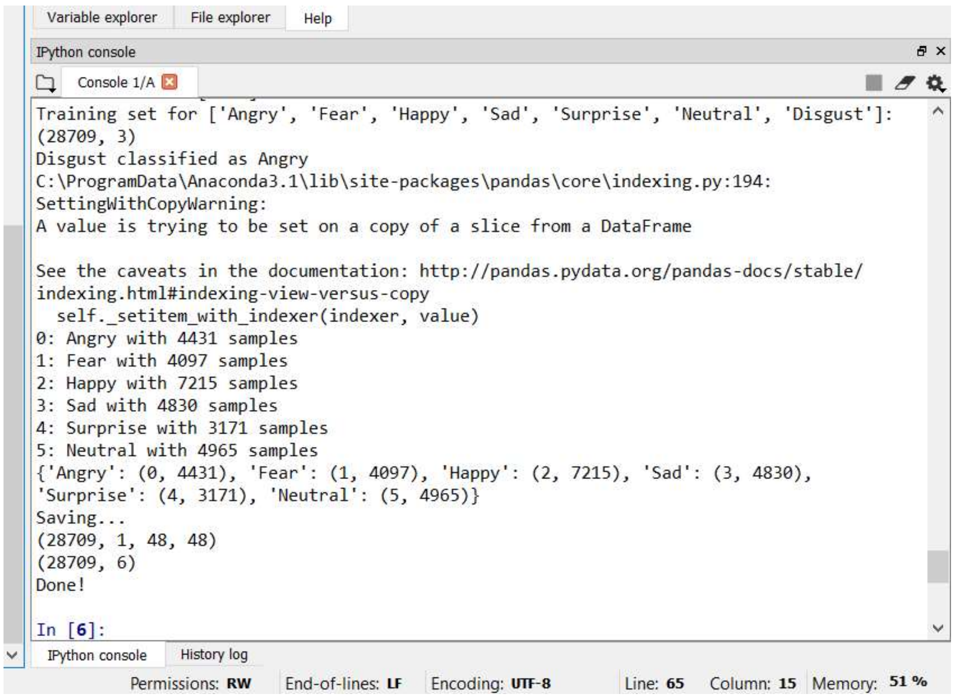

Neutral
Angry
Happy
Angry
Happy
Surprise
Surprise
Neutral
Sad
Happy
Happy
Happy
Fear
Angry
Happy
Sad
Angry
Angry
1.0
0.5
0.0
angry
fear
happy
sad
surprise
neutral

Fig.9.5.

10.RESULT

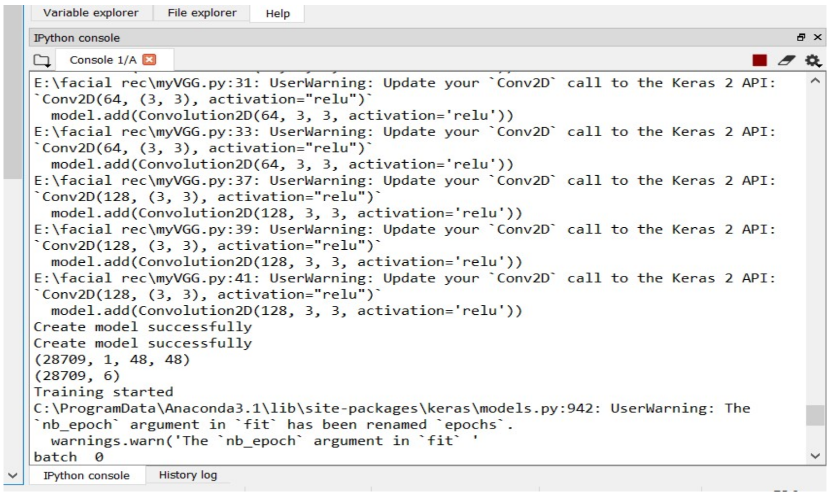
Dataset Trained Successfully



```
Training set for ['Angry', 'Fear', 'Happy', 'Sad', 'Surprise', 'Neutral', 'Disgust']:
(28709, 3)
Disgust classified as Angry
C:\ProgramData\Anaconda3.1\lib\site-packages\pandas\core\indexing.py:194:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy
  self._setitem_with_indexer(indexer, value)
0: Angry with 4431 samples
1: Fear with 4097 samples
2: Happy with 7215 samples
3: Sad with 4830 samples
4: Surprise with 3171 samples
5: Neutral with 4965 samples
{'Angry': (0, 4431), 'Fear': (1, 4097), 'Happy': (2, 7215), 'Sad': (3, 4830),
'Surprise': (4, 3171), 'Neutral': (5, 4965)}
Saving...
(28709, 1, 48, 48)
(28709, 6)
Done!

In [6]:
```

Fig.10.1.

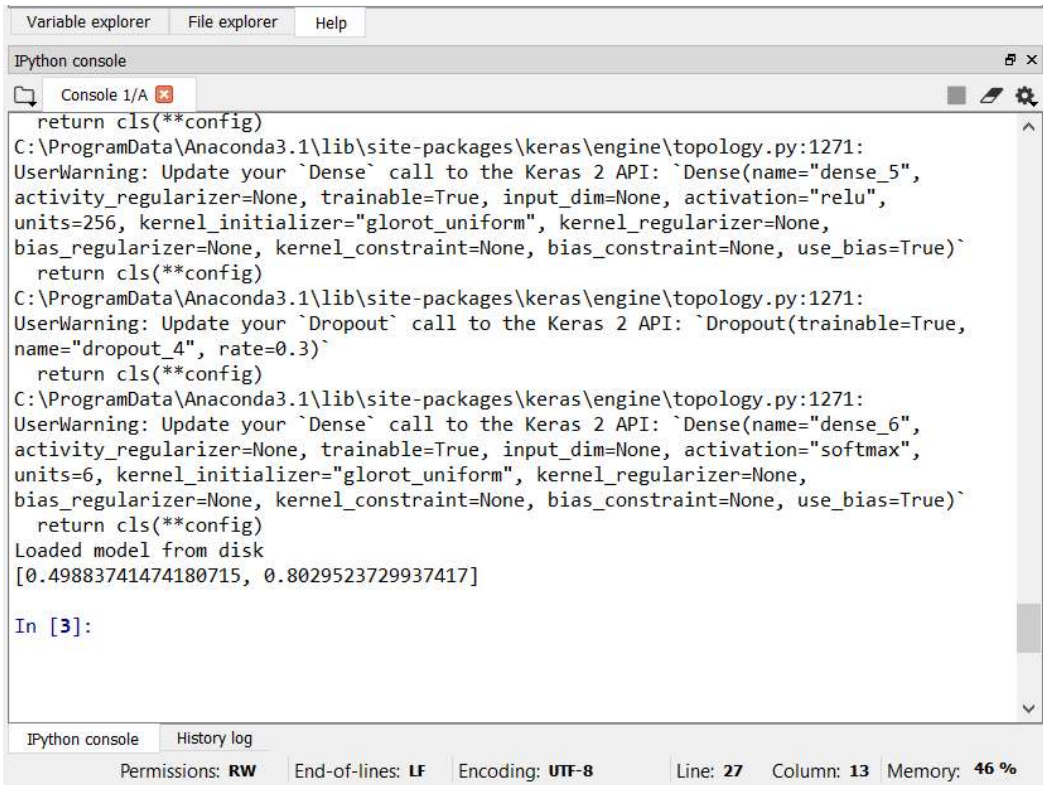
JSON Model Created Successfully



```
E:\facial rec\myVGG.py:31: UserWarning: Update your `Conv2D` call to the Keras 2 API:
`Conv2D(64, (3, 3), activation="relu")`
  model.add(Convolution2D(64, 3, 3, activation='relu'))
E:\facial rec\myVGG.py:33: UserWarning: Update your `Conv2D` call to the Keras 2 API:
`Conv2D(64, (3, 3), activation="relu")`
  model.add(Convolution2D(64, 3, 3, activation='relu'))
E:\facial rec\myVGG.py:37: UserWarning: Update your `Conv2D` call to the Keras 2 API:
`Conv2D(128, (3, 3), activation="relu")`
  model.add(Convolution2D(128, 3, 3, activation='relu'))
E:\facial rec\myVGG.py:39: UserWarning: Update your `Conv2D` call to the Keras 2 API:
`Conv2D(128, (3, 3), activation="relu")`
  model.add(Convolution2D(128, 3, 3, activation='relu'))
E:\facial rec\myVGG.py:41: UserWarning: Update your `Conv2D` call to the Keras 2 API:
`Conv2D(128, (3, 3), activation="relu")`
  model.add(Convolution2D(128, 3, 3, activation='relu'))
Create model successfully
Create model successfully
(28709, 1, 48, 48)
(28709, 6)
Training started
C:\ProgramData\Anaconda3.1\lib\site-packages\keras\models.py:942: UserWarning: The
`nb_epoch` argument in `fit` has been renamed `epochs`.
  warnings.warn('The `nb_epoch` argument in `fit` '
batch  0
```

Fig.10.2.

Accuracy Check Of The Model



```
  return cls(**config)
C:\ProgramData\Anaconda3.1\lib\site-packages\keras\engine\topology.py:1271:
UserWarning: Update your `Dense` call to the Keras 2 API: `Dense(name="dense_5",
activity_regularizer=None, trainable=True, input_dim=None, activation="relu",
units=256, kernel_initializer="glorot_uniform", kernel_regularizer=None,
bias_regularizer=None, kernel_constraint=None, bias_constraint=None, use_bias=True)`
  return cls(**config)
C:\ProgramData\Anaconda3.1\lib\site-packages\keras\engine\topology.py:1271:
UserWarning: Update your `Dropout` call to the Keras 2 API: `Dropout(trainable=True,
name="dropout_4", rate=0.3)`
  return cls(**config)
C:\ProgramData\Anaconda3.1\lib\site-packages\keras\engine\topology.py:1271:
UserWarning: Update your `Dense` call to the Keras 2 API: `Dense(name="dense_6",
activity_regularizer=None, trainable=True, input_dim=None, activation="softmax",
units=6, kernel_initializer="glorot_uniform", kernel_regularizer=None,
bias_regularizer=None, kernel_constraint=None, bias_constraint=None, use_bias=True)`
  return cls(**config)
Loaded model from disk
[0.49883741474180715, 0.8029523729937417]

In [3]:
```

Fig.10.3.

11. CONCLUSION

In this case, when the model predicts incorrectly, the correct label is often the second most likely emotion.

The facial expression recognition system presented in this research work contributes a resilient face recognition model based on the mapping of behavioral characteristics with the physiological biometric characteristics. The physiological characteristics of the human face with relevance to various expressions such as happiness, sadness, fear, anger, surprise and disgust are associated with geometrical structures which restored as base matching template for the recognition system.

The behavioral aspect of this system relates the attitude behind different expressions as property base. The property bases are alienated as exposed and hidden category in genetic algorithmic genes. The gene training set evaluates the expressional uniqueness of individual faces and provide a resilient expressional recognition model in the field of biometric security.

The design of a novel asymmetric cryptosystem based on biometrics having features like hierarchical group security eliminates the use of passwords and smart cards as opposed to earlier cryptosystems. It requires a special hardware support like all other biometrics system. This research work promises a new direction of research in the field of asymmetric biometric cryptosystems which is highly desirable in order to get rid of passwords and smart cards completely. Experimental analysis and study show that the hierarchical security structures are effective in geometric shape identification for physiological traits.

12.FUTURE SCOPE

It is important to note that there is no specific formula to build a neural network that would guarantee to work well. Different problems would require different network architecture and a lot of trail and errors to produce desirable validation accuracy. **This is the reason why neural nets are often perceived as "black box algorithms."**.

In this project we got an accuracy of almost 70% which is not bad at all comparing all the previous models. But we need to improve in specific areas like-

- **number and configuration of convolutional layers**
- **number and configuration of dense layers**
- **dropout percentage in dense layers**

But due to lack of highly configured system we could not go deeper into dense neural network as the system gets very slow and we will try to improve in these areas in future.

We would also like to train more databases into the system to make the model more and more accurate but again resources becomes a hindrance in the path and we also need to improve in several areas in future to resolve the errors and improve the accuracy.

Having examined techniques to cope with expression variation, in future it may be investigated in more depth about the face classification problem and optimal fusion of color and depth information. Further study can be laid down in the direction of allele of gene matching to the geometric factors of the facial expressions. The genetic property evolution framework for facial expressional system can be studied to suit the requirement of different security models such as criminal detection, governmental confidential security breaches etc.

13.REFERENCES

[1]. **A literature survey on Facial Expression Recognition using Global Features** by Vaibhav kumar, J. Mistry and Mahesh M.Goyani, International Journal of Engineering and Advanced Technology (IJEAT),April, 2013 [http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.645.5162&rep=rep1&typ e=pdf]

[2]. **Convolutional Neural Networks (CNN) With TensorFlow** by Sourav from Edureka[https://www.youtube.com/watch?v=umGJ30-15_A]

[3]. **A Survey on Facial Expression Recognition Techniques** by Swati Mishra(Chhattisgarh Swami Vivekanand Technical University, Department of Computer Science & Engineering, Raipur Institute of Technology, Mandir Hasaud, Raipur, Chhattisgarh, India) and Avinash Dhole(Assistant Professor, Department of Computer Science & Engineering, Raipur Institute of Technology, Chhattisgarh Swami Vivekanand and Technical University, Mandir Hasaud, Raipur, Chhattisgarh, India) , International Journal of Science and Research (IJSR),2013 [https://pdfs.semanticscholar.org/e241/25e4e9471a33ba2c7f0979541199caa02f8b.pd f]

[4]. **Recognizing Facial Expressions Using Deep Learning** by Alexandru Savoiu Stanford University and James Wong Stanford University [http://cs231n.stanford.edu/reports/2017/pdfs/224.pdf]

[5]. **Deep Learning Simplified** by Sourav from Edureka[https://www.youtube.com/watch?v=dafuAz_CV7Q&list=PL9ooVrP1hQOEX8BK D plfG86ky8s7Oxbzg]

[6]. **Predicting facial expressions with machine learning algorithms** by Alex Young, Andreas Eliasson, Lukas Weiss, Ara Hayrabedian.

[7]. **"Robust Real-Time Face Detection"**, International Journal of Computer Vision 57(2), 137-154,2004.

[8].“**Facial expressions of emotions: an old controversy and new finding discussion**”, by P.

Ekman, E. T. Rolls, D. I. Perrett, H. D. Ellis, Pill Trans. Royal Soc. London Ser. B, Biol. Sci., 1992, vol. 335, no. 1273, pp. 63-69.

[9].Going Deeper in Facial Expression Recognition using Deep Neural Networks, by Ali Mollahosseini1, David Chan2, and Mohammad H. Mahoor1 Department of Electrical and Computer Engineering, Department of Computer Science, University of Denver, Denver, CO

[10]. Journal on Convoluted Neural Network by IIIT,Hyderabad.

[11]. Journal on Artificial Intellegence by Prof. M.K. Anand, 2014.

[12].Journal by Ghuangjhow University on Image Processing.

[13]. Wikipedia- Artificial Neural Netwok & Convoluted Neural Netwok

[14]. Neeta Sarode et. Al ./(IJCSE) International Journal on Computer Science.

[15]. Facial Expression Detection Techniques: Based on Viola and Jones algorithm and Principal Component Analysis by Samiksha Agrawal and Pallavi Khatri, ITM University Gwalior(M.P.), India, 2014.

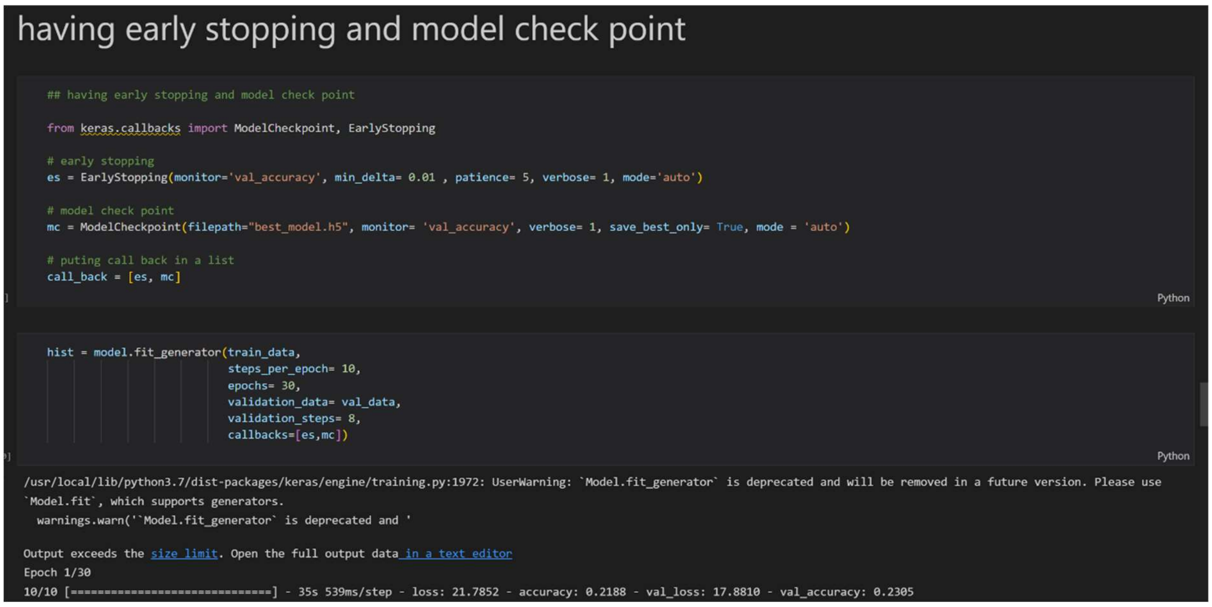
[16]. Project_Report by Udacity, INDIA on **Image Processing,** 2013

14. APPENDIX

Videotester.py

```
import os
import cv2
import numpy as np
from keras.preprocessing import image
import warnings
warnings.filterwarnings("ignore")
#from keras.preprocessing.image import load_img, img_to_array
from tensorflow.keras.utils import img_to_array
import keras.utils as image
from keras.models import  load_model
import matplotlib.pyplot as plt
import numpy as np

# load model
model = load_model("best_model.h5")



face_haar_cascade=cv2.CascadeClassifier(cv2.data.haarcascades                                    +
'haarcascade_frontalface_default.xml')



cap = cv2.VideoCapture(0)

while True:
    ret, test_img = cap.read()  # captures frame and returns boolean value and captured image
    if not ret:
        continue
    gray_img = cv2.cvtColor(test_img, cv2.COLOR_BGR2RGB)

    faces_detected = face_haar_cascade.detectMultiScale(gray_img, 1.32, 5)

    for (x, y, w, h) in faces_detected:
        cv2.rectangle(test_img, (x, y), (x + w, y + h), (255, 0, 0), thickness=7)
        roi_gray = gray_img[y:y + w, x:x + h]  # cropping region of interest i.e. face area from  image
        roi_gray = cv2.resize(roi_gray, (224, 224))
        img_pixels = image.img_to_array(roi_gray)
        img_pixels = np.expand_dims(img_pixels, axis=0)
        img_pixels /= 255

        predictions = model.predict(img_pixels)

        # find max indexed array
        max_index = np.argmax(predictions[0])

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

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

    resized_img = cv2.resize(test_img, (1000, 700))
    cv2.imshow('Facial emotion analysis ', resized_img)

    if cv2.waitKey(10) == ord('q'):  # wait until 'q' key is pressed
        break

cap.release()
cv2.destroyAllWindows
```

Emotion_Detection.ipyn

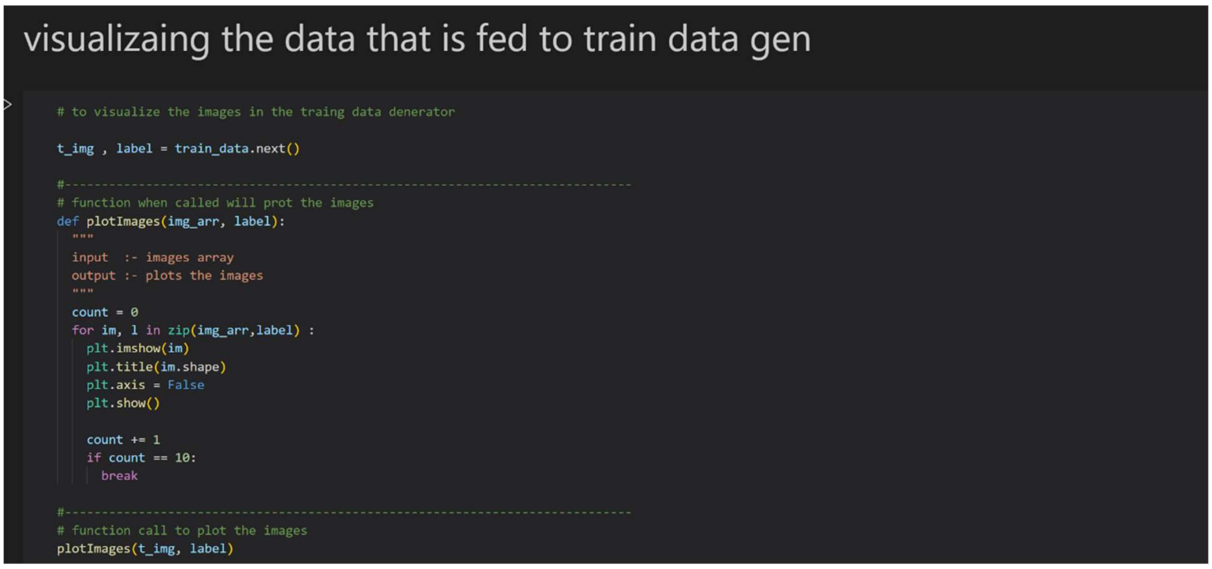


having early stopping and model check point

```python
## having early stopping and model check point

from keras.callbacks import ModelCheckpoint, EarlyStopping

# early stopping
es = EarlyStopping(monitor='val_accuracy', min_delta= 0.01 , patience= 5, verbose= 1, mode='auto')

# model check point
mc = ModelCheckpoint(filepath="best_model.h5", monitor= 'val_accuracy', verbose= 1, save_best_only= True, mode = 'auto')

# puting call back in a list
call_back = [es, mc]
```

```python
hist = model.fit_generator(train_data,
                           steps_per_epoch= 10,
                           epochs= 30,
                           validation_data= val_data,
                           validation_steps= 8,
                           callbacks=[es,mc])
```

```
/usr/local/lib/python3.7/dist-packages/keras/engine/training.py:1972: UserWarning: `Model.fit_generator` is deprecated and will be removed in a future version. Please use
`Model.fit`, which supports generators.
  warnings.warn('`Model.fit_generator` is deprecated and '

Output exceeds the size limit. Open the full output data in a text editor
Epoch 1/30
10/10 [==============================] - 35s 539ms/step - loss: 21.7852 - accuracy: 0.2188 - val_loss: 17.8810 - val_accuracy: 0.2305
```



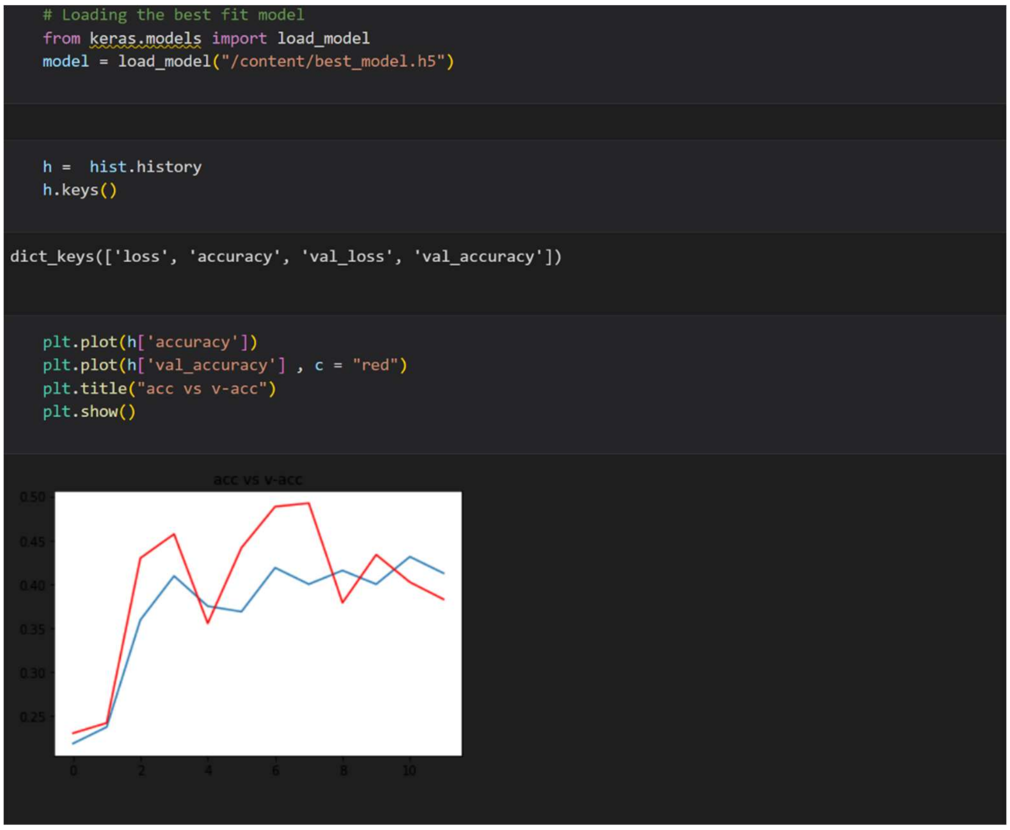
visualizaing the data that is fed to train data gen

```python
# to visualize the images in the traing data denerator

t_img , label = train_data.next()

#----------------------------------------------------------------------------
# function when called will prot the images
def plotImages(img_arr, label):
  """
  input  :- images array
  output :- plots the images
  """
  count = 0
  for im, l in zip(img_arr,label) :
    plt.imshow(im)
    plt.title(im.shape)
    plt.axis = False
    plt.show()

    count += 1
    if count == 10:
      break

#----------------------------------------------------------------------------
# function call to plot the images
plotImages(t_img, label)
```

(224, 224, 3)
(224, 224, 3)

(224, 224, 3)
(224, 224, 3)

(224, 224, 3)
(224, 224, 3)
0
50
100
150
200

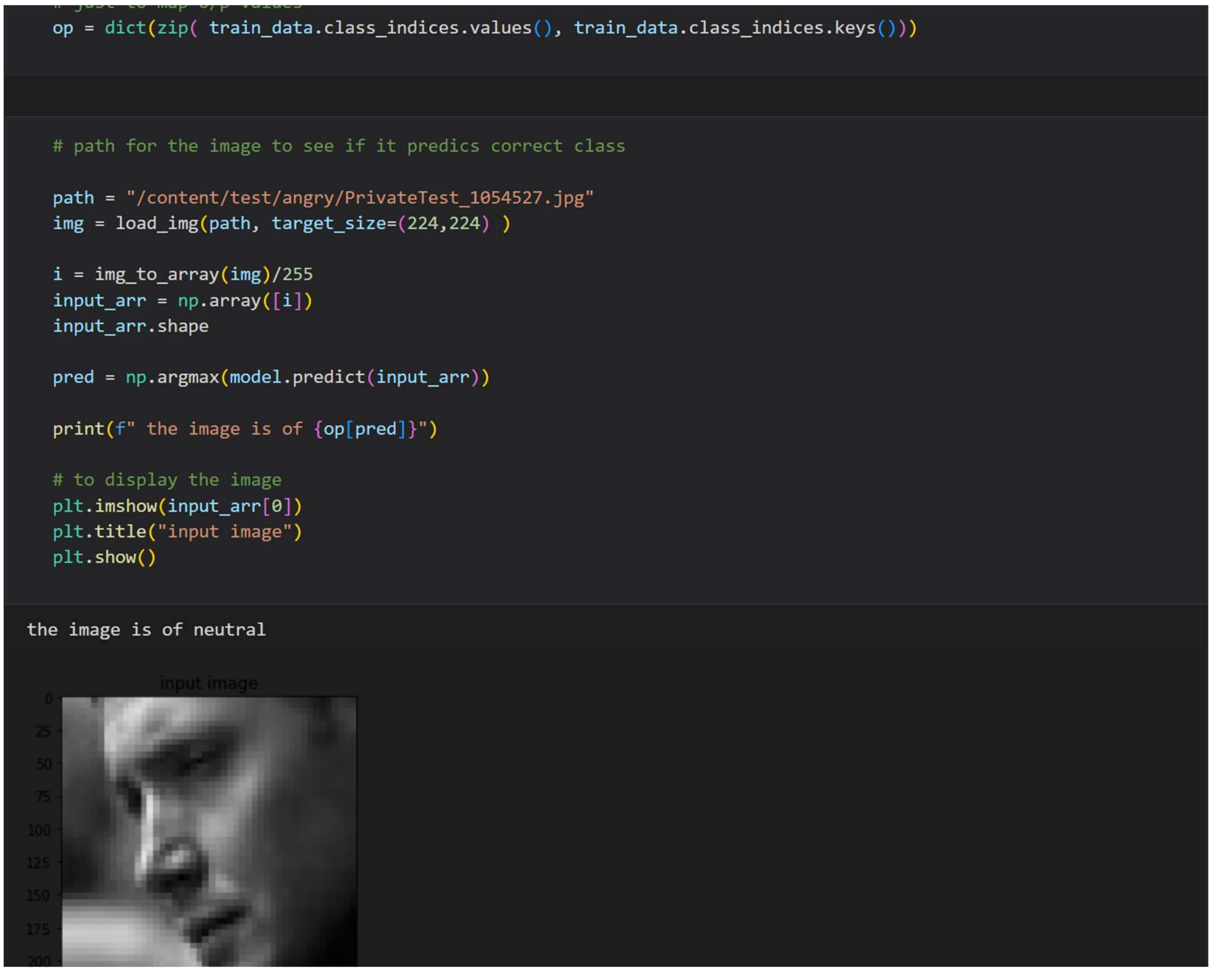


```
# Loading the best fit model
from keras.models import load_model
model = load_model("/content/best_model.h5")
```

```
h =  hist.history
h.keys()
```

```
dict_keys(['loss', 'accuracy', 'val_loss', 'val_accuracy'])
```

```
plt.plot(h['accuracy'])
plt.plot(h['val_accuracy'] , c = "red")
plt.title("acc vs v-acc")
plt.show()
```



```
# just to map o/p values
op = dict(zip( train_data.class_indices.values(), train_data.class_indices.keys()))
```

```
# path for the image to see if it predics correct class

path = "/content/test/angry/PrivateTest_1054527.jpg"
img = load_img(path, target_size=(224,224) )

i = img_to_array(img)/255
input_arr = np.array([i])
input_arr.shape

pred = np.argmax(model.predict(input_arr))

print(f" the image is of {op[pred]}")

# to display the image
plt.imshow(input_arr[0])
plt.title("input image")
plt.show()
```

```
the image is of neutral
```