A
FINAL PROJECT REPORT
ON

"Facial Emotion Recognition Using Convolutional Neural Networks"

SUBMITTED TO SAVITRIBAI PHULE PUNE UNIVERSITY FOR THE PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

BACHELOR OF ENGINEERING
IN
ELECTRONICS AND TELECOMMUNICATION

Submitted by

Venkatesh Mullur
Atharva Dastane
Viraj Sanap

Under the guidance of

Prof. Mrs. Trupti Kudale

Department of Electronics and Telecommunication Engineering
Pune Vidyarthi Griha's College of Engineering and Technology,
Pune-411009

Academic Year 2019 - 2020

Undertaking

We, the students of Department of Electronics and Telecommunication Engineering, PVG's College of Engineering and Technology, Parvati, Pune-411009

1. Venkatesh Mullur EXAM SEAT NO.:71701605G SIGNATURE:
2. Atharva Dastane EXAM SEAT NO.:71701386D SIGNATURE:
3. Viraj Sanap EXAM SEAT NO.:71701696L SIGNATURE:

do hereby undertake on 01.06.2019 the following:

1. We are aware of the **University Grants Commission (Promotion Of Academic Integrity and Prevention of Plagiarism In Higher Educational Institutions) Regulations, 2018 and Plagiarism Policy of Savitribai Phule Pune University Pune,**
2. We are also aware that **Department of Electronics and Telecommunication Engineering, PVG's College of Engineering and Technology, Parvati, Pune** has established **Departmental Academic Integrity Panel (DAIP)** as per **UGC regulations 2018** as mentioned above in (1),
3. We are aware of the consequences if found guilty of doing any act of plagiarism defined in the UGC regulations 2018 as mentioned above in (1),
4. We shall abide by the rules, regulations and code of conducts for the students of UGC, SPPU and Department of E&TC,
5. We further undertake and declare that the thesis submitted by us is scanned using anti-plagiarism software as decided by the department and report of the same has been submitted and is free from any kind of plagiarism mentioned above (1) in **UGC REGULATIONS, 2018,**
6. I understand that non-compliance of the **Academic Integrity and Prevention of Plagiarism** may results in **disciplinary action on us as per the University Grants Commission (Promotion of Academic Integrity and Prevention of Plagiarism in Higher Educational Institutions) Regulations, 2018.**

Department of Electronics and Telecommunication Engineering
Pune Vidyarthi Griha's College of Engineering and Technology,
Pune -411009

CERTIFICATE

This is to certify that the project entitled, **"Facial Emotion Recognition"**, is a work of

1. Venkatesh Mullur EXAM SEAT NO.:71701605G
2. Atharva Dastane EXAM SEAT NO.:71701386D
3. Viraj Sanap EXAM SEAT NO.:71701696L

Towards the partial fulfillment of the degree of **Bachelor of Engineering in Electronics and Telecommunication Engineering** to be awarded by the **Savitribai Phule Pune University, Pune** at **Pune Vidyarthi Griha's College of Engineering and Technology, Pune** during the academic year 2019-2020.

Prof. Mrs. Trupti Kudale	**Dr.Prof.Y.B.Thakre**	**Prof.K.J.Kulkarni**
Project Guide	**HOD E&TC**	**Principal**

Place: Pune

Date:

Contents

List of Figures

ACKNOWLEDGEMENT

We are very thankful to our project guide Prof. Mrs.Trupti Kudale and Prof. Meenakshi Atre, Department of Electronics and Telecommunication Engineering, Pune Vidyarthi Griha's College of Engineering and Technology, Pune; for her invaluable guidance and assistance, without which the accomplishment of the task would never have been possible. We also thank her for giving us an opportunity to explore into the real world and realize the interrelation. We extend our most sincere thanks to her for her cooperation during the project work and also, we are thankful to all the professors and teachers for the support and cooperation. It is a privilege to express our gratitude to our teachers and project guide for providing us an excellent environment to complete our work successfully.

VENKATESH MULLUR
ATHARVA DASTANE
VIRAJ SANAP

ABSTRACT

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 have identified five human emotions namely Neutral, Sad, Angry, Happy, Drowsy.

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 have identified five human emotions namely Neutral, Sad, Angry, Happy, Drowsy.

Chapter 1

Introduction:

1.1 Introduction of proposed system

Image processing is a method to perform some operations on an image, in order to get an enhanced image or to extract some useful information from it. It is a type of signal processing in which input is an image and output may be image or characteristics/features associated with that image. Nowadays, image processing is among rapidly growing technologies. It forms core research area within engineering and computer science disciplines too.
Image processing basically includes the following three steps:

- Importing the image via image acquisition tools;
- Analysing and manipulating the image;
- Output in which result can be altered image or report that is based on image analysis.

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.

1.2 Motivation

The main motivation behind selecting this topic was our curiosity as well as interest in various aspects of digital image processing. We wanted to explore in depth, various real world problems related to image processing and how to tackle their solutions by learning new techniques. Also an important role was also played by our faculty members in selecting this domain.

1.3 Objectives

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 FIVE different expression class such as :

- Happy
- Neutral
- Sad
- Angry
- Surprise
- Disgust

1.4 Expected Outcomes

- The system should successfully capture an image and process it.
- The system will then detect face and find out the emotion.
- The system will classify and display one of the five emotions.

1.5 Social Relevance

- Using this system the output can be used for market research in various areas.
- The system is designed to detect drowsiness of a driver and alert him for a nap or a break.
- Also happy and sad emotions are used to display a viewers reactions toward a movie and based on the reactions movie reviews can be improved. Also the genre of movie can be decided based on these emotions.
- System can also be used for detecting angry emotion, it can tell the impact video games can have on the user (player) .

1.6 Organization of Report

Literature Survey

Design of the Proposed System
The proposed system consists of several different steps which are dependent on each other for their functioning. Starting with data collection , preprocessing ,filtering ,detection, feature extraction and emotion classification.

Simulation and Testing

This project will be developed using Python3. IDE used is Jupyter Notebook.With the help of various libraries in Python each block the project will be developed and tested to form an integrated outcome. Expected results and outcomes are included in this section.

Result Analysis and Conclusion

The preprocessing stage has been tested and simulated. Face detection of captured image is tested and simulated. Further operations will be carried out.

Chapter 2

Literature Survey

2.1 Face Detection Using Viola Jones Algorithm

This paper describes a machine learning approach for visual object detection which is capable of processing images extremely rapidly and achieving high detection rates [1].

This work is distinguished by three key contributions. The first is the introduction of a new image representation called the "Integral Image" which allows the features used by our detector to be computed very quickly. The second is a learning algorithm, based on AdaBoost, which selects a small number of critical visual features from a larger set and yields extremely efficient classifiers[2].

The third contribution is a method for combining increasingly more complex classifiers in a "cascade" which allows background regions of the image to be quickly discarded while spending more computation on promising object-like regions. The cascade can be viewed as an object specific focus-of-attention mechanism which unlike previous approaches provides statistical guarantees that discarded regions are unlikely to contain the object of interest.

In the domain of face detection the system yields detection rates comparable to the best previous systems. Used in real-time applications, the detector runs at 15 frames per second without resorting to image differencing or skin color detection.

2.2 Convolution Neural Network

Emotions are a powerful tool in communication and one way that humans show their emotions is through their facial expressions. One of the challenging and powerful tasks in social communications is facial expression recognition, as in non-verbal communication, facial expressions are key. In the field of Artificial Intelligence, Facial Expression Recognition (FER) is an active research area, with several recent studies using Convolutional Neural Networks (CNNs). In this paper, we demonstrate the classification of FER based on static images, using CNNs, without requiring any pre-processing or feature extraction tasks.

The paper also illustrates techniques to improve future accuracy in this area by using pre- processing, which includes face detection and illumination correction. Feature extraction is used to extract the most prominent parts of the face, including the jaw, mouth, eyes, nose, and eyebrows.

Furthermore, we also discuss the literature review and present our CNN architecture, and the challenges of using max-pooling and dropout, which eventually aided in better performance. We obtained a test accuracy of 61.7 percent on FER2013 in a seven-classes classification task compared to 75.2 percent in state-of-the-art classification.[3].

In order to reduce the complexity for extracting artificial features from the face image in facial expression recognition (FER), a novel method is proposed based on convolutional neural network (CNN) in this paper. This method first preprocesses the facial expression images, then some trainable convolution kernels are used to extract facial expression features, and second, the largest pooling layer is used to fewer dimensions, finally seven types of facial expressions are recognized with the Softmax classifier. The proposed method is verified with Kaggle facial expression recognition challenge dataset (FER2013). The experimental results show that the method has good recognition performance and generalization ability[4].

The human emotion recognition has attracted interest of many problem solvers in the field of artificial intelligence. The emotions on a human face say so much about our thought process and give a glimpse of what's going on inside the mind. Real time emotion recognition is to acquaint the machine with human like ability to recognize and analyse human emotions. This project aims to categorize a facial image into one of the seven emotions which we are considering in this study, by building a multi class classifier. In this paper we are using convolutional neural networks (CNNs) for training over gray scale images obtained from fer2013 dataset. We experimented with different depths and max pooling layers to get the best accuracy and ultimately achieving 89.98 percent accuracy. To combat overfitting, we have used technique like dropout. We are also analyzing the performance of different network architectures like shallow network and modern deep network in recognizing human emotion.

We also present the real-time implementation of emotion recognition in web-camera which provides accurate results for multiple faces simultaneously. The results obtained from the research are quite interesting. [11].

2.3 Principal Component Analysis, Eigen Faces

Eigen face or Principal Component Analysis (PCA) methods have demonstrated their success in face recognition, detection, and tracking. The representation in PCA is based on the second order statistics of the image set, and does not address higher order statistical dependencies such as the relationships among three or more pixels. Recently Higher Order Statistics (HOS) have been used as a more informative low dimensional representation than PCA for face detection.

We investigate a generalization of PCA, Kernel Principal Component Analysis (Kernel PCA), for learning low dimensional representations in the context of face recognition. In contrast to HOS, Kernel PCA computes the higher order statistics without the combinatorial explosion of time and memory complexity.

While PCA aims to find a second order correlation of patterns, Kernel PCA provides a replacement which takes into account higher order correlations. We compare the recognition results using kernel methods with Eigen face methods on two benchmarks. Empirical results show that Kernel PCA outperforms the Eigen face method in face recognition[7].

A kernel principal component analysis (PCA) was recently proposed as a nonlinear extension of a PCA. The basic idea is to first map the input space into a feature space via nonlinear mapping and then compute the principal components in that feature space. This letter adopts the kernel PCA as a mechanism for extracting facial features.

Through adopting a polynomial kernel, the principal components can be computed within the space spanned by high-order correlations of input pixels making up a facial image, thereby producing a good performance[9].

2.4 Support Vector Machine

Affective computing has become a growing field of research activities due to its wide use of application in human computer interface. Emotion recognition is one of the state-of the- art techniques in determining current psychological state of human being. Human emotions are very overlapping in nature and thus it needs an efficient feature-extractor and classifier assembly.

This paper reports a novel non-invasive technique to classify human emotion through thermal images of face. Hus moment invariants of different patches have been fused with histogram statistical feature and used as robust features in multiclass support vector machine based classification.

It is found that emotions from thermal image can be classified by the proposed method with a satisfactory performance[6].In this paper, two novel methods for facial expression recognition in facial image sequences are presented. The user has to manually place some of Candid grid nodes to face landmarks depicted at the first frame of the image sequence under examination.

The grid-tracking and deformation system used, based on deformable models, tracks the grid in consecutive video frames over time, as the facial expression evolves, until the frame that corresponds to the greatest facial expression intensity[7].

The geometrical displacement of certain selected Candid nodes, defined as the difference of the node coordinates between the first and the greatest facial expression intensity frame, is used as an input to a novel multiclass Support Vector Machine (SVM) system of classifiers that are used to recognize either the six basic facial expressions or a set of chosen Facial Action Units (FAUs).

The results on the Cohn–Kanade database show a recognition accuracy of 99.7% for facial expression recognition using the proposed multiclass SVMs and 95.1% for facial expression recognition based on FAU detection.

Comparison Table

Sr.No.	Method/Technique	Result/Accuracy	Conclusion	Future Work
1	Neural Network +Rough Contour Estimation Routine(RCER)	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] 35% less computational time and 100% Recognition	Useful where larger database and less computation time	They want to repeat their experiment on larger and different databases.	
3	PCA + Eigenfaces	83% Surprise develop CK, 85% happiness in JAFFE, Fear was the most confused expression.	Compared with the facial expressions recognition was based on the video sequence, the one based on the static image is more difficult due to lack of temporal information.	Future work is to a facial expression recognition system which combines body gestures of the user with user facial expression.
4	2D Gabor filter	Gabor Filter bank used to locate edges	Multichannel Gabor filtration Scheme used for the detection of salient points and the extraction features for image retrieval applications.	They work on adding global and local colour histogram and parameters connected with the shapes of objects within images.
5	Local Gabor Filter + PCA+LDA	Obtained 97.33% Recognition rate with the help of PCA+LDA features.	They conclude that PCA+LDA features partially eliminate sensitivity of illumination	

Chapter 3

DESIGN OF PROPOSED SYSTEM

3.1 Introduction

As per various literature surveys it is found that for implementing this project four basic steps are required to be performed. The objective was kept in mind while executing the design. Each block of the block diagram has been selected as per the requirements of the project. The block diagram steps are described in detailed and in depth.

3.2 Block Diagram Of Proposed System

3.2.1 Block Diagram

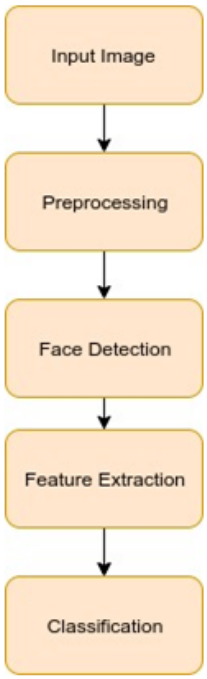

Figure 3.1: Block Diagram

3.3 Explanation Of Block Diagram

For our project we will require an input image which is obtained through a camera. The input image will be pre-processed before being used for emotion analysis. The pre-processing will block will make sure the image has all required features and will improve the image quality through image enhancement techniques. Also if the image is of poor quality the image restoration techniques would be done.

Once the image is pre-processed the face detection is done using the Voila Jones algorithm. Feature extraction is done after the face is identified from the face detection technique. Features required are then compared and found out through the database. Emotion classification is done based on set of parameters of features and information after the comparison.

3.4 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.

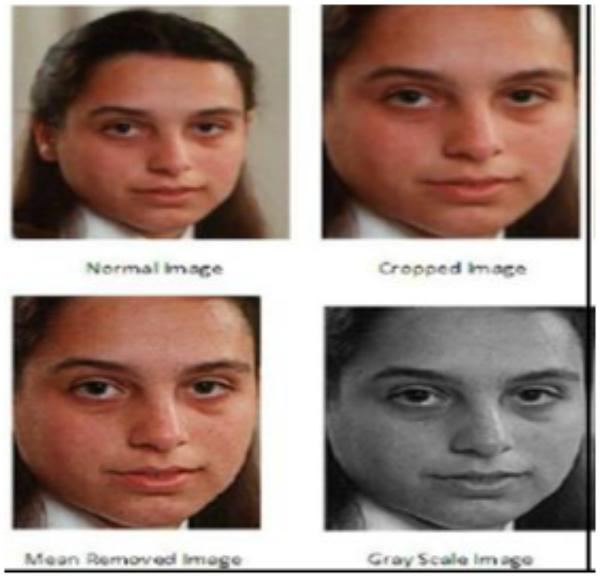

Figure 3.2: Preprocessing

3.5 Face Recognition

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 recognition”.

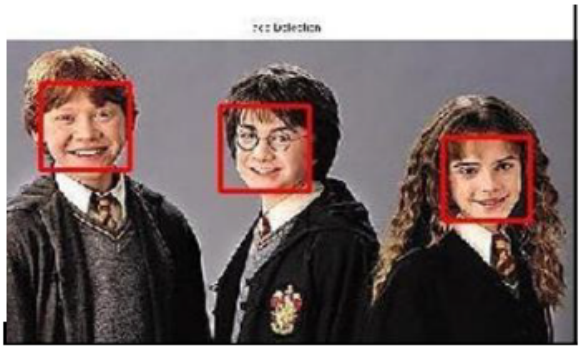

Figure 3.3: Face Detection

3.6 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 area.
a. Lips
b. Eyes
c. Eyebrows
d. Fore head Skin Area

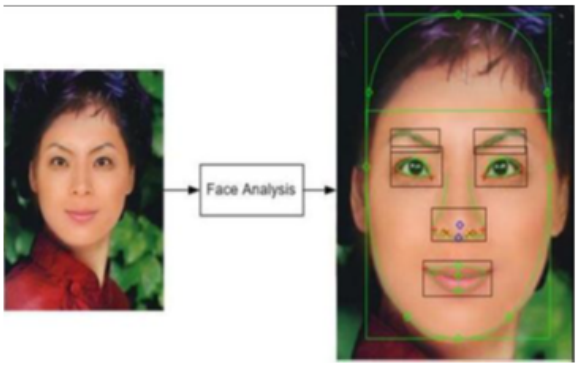

Figure 3.4: Feature Extraction

3.7 Emotion Classification

Basically, the machine classifies the emotion on the face based on different aspects such as eye are, brow area and mouth are. Normally the area changes when the emotion in the image changes.

Figure 3.5: Emotion Classification

3.8 Working of Block Diagram

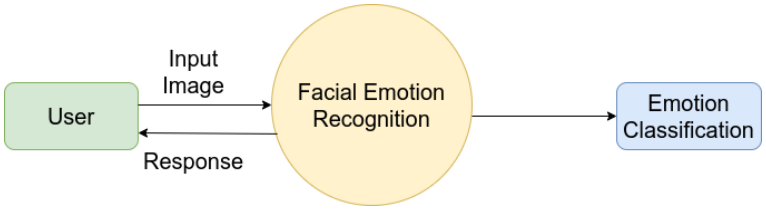

Figure 3.6: Block Diagram Stage 1

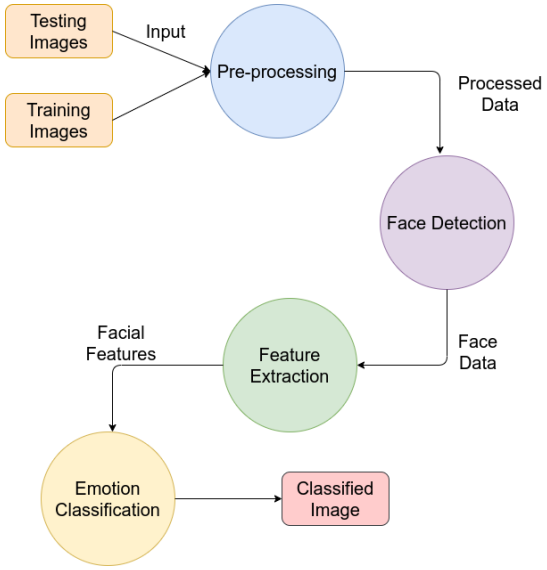

Figure 3.7: Block Diagram Stage 2

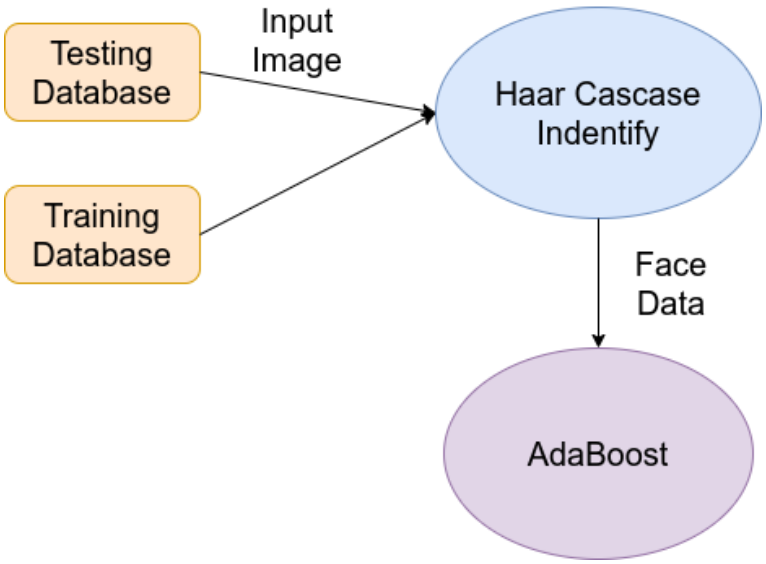

Figure 3.8: Block Diagram Stage 3

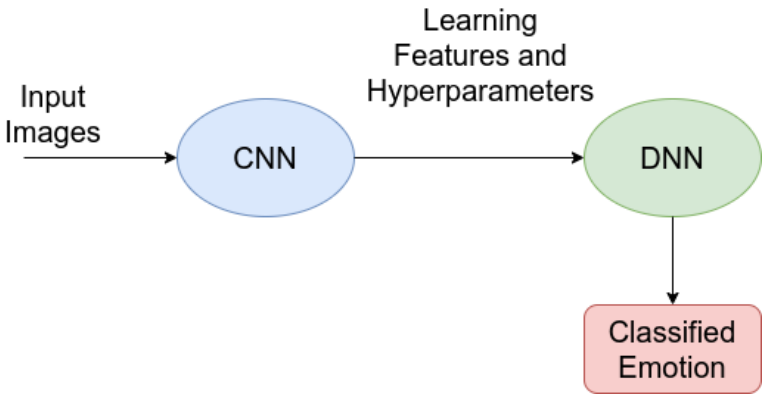

Figure 3.9: Block Diagram Stage 4

3.9 Specifications

3.9.1 Hardware specifications

• Webcam - HP true vision webcam(front facing) with integrated dual array digital microphone.

3.9.2 Software specifications

As the project is developed in Python , we have used Visual Studio Code for Python 3.6.5 and Spyder .

• Visual studio code is an IDE developed by Microsoft for Windows, Linux and MacOS. It includes support for debugging and embedded Git control and Github, syntax highlighting, intelligent code completion , snippets and code refactoring. It is highly customizable and allowing users to change theme ,keyboard shortcut and preferences and install extensions that add additional functionality .

• Spyder (formerly pydee) 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. Spyder is extensible with plugins, includes support for interactive tools for data inspection and embeds python-specific code quality assurance and introspection instruments, such as Pyflakes, Pylint and rope. It is available cross-platform through anaconda, on windows with winpython and python (x,y), on MacOS through macports, and on major linux distributions such as arch Linux, Debian, Fedora, Gentoo Linux, Opensuse And Ubuntu.

3.10 System Design Development

(Criterion for Selection of Components with justification and Complete Circuit Diagram)
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 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.

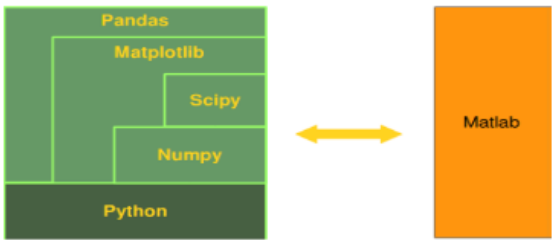

Figure 3.10: Comparison between Matlab and Python

3.11 Selection of Techniques

3.11.1 Face Detection

Facial Detection means that the machine should differentiate between faces and other objects which are present in that given image. Face detection is a technology used by computer systems to detect faces in a given digital image. Automatic face detection is a very complex method in the field of Image Processing and can be done using many techniques/algorithms. The algorithms used for detecting faces in a digital image are Viola-Jones, Cascade Neural Networks (CNN) and Eigen Faces.
For these algorithms to work at its fullest, they normally focus on the frontal faces of humans. Humans can comfortably detect faces on their own but it is not easy for the computers to do so. As there are many combinations in an image, it is very difficult for a computer to detect a face. A human face contains lots of variations such as spectacles, beard, pimples, freckles, hair colour etc. For successfully completing this challenging task, a computer needs to be trained for the same. Here are some algorithms used for face detection:

3.11.2 Viola-Jones algorithm:

Paul Viola and Michael Jones proposed this framework to detect faces. It achieves a high face detection rate of 15 frames per second and is implemented using OpenCV. OpenCV is an open source library for computer vision, image processing and machine learning and now features real time applications. Face detection is achieved using four major steps, they are as follows:

- Selection of a Haar Feature
- Forming an Integral Image
- Forming an Integral Image
- Cascading Classifiers

Haar feature:

Selecting a Haar feature is based on similarities in a human face. There are almost more than 16,000+ features available for detecting a face or other objects. The features are as shown below.

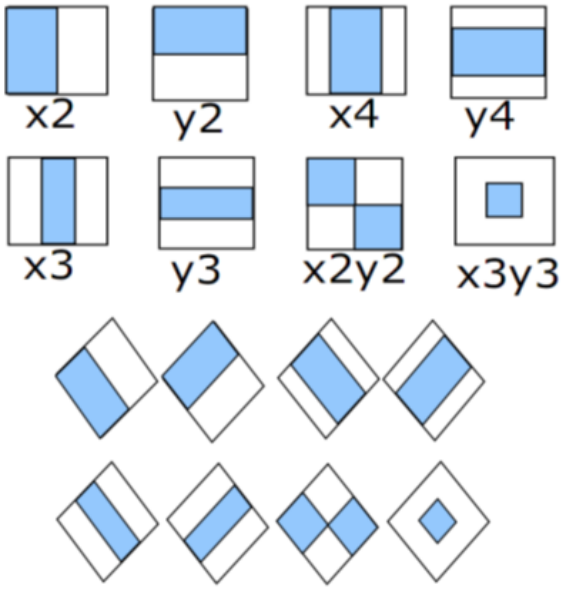

Figure 3.11: Haar Cascade

The sum of the pixels from the white rectangles are subtracted from the sum if the pixels from the dark rectangles. Viola-Jones algorithm uses a 24*24 window. Basically, a Haar feature is something that is common in every face irrespective of all the variations. For example, the eye brow part is darker than the eye part or the vertical nose part is brighter than the two cheeks besides. The eye pupil is usually darker than the surrounding white eye region. So, these may be suitably called as Haar features.
Value of each feature is obtained by subtracting the white region from the dark region and each feature gives a certain single value. Δ of the region is calculated by subtracting the sum of all white pixels from the sum of all dark pixels. The closer the value of Δ towards 1, the greater are the chances of it being a Haar feature. To calculate Δ:

An ideal Haar feature has Δ =1. In a larger image, these calculations tend to be tedious and time consuming, so Viola-Jones came up with the concept of Integral Images which is discussed

$$\Delta = \text{dark} - \text{white} = \frac{1}{n}\sum_{\text{dark}}^{n} I(x) - \frac{1}{n}\sum_{\text{white}}^{n} I(x)$$

Figure 3.12: Formula of Haar

in the next part.

Integral Image:

As discussed above, the concept of integral images was put up to calculate the features rapidly. The integral image at x, y is equal to the sum of all pixels above and left to x and y including x and y.

I, i (x, y) = Σ i (x', y')

Where x' ≤ x and y'≤ y. I, i (x, y) is an Integral image and (x, y) is the source image.

Calculation of an integral image is shown in the figure below.

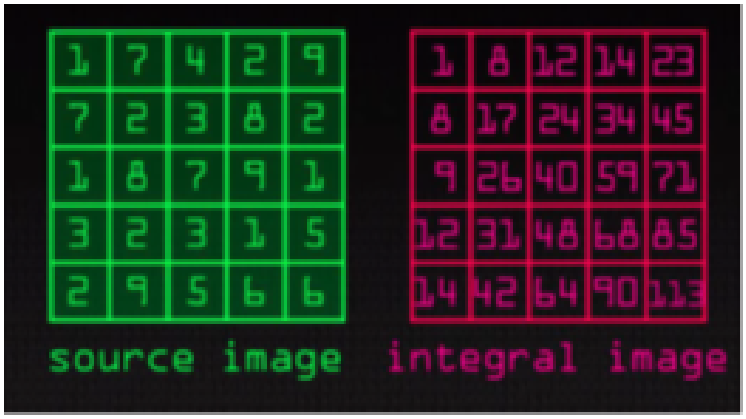

Figure 3.13: Calculation of Integral image

AdaBoost :

AdaBoost is a machine learning algorithm which helps finding the best features among the 160,000+ Haar features. For classifying the image, that is differentiating the face images, a classifier is created using AdaBoost.

AdaBoost selects a small number of critical features from a larger set of classifiers and yield efficient classification. The speed with which features maybe evaluated, does not compensate with the number of features available. For example, 24*24 window having 160,000+ on every image makes this process time consuming and expensive as well. Thus, the object/face detection algorithm uses an efficient machine learning algorithm "AdaBoost" for smartly selecting the appropriate features and also to train the classifier to use them. AdaBoost constructs a strong classifier as a linear combination of weighted simple weak classifiers.

F(x) = a1f1(x) + a2f2(x) + a3f3(x) + a4f4(x) + a5f5(x) + ... F(x) is strong classifier and f(x) is weak classifier. Weak classifier always provides binary value; 0 and 1. If the feature is

present the value of weak feature will be 1, otherwise value will be 0. Thus, the strong classifier is created.

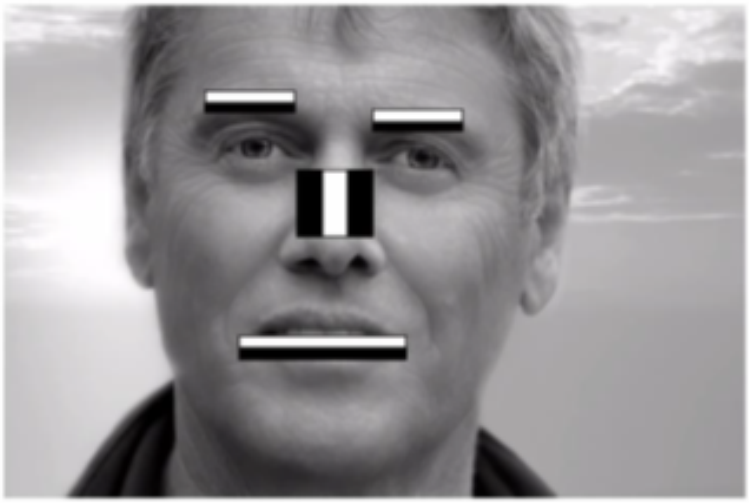

Figure 3.14: Relevance of HAAR features in AdaBoost

Cascading: Assuming we have an image of resolution 640*450, we need to move more than 2,500 windows of 24*24 on this image to detect whether it is a face. Linear checking of each feature can be expensive and also it is not efficient.

But, instead of using a window 2,500 times, we can create a cascade for easier calculations. In this, the first 10 features will be evaluated in the first classifier, the next 20 in the next classifier and so on. Basically, we are making a cascade of classifiers in which we can eliminate some of classifiers based on the results of the previous classifiers.

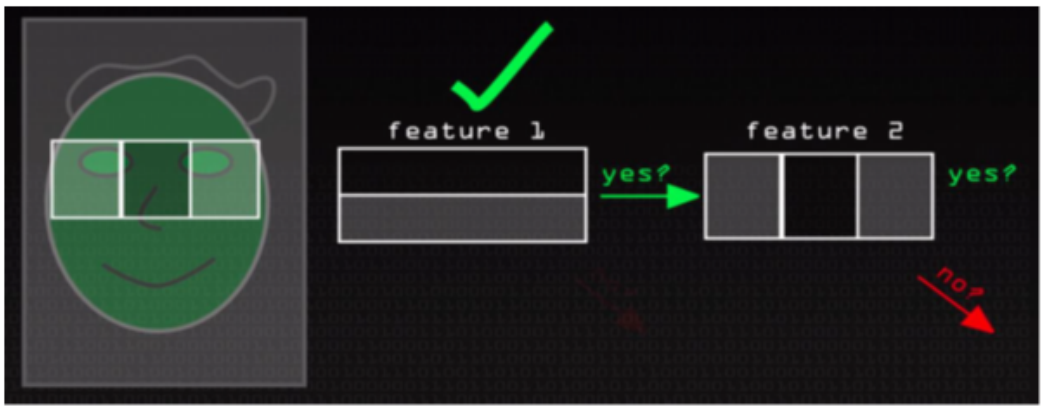

Figure 3.15: Cascade 1

As given above, if the first classifier is passed then it goes to the next stage. But, the first classifier is not passed then the next classifier containing those features will be eliminated.

In this process, increase the complexity of our classification but the efficiency and time is improved. Given below is a cascade for detecting a face using cascade classifier.

Figure 3.16: Cascade 2

3.11.3 The Dataset:

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 7 emotion classes: anger, disgust, fear, happiness, sadness, surprise, and neutral.

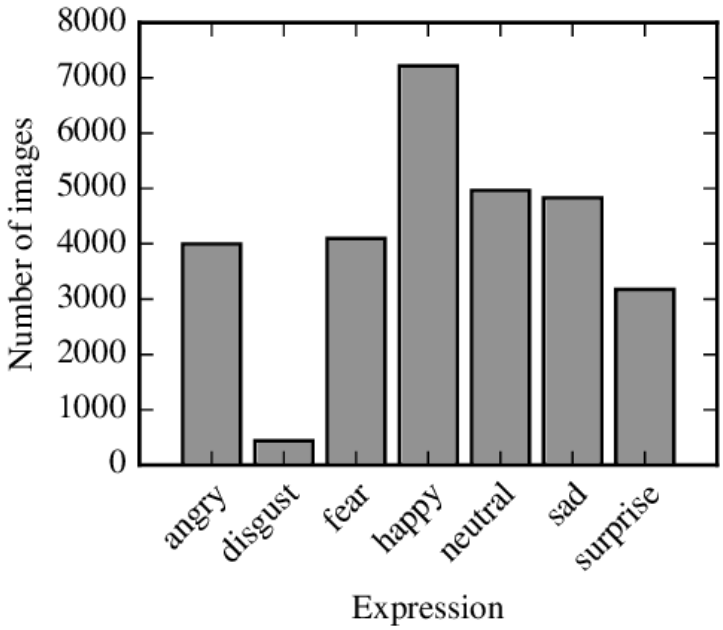

Figure 3.17: The distribution of dataset

3.11.4 Artificial Neural Network:

Artificial neural networks are the best used in machine learning. The "neural" in the network are brain-inspired systems which are intended to learn like human's use "neurons". Neural networks comprise input and output layers, a set of hidden layers comprising nodes or neurons that help in the learning stages from the input into something that the output layer can use. They are magnificent for pattern extraction which is too complex numerically for a programmer to extract and teach the machine to classify or learn.

Neural Networks work very similar to how human body works, so they are normally called as artificial neural networks. Neural networks have been around since 1940's, previously known as "Perceptron" have become a major part of Artificial Intelligence and Machine Learning. The intelligence in these machine learning algorithms have become advanced because of the arrival of the concept called Backpropagation. Backpropagation allows the model to adjust its value in such a way that the user does not expect. That is, when the image is actually a dog but the model mistakenly classifies it as a cat. This concept drastically increases the efficiency of the model to classify. Deep neural network is a kind of network which has more than one hidden layer and each layer is a fully connected layer. That means every node in a layer is connected to every node in the next layer. Using the fully connected network, the layers use different input and output functions and extract features until it can recognize what it wants.

There are different types of neural networks, each of them has its own specific use cases and levels of complexity. The most basic type of neural network is called a feedforward neural network, in which information travels in only one direction from input to output. A more widely used type of network is the recurrent neural network, in which data can flow in multiple directions. These neural networks possess greater learning abilities and are widely employed for more complex tasks such as learning handwriting or language recognition. There are also Convolutional Neural Networks in which, the input is an image or a set of images and then features are extracted using convolution operation and then at the end a fully connected layer is placed. The convolution layers are used for feature extraction and the classification process is done by the fully connected layer.

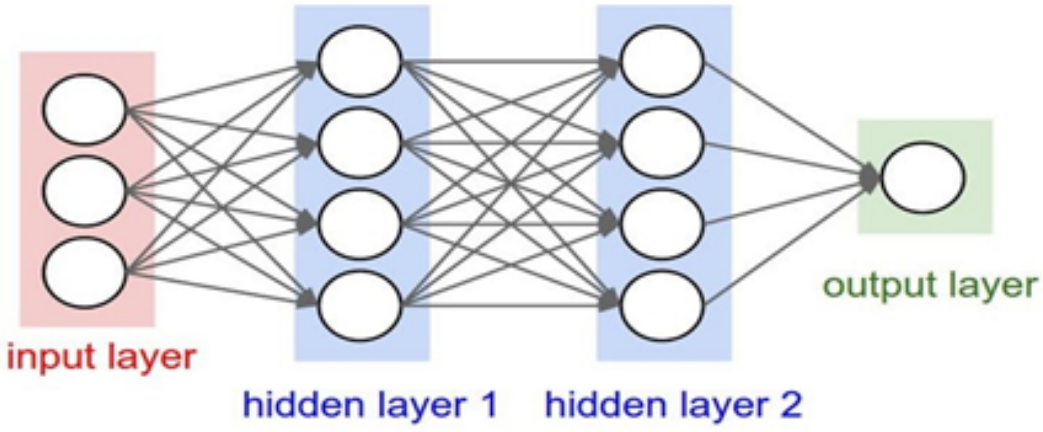

Figure 3.18: Basic architecture of neural network showing hidden layers

3.11.5 Feature Extraction: Convolutional Neural Network (CNN):

A CNN is a type of a neural network which is used for face detection and feature extractions on a given digital image. Basically, a neural network for detecting faces consists of an input layer, the hidden layers and the output layer. The input layer consists of all the values of pixels in the given input image. Suppose we give an input image of resolution 28*28 that is total of 784 pixels, the input layer will have 784 nodes with those pixel values.

Now, the hidden layer will consist a number of nodes which are connected to the previous that is the input layer. Each node in the hidden layer is connected to every node on the input layer with the corresponding weights to it. So, assuming there are two hidden layers in out neural network, the first hidden layer will detect the edges of the face and the second hidden layer will detect the patterns which are present on every face in the input image. The output of that node on the basis of its input or set of inputs is called as its activation function.

As previously mentioned, the input layer will contain all the pixels in the input image, the first hidden layer will have the edges and the second hidden layer will have the patterns on a face. According to inputs or the set of inputs to the particular node, the activation function triggers and the corresponding outputs will be given to the nodes on the next layer. Thus, when the nodes of the final layer are activated, the node having a maximum value will be considered as the prediction of that neural network. For example, if the final layer contains a node which specifies that the face is happy, and has the maximum values compared to other nodes of that layer, the output of that neural network will be a “Happy Face”.

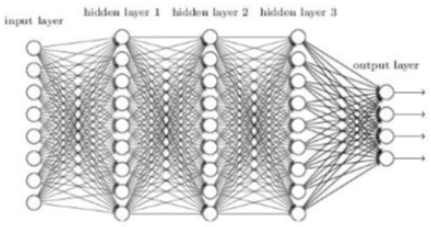

Figure 3.19: Typical Deep Neural Network having multiple classes

It is a type of neural network normally used for determine faces in an image. The basic architecture of a CNN is given below.

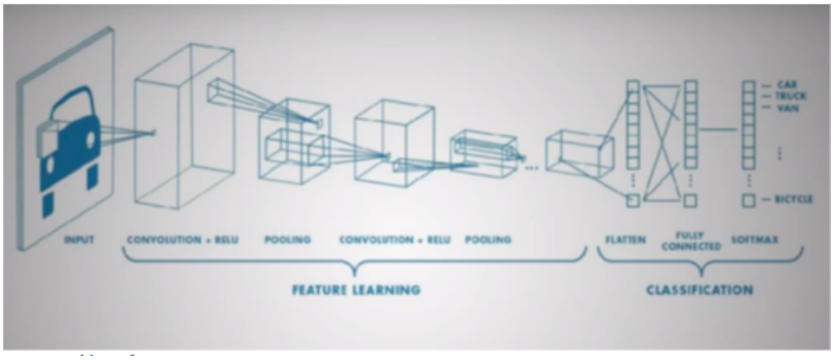

Figure 3.20: Inside the Neural Nets: Feature Extraction and Classification

Why use a Convolutional Neural Network and not a Deep Neural Network?

1. Suppose in a dataset there are 100 images of size 28*28 pixels. The input to a deep neural network will be the values data from each pixel. That is each node will represent every pixel from that images. For example, in this dataset, every image has 28x28 pixels per image. That will be 784 pixels per image and in total 784x100 pixels. For a deep network, it is not practical to have such a high number of input nodes as the hidden layers is going to extract features and will be very complex. So, convolution and max pooling layers are going to extract more significant data in the form of feature maps and the least significant data will be discarded.

2. Convolution extracts features of images and converts it into lower dimensions without losing its characteristics.

3. The features extracted in a CNN are shift invariant. That is if the font, size, location of the desired pattern in the image changes, it does not affect the output.

What are the advantages of using CNN?

- Parameter Sharing: The input layer in the fully connected layer is increased by the image size. During convolution, same pixels are used for training. The concept of dropout is used in these layers to reduce the effect of nodes having odd data. It is also used to overcome the effect of over-fitting.

- Sparsity of Connections: For each layer, each output value depends on a small number of inputs instead of taking into account all the inputs.

What is the difference between CNN and DNN?

CNN is a type of neural network normally used for determine faces in an image. Convolutional Neural Networks are very similar to Deep Neural Network, they are made up of neurons that have learnable weights and biases. Each neuron receives some inputs, performs a dot product or convolution and then they either fire the neurons or they don't. That is the activation of the neurons is dependent on the activation function. The whole network still works as if a differentiating a score function: from the raw image pixels on the inputs to classification scores at the output. The accuracy of the output is enhanced by measuring loss using loss functions. The loss is calculated and it is tried to reduce by using optimizers.

Architecture of the CNN:

As the input to the ConvNets are images, the convolution is the first layer. Followed by convolution, maxpooling is preferred. These two layers come hand in hand, that is after every convolution layer or set of convolution layer, a maxpooling layer is used. In our classification problem, we have used the following architecture.

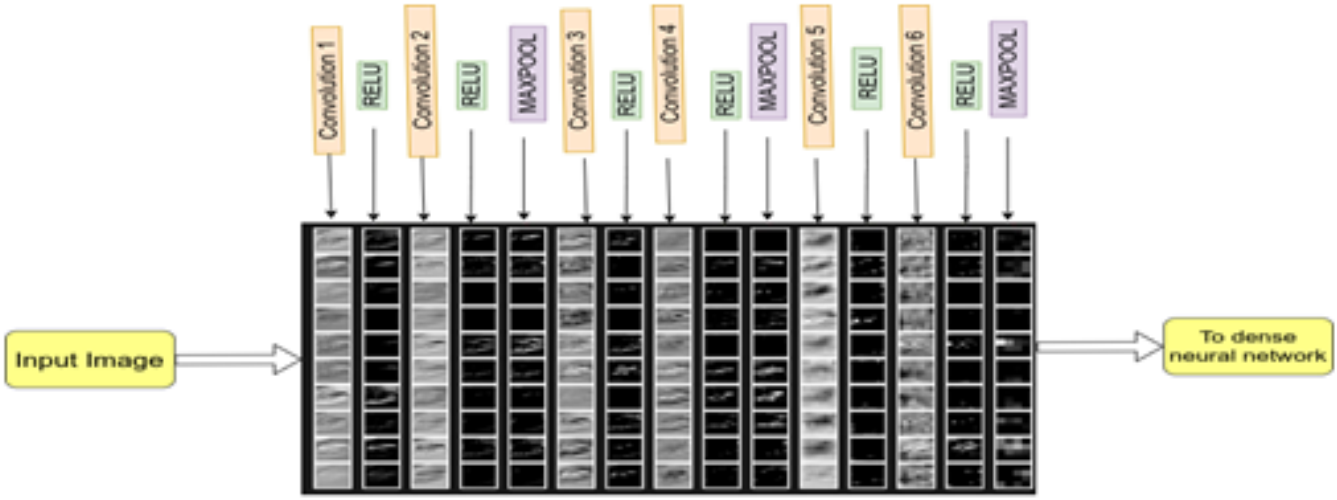

Figure 3.21: Inside the convolution layers: Convolution, Maxpool, RELU

- INPUT [48x48x1] will hold the raw pixel values of the image, in this case an image of width 48, height 48, and grayscale images. Normally they are RGB, but FER2013 has a grayscale set of images.
- CONV layer will compute the output of neurons that are connected to local regions in the input, each computing a dot product between their weights and a small region they are connected to in the input volume. This may result in volume such as [46x46x64] if we decided to use 64 filters. Here the width and height of images is decreased and the number of features is increased. That is previously, there was only one feature, after the convolution operation the features are increased to 64.
- RELU layer will apply an elementwise activation function, such as max (0, x) thresholding at zero. This leaves the size of the volume unchanged ([46x46x64]).
- POOL layer will perform a downsampling operation along the spatial dimensions (width, height), resulting in volume such as [23x23x64].
- FC (i.e. fully-connected) layer will compute the class scores, resulting in volume of size [1x1x7], where each of the 7 numbers correspond to a class score that is every emotion. As with ordinary Neural Networks and as the name implies, each neuron in this layer will be connected to all the numbers in the previous volume.

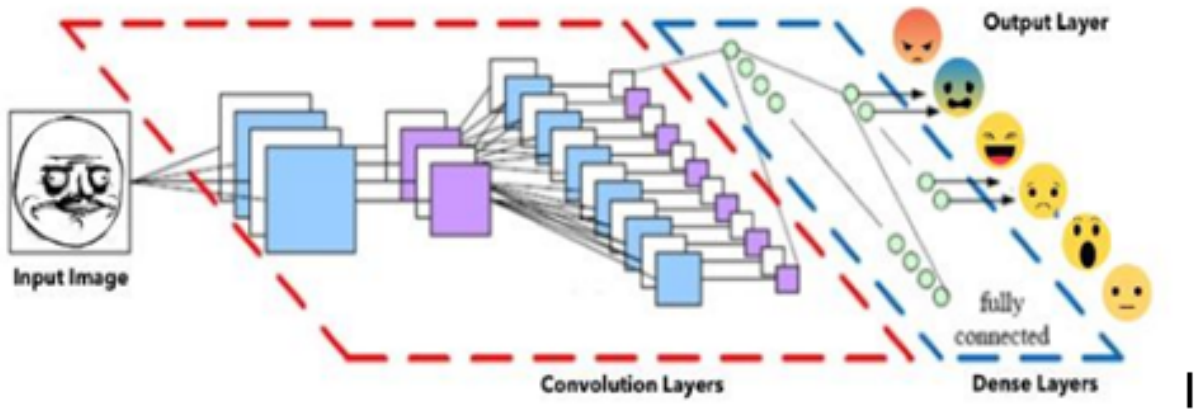

Figure 3.22: Basic idea of data traversal

Modes of Convolution:

- Basic notations:
  Filter length = k
  Input length = n
  In TensorFlow, a signal or a tensor of images is represented by (H, W, C)
  Where, H = height of the image
  W = width of the image
  C = number of features or colourmaps in that image.
  Using this notation, an RGB image is represented by (H, W, 3) and a grayscale image can be represented by (H, W, 1)

- Valid Mode: Filter can never touch the outside of the input image. As per the above notation for filter = kxk and input length = n, the output length = n-k+1

- Full Mode: Filter can go outside the input image. Zero or Pixel padding is used. The filter can go outside the image in such a way that there should be atleast one overlapping element. Output length = n+k-1

- Same Mode: In this mode, the user can set the amount of padding required. The output size and the filter size can be decided by the user by using padding. Suppose p = padding on both sides of the image, the p= (k-1)/2

The Convolution Process: Suppose the input image is HWC1, where C1 is the number of colourmaps in that image. To it a filter of KK2 is convolved, the output observed is,

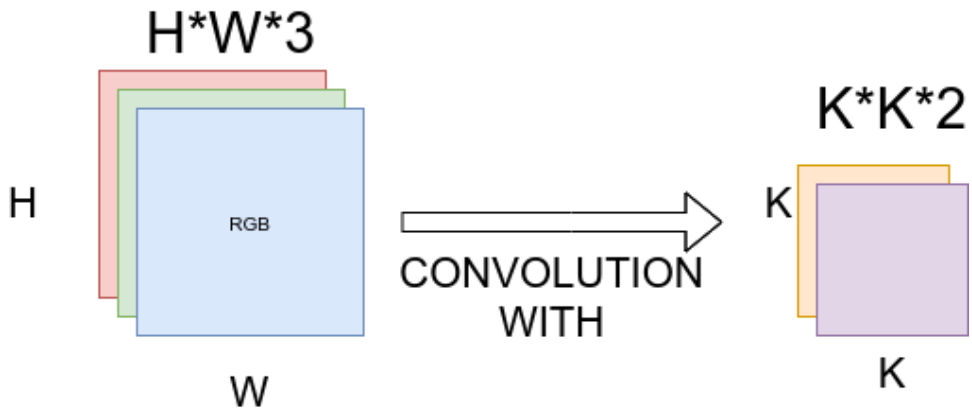

Figure 3.23: Basic Convolution Process

$$OUT(i,j) = \Sigma_{k1=1}{}^{k} \Sigma_{k2=1}{}^{k} \Sigma_{C=1}{}^{C} in[\, i - K1, j - K2, C] \; filter[\, C, K1, K2]$$

Figure 3.24: Formula for Convolution of 3D input image with a 3D filter

Here, we can see that the convolution of a 3D input image and a 3D filter is a 2D image and the depth parameter is reduced.

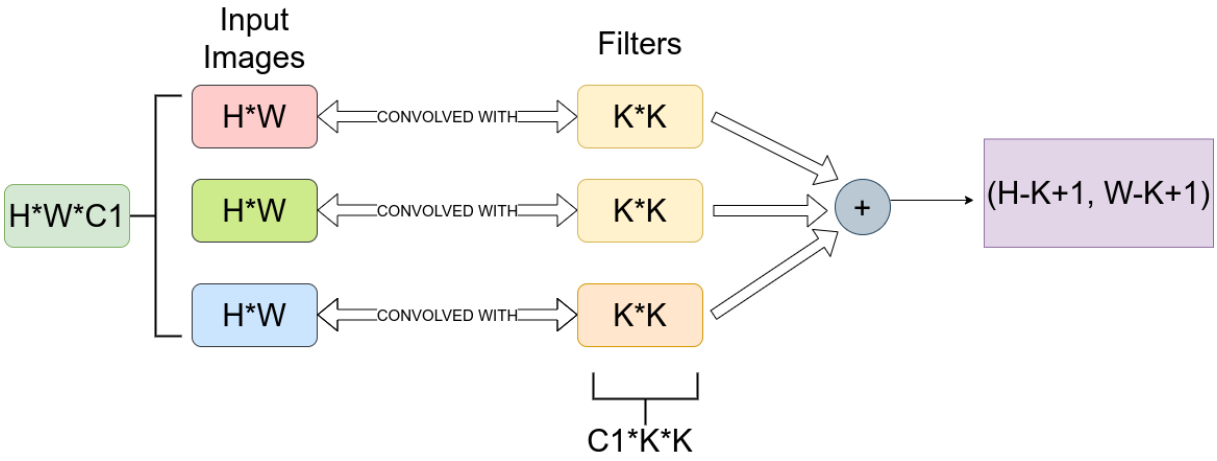

Figure 3.25: Convolution of 3D input image with a 3D filter

In the above figure, input image has RGB colourmaps and thus C1 is three. Suppose filter KK is also repeated three times, then the filter tensor will be (C1, K, K). The output of this convolution in valid mode will be (H-K+1, W-K+1).

In the figure below; the filter KK is repeated and stacked C2 times. Similarly, if (C1, K1, K2) is repeated C2 times, the size of the filter tensor will be (C1, K1, K2, C2).

Therefore, to produce an output of 3 dimension by using an input of 3D, we should change the filter size into 4D tensor as explained above.

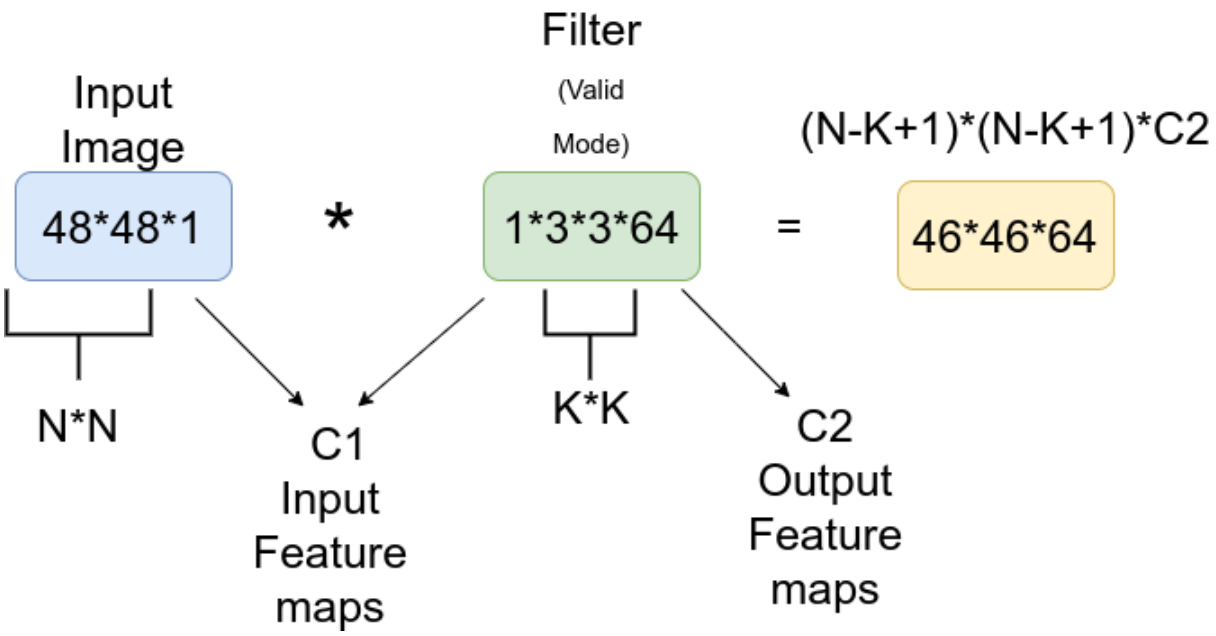

Figure 3.26: Convolution with an appropriate filter to get a 3D output image

- C1 = input feature maps. C1 = 1 for grayscale image.
- C2 = Output feature maps.

  Here C2 is any number of feature maps depending on the features extracted by the filters.
- For example, one feature map from C2 features would be for edge detection, other could be for rotation of angle detection etc.
- These are called as activation features.

The actual output from our model is given below:

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 46, 46, 64)        640
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 46, 46, 64)        36928
_________________________________________________________________
batch_normalization_1 (Batch (None, 46, 46, 64)        256
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 23, 23, 64)        0
_________________________________________________________________
dropout_1 (Dropout)          (None, 23, 23, 64)        0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 23, 23, 128)       73856
_________________________________________________________________
batch_normalization_2 (Batch (None, 23, 23, 128)       512
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 23, 23, 128)       147584
_________________________________________________________________
batch_normalization_3 (Batch (None, 23, 23, 128)       512
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 11, 11, 128)       0
_________________________________________________________________
dropout_2 (Dropout)          (None, 11, 11, 128)       0
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 11, 11, 256)       295168
_________________________________________________________________
batch_normalization_4 (Batch (None, 11, 11, 256)       1024
_________________________________________________________________
conv2d_6 (Conv2D)            (None, 11, 11, 256)       590080
_________________________________________________________________
batch_normalization_5 (Batch (None, 11, 11, 256)       1024
_________________________________________________________________
conv2d_7 (Conv2D)            (None, 11, 11, 256)       590080
_________________________________________________________________
batch_normalization_6 (Batch (None, 11, 11, 256)       1024
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 5, 5, 256)         0
_________________________________________________________________
dropout_3 (Dropout)          (None, 5, 5, 256)         0
_________________________________________________________________
conv2d_8 (Conv2D)            (None, 5, 5, 512)         1180160
_________________________________________________________________
batch_normalization_7 (Batch (None, 5, 5, 512)         2048
_________________________________________________________________
conv2d_9 (Conv2D)            (None, 5, 5, 512)         2359808
_________________________________________________________________
batch_normalization_8 (Batch (None, 5, 5, 512)         2048
_________________________________________________________________
conv2d_10 (Conv2D)           (None, 5, 5, 512)         2359808
_________________________________________________________________
batch_normalization_9 (Batch (None, 5, 5, 512)         2048
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 2, 2, 512)         0
_________________________________________________________________
dropout_4 (Dropout)          (None, 2, 2, 512)         0
_________________________________________________________________
flatten_1 (Flatten)          (None, 2048)              0
_________________________________________________________________
dense_1 (Dense)              (None, 512)               1049088
_________________________________________________________________
dropout_5 (Dropout)          (None, 512)               0
_________________________________________________________________
dense_2 (Dense)              (None, 256)               131328
=================================================================
dropout_4 (Dropout)          (None, 2, 2, 512)         0
_________________________________________________________________
flatten_1 (Flatten)          (None, 2048)              0
_________________________________________________________________
dense_1 (Dense)              (None, 512)               1049088
_________________________________________________________________
dropout_5 (Dropout)          (None, 512)               0
_________________________________________________________________
dense_2 (Dense)              (None, 256)               131328
_________________________________________________________________
dropout_6 (Dropout)          (None, 256)               0
_________________________________________________________________
dense_3 (Dense)              (None, 128)               32896
_________________________________________________________________
dropout_7 (Dropout)          (None, 128)               0
_________________________________________________________________
dense_4 (Dense)              (None, 7)                 903
=================================================================
Total params: 8,858,823
Trainable params: 8,853,575
Non-trainable params: 5,248
_________________________________________________________________
```

Figure 3.27: Layers and their output from our model

Parameter Calculations:

- For ith layer in the convolution layer,

  Number of parameters = [(filter size) (number of features of (i-1)th layer +1] (number of features of ith layer)

  For example, if we consider conv layer 5 that is conv2d-5 then

  Number of features of (i-1)th layer is = 128

  Number of features of ith layer is = 256

  Filter size = 3 3

  Therefore,

  Number of parameters of conv2d-5 = [(33128) + 1] 256 = 295,168
  parameter which match with the output we got.

- For ith layer in the dense layer,

  Number of parameters = [(number of features of (i-1)th layer (number of features of ith layer)] + number of features of ith layer as bias

  Let's consider, dense-2 layer,

  Number of features of (i-1)th layer is = 512

  Number of features of ith layer is = 256

  Number of parameters of dense-2 layer = [512 256] + 256 = 131,328
  which matches with the output we got.

Pooling layer:

Pooling layer is commonly inserted in-between successive Convolution layers in a Convolutional Neural Net architecture. Its function is to progressively reduce the spatial size of the conv layer to reduce the number of parameters and computation in the net, and hence reduces the possibility of overfitting.
The most common pooling layer is having filters of size 2x2 applied with a stride of 2 down samples every featuremap in the input by 2 along both width and height, discarding around 75% of the activations. Every Maxpool operation would be taking a max of 4 numbers that is little 2x2 region in some kernel.
The depth dimension remains unchanged. We don't pool features; only spatial features are pooled to reduce computation and overfitting. A pooling layer operates on each feature maps separately to create a new set of same number of pooled feature maps. Pooling involves selection of a pooling operations like a filter applied on a feature map. The size of the pooling operation is smaller than the size of filter maps. That is the filter of pooling is smaller than the filter in the feature maps. Pooling layer always reduces the size of each feature map by the factor of 2. There are mainly two types of pooling; Average pooling and Maximum pooling. In all cases, the pooling helps to make the representation become approximately invariant to small translations of the input. Invariance to translation means that if we translate the input by small amount, the values of most of the pooled outputs do not change. The basic working of maxpool is shown below:

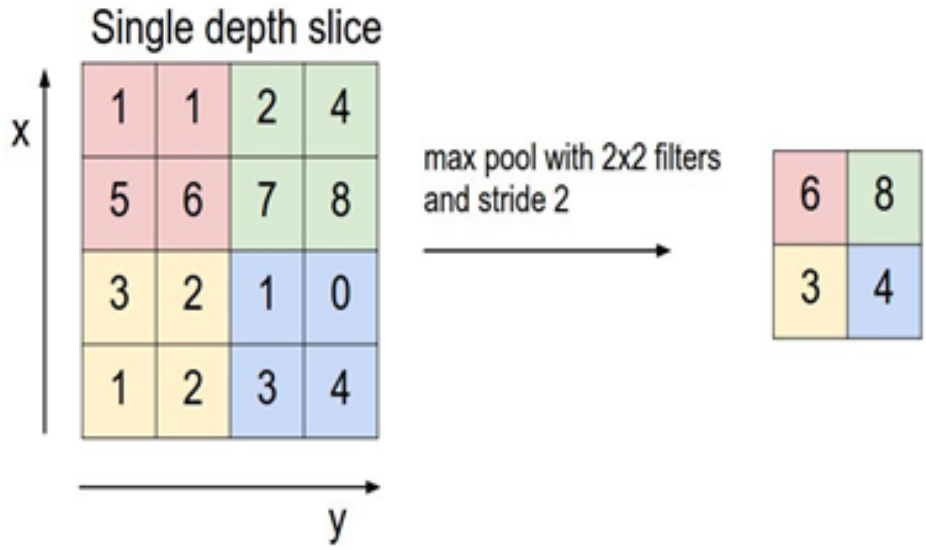

Figure 3.28: Maxpooling operation

• When we perform Maxpooling in succession with convolution, two major effects are seen:

a. Size of the image shrinks.

b. Feature maps increases.

• Advantages of maxpooling is:

a. Reduces size, thus reduces the possibility of overfitting.
b. Speeds up computation.
c. Features are made robust.

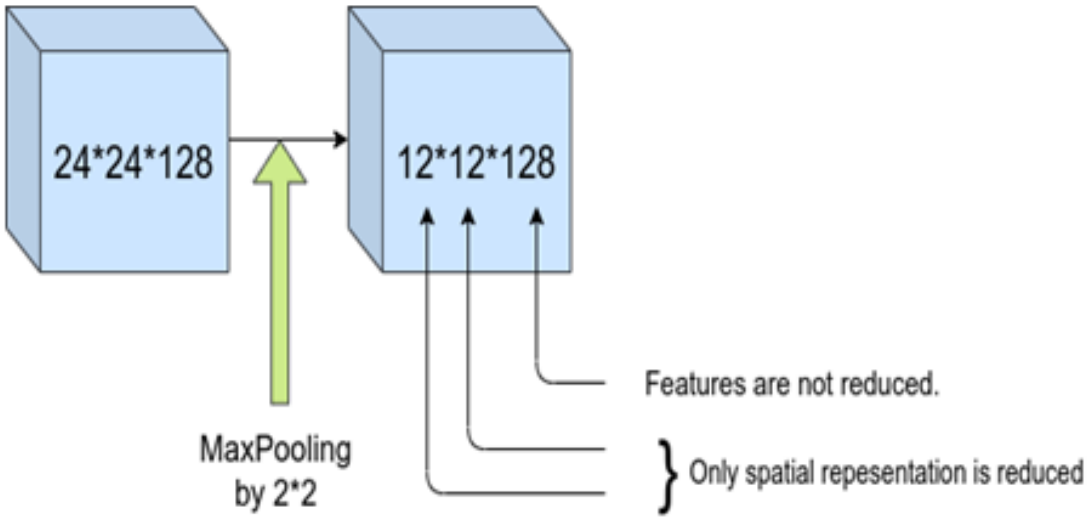

Figure 3.29: Effect of maxpooling

Strides:
a. It is a concept to hop over some required pixels in row/ column of every image. In our project, we have decided stride as 2.
b. This means that while performing convolution operation or maxpooling operation, the filter is going to skip one pixel and hop over it to reduce the computation.
c. Using strides, the model becomes faster and more efficient. No extra storage is required and can be used to design the correct filter size.
d. Suppose S = Strides, and as referring to the earlier mentioned notation, if NxN image is convolved with fxf filter using S = 2 and Padding P:
(N x N) * (f x f) = [((N+2p-f)/s + 1), ((N+2p-f)/s + 1)]

• The bigger is the filter size, more global information is captured whereas smaller the filter size more local information is captured. Average pooling is also used in some cases. Avegare pooling calculates the average of the slide and shows output as the mean of pixels. In average pooling, the pixels having more data is suppressed and only some number of pixels are given out. Therefore, average and minimum pooling is not preferred. The diagram below shows how the conv layer and maxpooling layer go hand in hand.

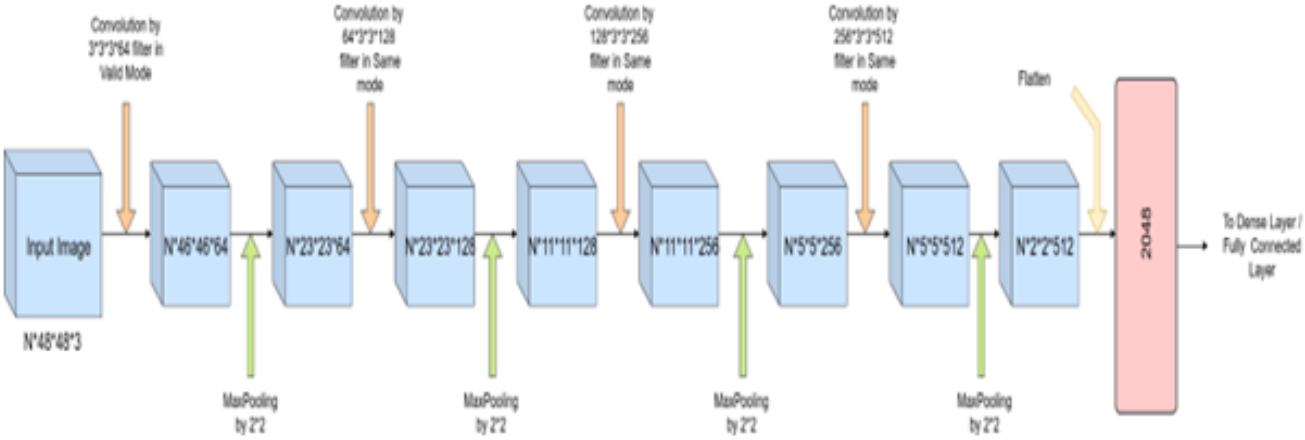

Figure 3.30: Convolution and Pooling layer succession

Dropout and Batch Normalization: We normalize the input layer by adjusting and scaling the activations. For example, when we have features from 0 to 1 and some from 1 to 1000, we should normalize them to speed up learning. If the input layer is benefiting from it, why not do the same thing also for the values in the hidden layers, that are changing all the time, and get 10 times or more improvement in the training speed.
To increase the stability of a neural network, batch normalization normalizes the output of a previous activation layer by subtracting the batch mean and dividing by the batch standard deviation. Dropout is a concept where the output of a random number of neurons is dropped and not considered. It is a technique of regularising and reducing the computation and overfitting.

Activation Functions:
Activation functions are given for every layer to fire the output of each neuron. Basically, every neuron calculated the weighted sum of the inputs and adds bias to it.

$$Y = \sum (weight * input) + bias$$

Figure 3.31: Output of every node: Weighted sum of weights and biases

Y here is the output of every neuron which can lie in the range of + ∞ to -∞ . If the range of output is so high, there is a problem in decided whether the output should be fired or not. To overcome this problem, the activation functions are decided for each layer. There are many activation functions such as tanH, Step function, Sigmoid, ReLu,etc.

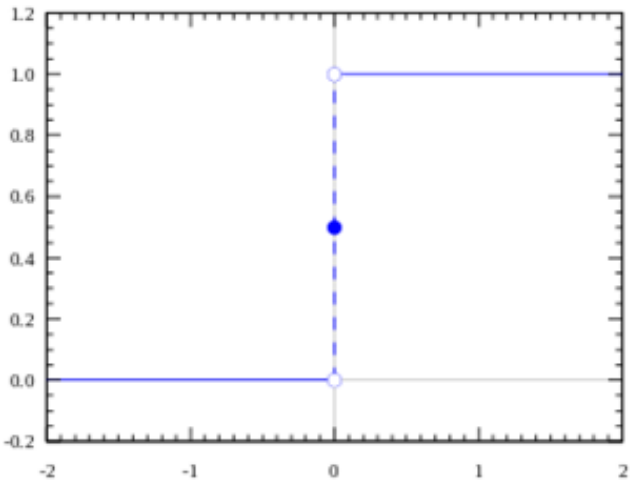

Figure 3.32: Activation Function 1: Step Function

Activation function A = “activated” if Y<threshold else not
Alternatively, A = 1 if Y>threshold, 0 otherwise.

Following is the equation of sigmoid function: Y = 1 / (1+ e-x)

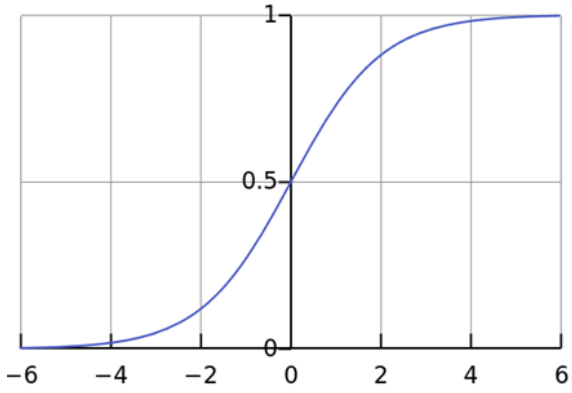

Figure 3.33: Activation Function 2: Sigmoid Function

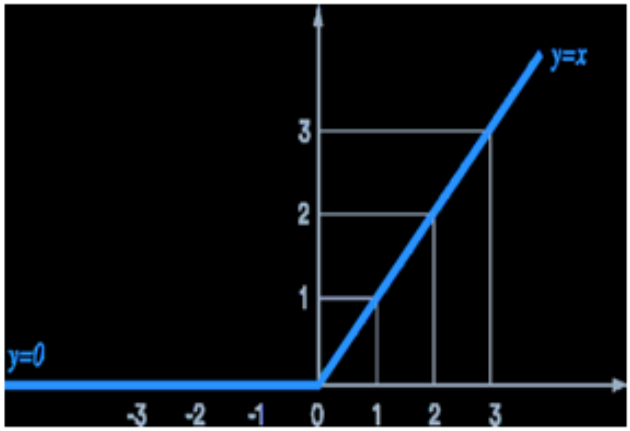

Figure 3.34: Activation Function 3: RELU Function

Following is the equation for ReLu:
A(x) = max (0, x)
The relu function is given by:
$y = 0 for x <= 0;$
$= x for x > 0.$

The reason we are applying this activation function is to increase the linearity of the objects. The purpose behind it is that most of the images are naturally non-linear.

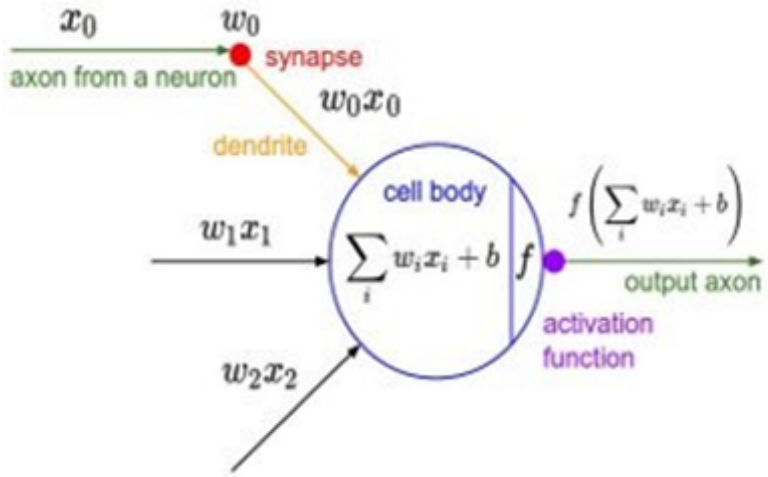

Figure 3.35: Anatomy of a neuron

Above diagram shows the anatomy of the neuron and how the output of each neuron is triggered according to the weighted sum of the inputs and the biases.

Fully Connected Layer:
Fully Connected Layer is simply, feed forward neural networks. Fully Connected Layers form the last few layers in the network. The input to the fully connected layer is the output from the final Pooling or Convolutional Layer, which is flattened and then fed into the fully connected layer.

The output from the Pooling and Convolutional Layer is a 3-dimensional matrix, to flatten that is to unroll all its values into a vector. The fully connected or the dense neural network is used for classification similar to where convolution layers were used for feature extraction. This connects every node in one layer to every other node in another layer. There are two main types of traversal techniques in the dense neural network which helps in the training and classification.

1. Forward Propagation:
The process of traversal of data from the first layer to the last layer sequentially through the activation functions is called as forward propagation. The whole process of weight initialization, weighted sum, firing of neurons, extraction of features through the traversal comes under forward propagation.

Weight initialization is a very important concept in forward propagation. The weights and biases are the hyperparameters of the neural network and are constantly updating every iteration. Every node is connected to the nodes in the next layer using weights. These weights decide the activation or deactivation of the node. To enhance and normalize these outputs, the activations functions are used. Now the weights and biases are randomly allocated. These weights maybe assigned from (0 to 1). They keep on updating as the number of iterations increase. If the weights are initialized randomly, a problem of exploding and vanishing gradient is observed.

To overcome this problem, a new technique of glorot initialization is used. It calculates the number of neurons in the layer (width) and divides the randomly assigned weight by the width. This ensures that the weight initialization is robust and less error is observed. This glorot initialization also reduces the variance problem, that is if the randomly assigned weights have mean at 0 and high variance, then the output is affected and training requires more iterations and time. The glorot initialization reduces this variance by dividing with the width.

2. Back Propagation: It allows the data traversal in the backward direction that is from output layer to the input layer. While propagation in the backward direction, it calculated gradient. Gradient is calculated by the derivative of loss (cost) function with respect to the weights times the learning rate.

Learning rate is the number of steps required to reach the minimum loss point from the starting point. It is assigned between 0.01 to 0.0001. If the learning rate is too high then the minimum loss point is not achieved and may loop over the required point whereas if the learning rate is too small then it requires more time to reach the minimum loss point.

To reach the minimum loss point, we must calculate the total losses. Loss function is used for the same. Categorical cross entropy is used as loss function as there is classification problem and the loss function output are measured in one hot encoded vector. Optimizers are used to enhance the loss calculation and reduce the loss of that model at a given iteration. Optimizers like Stochastic Gradient descend (SGD), Adam, Adagrad, Adamax are used to optimize the model by every epoch by updating the weights and reducing the losses. Loss is calculated by
Loss = Actual output – Ideal output

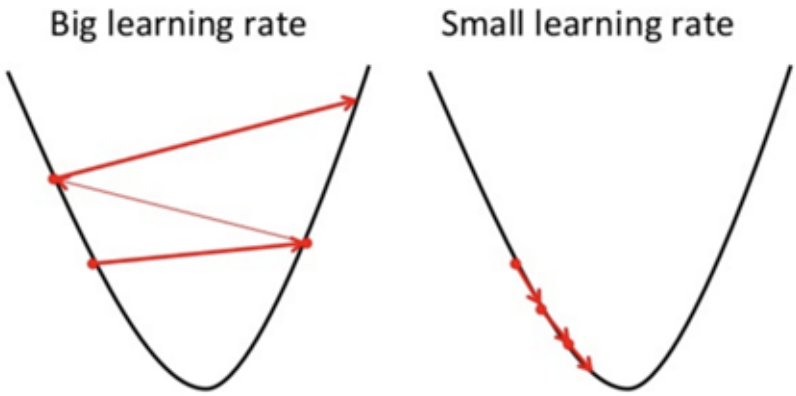

Figure 3.36: Effect of learning rate on the gradient descend

Ideal output is the value of that class which is already labelled. During the training process, for optimal updatation of hyperparameters gradient descend is used. It makes use of loss function. In the output layer, the loss is calculated and then it is differentiated with respect to the weights. To calculate the loss of the output layer, activation output of the previous layer is required. To find the activation output of the previous layer, the input to that layer should be known.

C0 = Total loss
α1= activation output of the node 1 of layer L
Z1 = input to node 1 from layer L
W12 = weight joining from 2nd node in L-1 layer to 1st node in layer L
Using the chain rule and partially differentiating the loss, we can find the gradient descend. At each epoch, for every node this gradient descend is calculated and updated till a point where the model gets minimum output.

To find the input, the weights joining its previous layer to that layer should be known. That is to find out the gradient of any class, weights, input and activation output of the previous layer should be known. This is mathematically expressed by the following equation:

$$\frac{\partial C_0}{\partial w_{12}^{(L)}} = \left(\frac{\partial C_0}{\partial a_1^{(L)}}\right)\left(\frac{\partial a_1^{(L)}}{\partial z_1^{(L)}}\right)\left(\frac{\partial z_1^{(L)}}{\partial w_{12}^{(L)}}\right).$$

Figure 3.37: Partial derivative of loss function by applying chain rule

3.11.6 Eigen Faces (Principal Component Analysis):

It is also called dimensionality reduction technique. The curse of dimensionality is eliminated in this technique. It is used for exploratory data analysis and for making predictive models. Consider that we are classifying on the basis of two features namely skin colour and area of nose. The searching area is bounded between this two-dimensional graph. But what will happen if we go on increasing the number of features? Yes, the adding of features will help in better classification. But, when we increase the number of features, the time required for calculations increase and more area is to be searched or scanned for finding these eigen faces/values. This will make sure that more time is required to train the machine.

The curse of dimensionality tells about a specific problem called as Overfitting. It is defined when we have more features than the dataset, the classifier tends to generalize the dataset on the basis of these features. This is effectively called as the "The Curse of dimensionality".

In this problem, the classifier becomes overconfident on the basis of the dataset. That is in the process of supervised learning, the machine tends to remember specific values and their outputs, and then it generalizes because we have given more features and the dataset available is less. So, the principle of PCA tells us that- "there exist a lower dimensional subspace which can still faithfully represent your object."

Basically, PCA can supply the user with a lower dimensional picture, "a shadow", we can say of that object when viewed from its most informative viewpoint. PCA is a mathematical procedure that uses orthogonal transformation to convert a set of correlated M variables into a set of values of K uncorrelated variables called as principal components/Eigen faces. Where K$\leq$ M ; Eigen faces $\leq$ Face Images (Original Input dataset).
The transformation is defined in such a way that first principle components or eigen faces shows the most dominant directions or features of dataset and each succeeding component in turn shows the next possible dominant feature of direction.

(Succeeding is uncorrelated to the preceding component). The succeeding components depict less features and more noise so, only first few eigen faces are selected. (say K). These K components almost safely represent a whole original dataset. In other words, each eigen face is contributing some feature to the data. Every image contains a little bit of features of any number of these K eigen faces.

To find the principal components:

- Find the covariance matrix.
  cov (x, y) = Σ (xi – x') (yi – y')/ (n+1) from i =1 to n.
- To find eigen values and eigen vectors.

$$\text{Cov}(A) = \begin{bmatrix} \frac{\sum(x_i - \bar{X})(x_i - \bar{X})}{N} & \frac{\sum(x_i - \bar{X})(y_i - \bar{Y})}{N} \\ \frac{\sum(x_i - \bar{X})(y_i - \bar{Y})}{N} & \frac{\sum(y_i - \bar{Y})(y_i - \bar{Y})}{N} \end{bmatrix}$$

$$= \begin{bmatrix} \text{Cov}(X,X) & \text{Cov}(Y,X) \\ \text{Cov}(X,Y) & \text{Cov}(X,Y) \end{bmatrix}$$

Figure 3.38: Covariance Matrix

- Find the determinant and find the eigen values and eigen vectors.

$$\det(\Sigma - \lambda I) = 0$$

Figure 3.39: Using Deteminant find the unknown values

- The greater the eigen vector, it is the principal component.

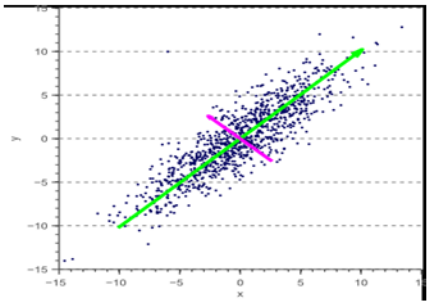

Figure 3.40: Principal Components

Above given are the principal components of the given dataset. The first principal component is placed in such a way that it is at the shortest distance from each of the points in the dataset and the second principal component is orthogonal to the first one.

3.11.7 Support Vector Machine (SVM):

In machine learning, support vector machines are supervised learning models with associated learning algorithms that analyse 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. Basically, the new examples are classified to categories and the categories are divided by a hyper- plane.

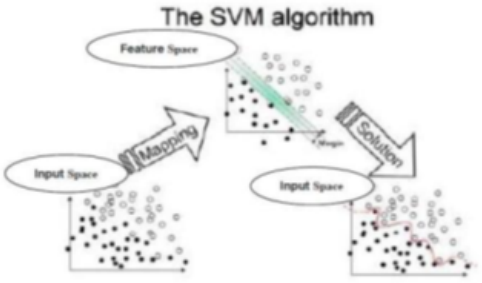

Figure 3.41: Working of SVM

ADVANTAGES: There are many folds advantages of using the supervised learning approach of Support Vector Machine (SVM). They are very effective when we have very high dimensional spaces. Also, when number of dimensions becomes greater than the existing number of samples, in such cases too SVM is found to be very effective. SVM uses a subset of training point also known as support vectors to classify different objects hence it is memory efficient. Support Vector Machines are versatile, for different decision function we can define different kernel as long as they provide correct result. Depending upon our requirement and application we can choose types of kernel which is most productive for our application.

DISADVANTAGES: The disadvantage of SVM is that if the number of features is much greater than the number of samples, the method is likely to give poor performances. SVM gives efficient result for small training samples as compared to large ones. SVMs do not directly provide probability estimates, so these must be calculated using indirect techniques. Also, we can have Non-traditional data like strings and trees as input to SVM instead of featured vectors. Lastly selecting appropriate kernel for the project is a big issue which depends upon user's requirement.

Chapter 4

RESULTS, SIMULATION AND ANALYSIS:

4.0.1 OUTPUTS:

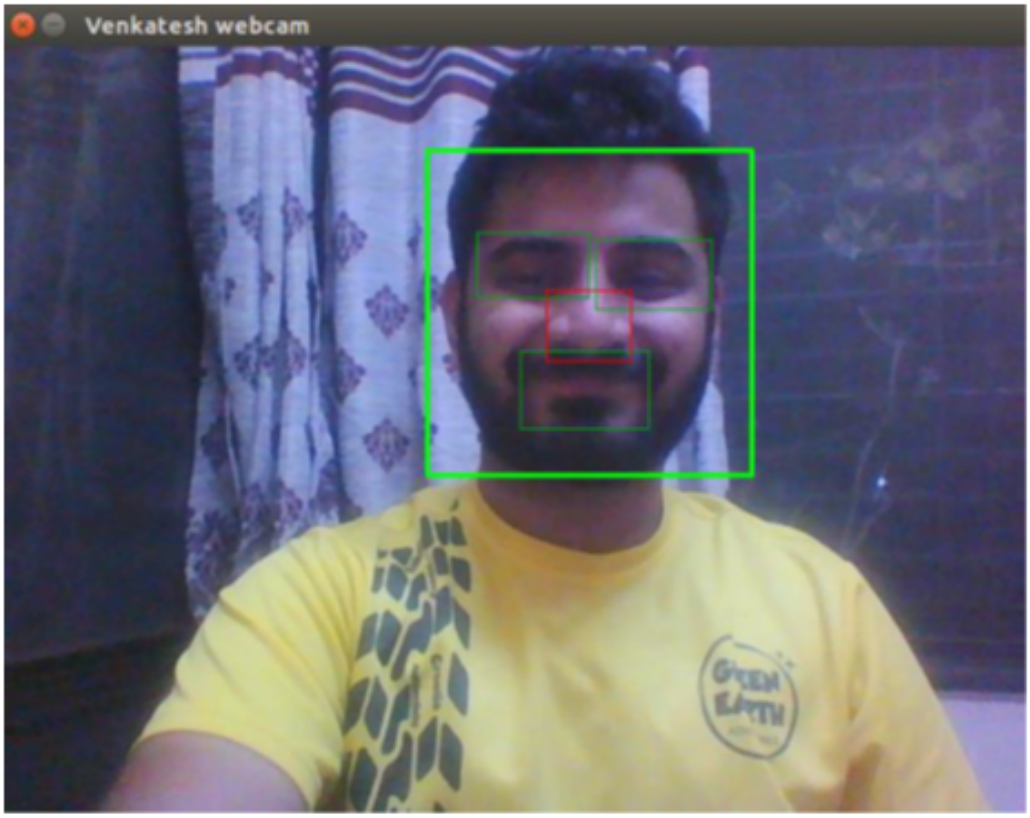

Figure 4.1: Face Detection

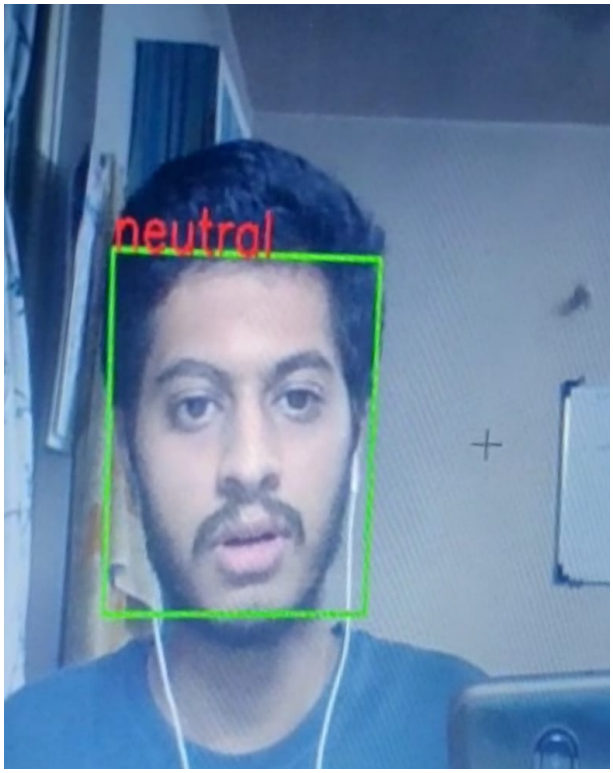

Figure 4.2: Neutral Face

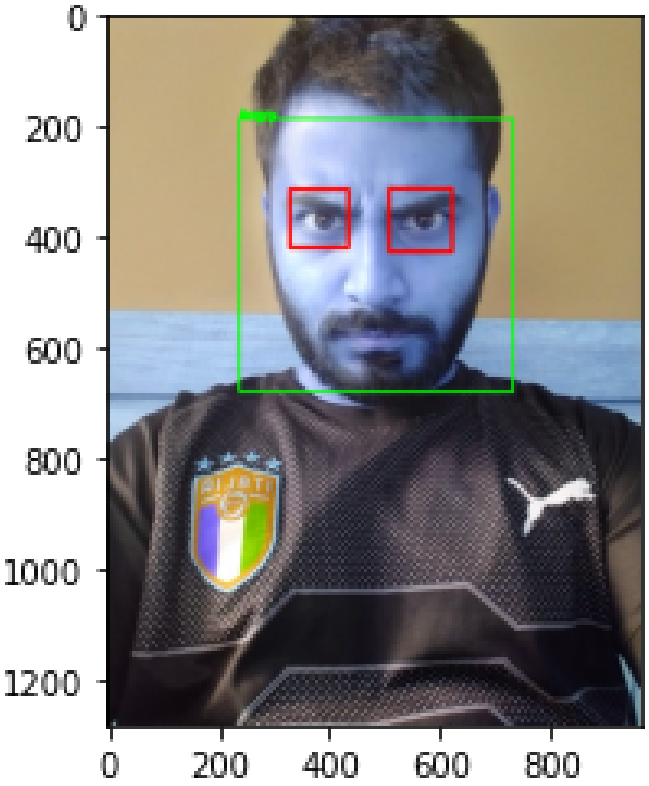

Figure 4.3: Angry Face

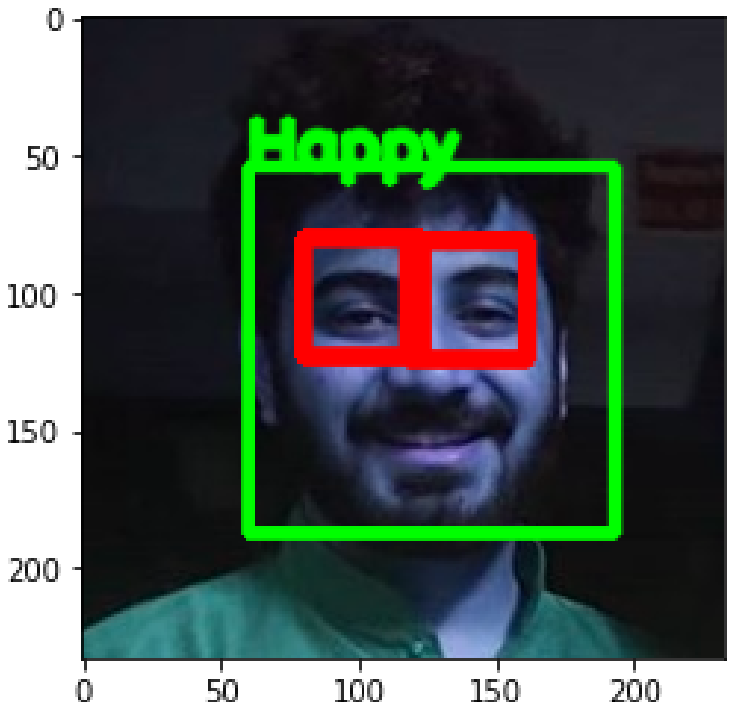

Figure 4.4: Happy Face

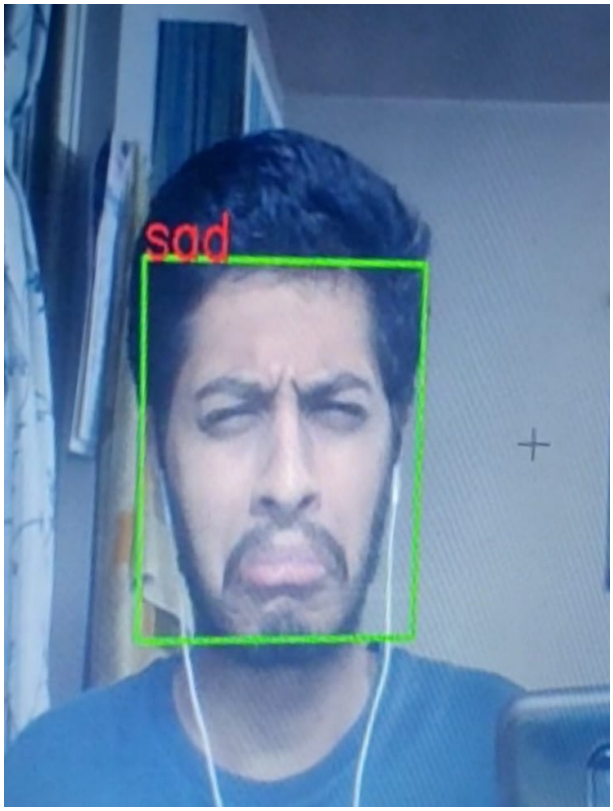

Figure 4.5: Sad Face

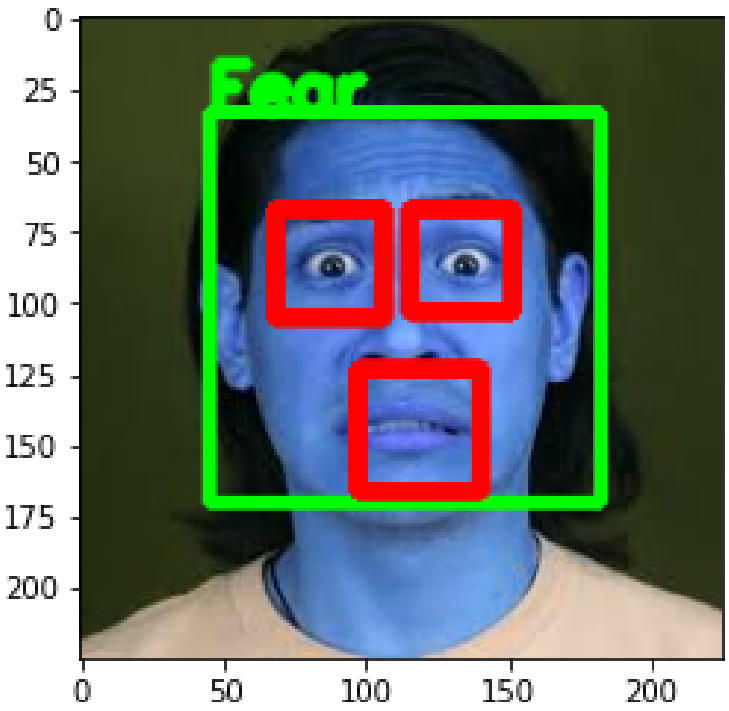

Figure 4.6: Scared Face

4.0.2 Plagiarism Report


SmallSEOTools

PLAGIARISM SCAN REPORT

Words	304	Date	June 09,2020
Characters	1875	Exclude Url	https://1drv.ms/w/s!ArVdJyqN52VugVs4GI110bZ-RQDs

0% Plagiarism	100% Unique	0 Plagiarized Sentences	18 Unique Sentences

Content Checked For Plagiarism

Figure 4.7: Plagiarism Report

Chapter 5

REFERENCES:

[1] Paul Viola and Michael Jones. "Rapid Object Detection using a Boosted Cascade of Simple Features". Accepted conference on computer vision and pattern recognition 2001

[2] Yoav Freund and Robert E. Schapire. "A decision-theoretic generalization of on-line learning and an application to boosting". In Computational Learning Theory: Eurocolt '95, pages 23–37. Springer-Verlag, 1995.

[[3] Facial Expression Recognition with Convolutional Neural Networks. Shekhar Singh Computer Science University of Nevada Las Vegas Las Vegas, USA. Fatma Nasoz Computer Science University of Nevada Las Vegas Las Vegas, USA

[4] Face Expression Recognition Based on Convolutional Neural Network* Lei Xu, Minrui Fei, Wenju Zhou, Aolei Yang 2018 Australian New Zealand Control Conference (ANZCC) Swinburne University of Technology Melbourne, Australia, Dec 7-8, 2018

[5] Jianzhu guo, zhen lei , (senior member, ieee), jun wan , (member, ieee), egils avots, (student member, ieee), noushin hajarolasvadi, (student member, ieee), boris knyazev, artem kuharenko, julio c. silveira jacques junior, xavier baró , hasan demirel, sergio escalera , jüri allik, and gholamreza anbarjafari , (senior member, ieee). "Dominant and Complementary Emotion Recognition From Still Images of Faces"

[6] Anushree Basu, Aurobinda Routray, Suprosanna Shit, Alok Kanti Deb." Human Emotion Recognition from Facial Thermal Image based on Fused Statistical Feature and Multi-Class SVM". IEEE INDICON 2015 1570200155.

[7] Ming-Hsuan Yang, Narendra Ahuja, David Kraegman. "Face recognition using kernel eigenfaces". 2000 IEEE

[8] Irene Kotsia and Ioannis Pitas, Senior Member, IEEE. "Facial Expression Recognition in Image Sequences Using Geometric Deformation Features and Support Vector Machines. IEEE transactions on image processing", vol. 16, no. 1, january 2007.

[9] Kwang In Kim, Keechul Jung, and Hang Joon Kim. “Face Recognition Using Kernel Principal Component Analysis”. IEEE Signal Processing Letters, Vol. 9, No. 2, February 2002.

[10] Jia Xiang, Gengming Zhu. “Joint Face detection and Facial Expression Recognition with MTCNN”. 2017 4th International Conference on Information Science and Control Engineering.

[11] Rohit Pathar, Abhishek Adivarekar, Arti Mishra, Anushree Deshmukh Human Emotion Recognition using Convolutional Neural Network in Real Time

5.1 APPENDIX

Project Plan

August 2019:

- 1st Week - Search on various topics of project
- 2nd Week-Research papers related to topic
- 4th Week-Presentation of first evaluation

September 2019:

- Research and study of various algorithms

October 2019:

- 1st week-Collection of image Database
- 2nd week-Study of OpenCV and preprocessing
- 3rd week-Face detection
- 4th week-Second evaluation

December 2019:

- Chose Convolutional Neural Network

January 2020:

- Started Working on CNN
- Preprocessing of Dataset

February 2020:

- Feature Extraction

April 2020:

- Completed CNN model architecture

May 2020:

- Testing of model on an image
- Real time testing

Work Distribution

Sr.No.	Work	Done by
1	Decision and search of various topics	done by all three of us
2	Research papers	Done by all three of us
3	Study of numpy and pandas	Atharva Dastane and Viraj Sanap
4	Face detection in python	Viraj Sanap
5	Study of Viola Jones for Face detection	Venkatesh Mullur
6	Report writing	Atharva Dastane and Viraj Sanap
7	Execution of Viola Jones technique for face detection	Venkatesh Mullur
8	Decision of CNN	All three of us decided
9	Study of CNN by courses	Atharva Dastane and Venkatesh Mullur
10	Understanding of CNN	All three of us
11	Study of JSON	Venkatesh Mullur
12	Study of Google authentication for Google colab and Study of colab	Venkatesh Mullur
13	Study of Tensorflow and Keras	All three of us
14	Study of Deep Learning features	viraj Sanap and Atharva Dastane
15	Writing code for CNN	Venkatesh Mullur
16	Finding the database	Venkatesh Mullur
17	Report and Presentation	Atharva Dastane and Viraj Sanap
18	Report in Latex	Atharva Dastane
19	Updating the report in Latex	Venkatesh Mullur and Atharva Dastane
20	Training and Testing of the model in the code	Venkatesh Mullur
21	Real Time Classification using webcam	viraj Sanap
22	Completion of report and Project	Venkatesh Mullur and Atharva Dastane