A Project Report on

FACIAL EMOTION RECOGNITION USING DEEP LEARNING

Submitted in partial fulfillment of the requirements for the award of degree of

BACHELOR OF TECHNOLOGY

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

J.N.T. UNIVERSITY, KAKINADA

Submitted by

U. SAIRAM (19F01A04N6)

K. VEERA RAJU (19PC1A0414) **V. REVANTH (19F01A04O7)**

S. SRINIVAS (19F01A04L8)

Under the Esteemed Guidance of

Dr. R.V.S. HARISH, M. Tech, Ph.D.

Professor

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

ST. ANN'S COLLEGE OF ENGINEERING & TECHNOLOGY: CHIRALA

(Autonomous)

(Approved by AICTE New Delhi, Permanently Affiliated to JNTUK, Kakinada)

(Accredited by NAAC, NBA & IE (I) with 'A' Grade)

2019-2023

ST.ANN'S COLLEGE OF ENGINEERING & TECHNOLOGY, CHIRALA

(Autonomous)

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

(Approved by AICTE, New Delhi & Affiliated to JNTUK, Kakinada)
(Accredited by NBA, IE(I) & NAAC with 'A' Grade)

CERTIFICATE

This is to certify that the project report entitled **"FACIAL EMOTION RECOGNITION USING DEEP LEARNING"** is submitted by U.Sairam (19F01A04N6), K.Veeraraju (19PC1A0414), V.Revanth (19F01A04O7) and S.Srinivas (19F01A04L8) **who carried out under my guidance and supervision, as a partial fulfillment for** the award of **BACHELOR OF TECHNOLOGY in ELECTRONICS AND COMMUNICATION ENGINEERING of J.N.T.U., Kakinada during the academic year 2022-2023**.

Viva Voce Date: ___________.

Internal Guide	Head of the Department
(Dr. R.V.S. HARISH)	**(Dr. K. JAGADEESH BABU)**

External Examiner

ACKNOWLEDGEMENT

The successful completion of any task is not possible without proper suggestion, guidance, and environment. The combination of these three factors acts like backbone to our project entitled "**FACIAL EMOTION RECOGNITION USING DEEP LEARNING**".

We express our sincere thanks to our guide **Dr. R.V.S. HARISH, M. Tech., Ph.D., MIEEE** and Head of Department of Electronics and Communication Engineering, **Dr. K. JAGADEESH BABU, M. Tech., Ph.D., SMIEEE**. for timely help, guidance and providing us with the most essential resources required for the completion of this work. Their invaluable moral support and guidance throughout the project helped us to a greater extent. We are thankful to his valuable suggestions and discussions during the project.

We would like to thank our principal, **Dr. M. VENU GOPALA RAO, MTech., Ph.D.** for providing support and stimulating environment. We would like to express our gratitude to the management of **St. Ann's College of Engineering & Technology** for providing us with pleasant environment and excellent laboratory facilities.

We would be thankful to all teaching and non-teaching staff of the Department of Electronics and Communication Engineering for the cooperation given for the successful completion of our project.

Last but not least, we express our heart full thanks to all our friends for all the help and cooperation extended in bringing out this project successfully in time.

PROJECT ASSOCIATES

U. SAIRAM	(19F01A04N6)
K. VEERARAJU	(19PC1A0414)
V.REVANTH	(19F01A04O7)
S. SRINIVAS	(19F01A04L8)

DECLARATION

This is to declare that the thesis titled "**FACIAL EMOTION RECOGNITION USING DEEP LEARNING**" is a bonafide work done by me, in partial fulfilment of the requirements for the award of the degree of **Bachelor of Technology** and submitted to the Department of Electronic & Communication Engineering, **St. Ann's College of Engineering & Technology, Chirala.**

I also declare that this project is a result of my own effort and that has not been copied from anyone and I have taken only citation from the source which are mentioned in the references.

This work was not submitted earlier at any other university or institute for the award of any degree.

Place: SACET, Chirala **U. SAIRAM**

Date: **19F01A04N6**

CONTENTS

List of Figures

List of Tables

ABSTRACT

Automatic facial expression analysis is an interesting and challenging problem and impacts important applications in many areas such as human–computer interaction and data-driven animation. Deriving an effective facial representation from original face images is a vital step for successful facial expression recognition. In recent years, facial emotion recognition (FER) has become a prevalent research topic as it can be applied in various areas. The existing FER approaches include handcrafted feature-based methods (HCF) and deep learning methods (DL). HCF methods rely on how good the manual feature extractor can perform. The manually extracted features may be exposed to bias as it depends on the researcher's prior knowledge of the domain. In contrast, DL methods, especially Convolutional Neural Network (CNN), are good at performing image classification.

The downfall of DL methods is that they require extensive data to train and perform recognition efficiently. Hence, we propose a deep learning method based on transfer learning of pre-trained AlexNet architecture for FER. We perform full model finetuning on the Alexnet, which was previously trained on the Imagenet dataset, using emotion datasets. The proposed model is trained and tested on widely used facial expression datasets, namely extended Taiwanese Facial Expression Image Database (TFEID). The proposed framework outperforms the existing state-of-the-art methods in facial emotion recognition by achieving the accuracy of 98% for the Taiwanese Facial Expression Image Database (TFEID) dataset.

Keywords: Facial emotion recognition, Convolution neural network, Alexnet, Taiwanese Facial Expression Image Database.

Chapter-1
Introduction

Facial expression is one of the most powerful, natural, and immediate means for human beings to communicate their emotions and intensions. Automatic facial expression analysis is an interesting and challenging problem and impacts important applications in many areas such as human–computer interaction and data-driven animation. Due to its wide range of applications, automatic facial expression recognition has attracted much attention in recent years. Though much progress has been made, recognizing facial expressions with high accuracy remains difficult due to the subtlety, complexity, and variability of facial expressions.

Deriving an effective facial representation from original face images is a vital step for successful facial expression recognition. There are two common approaches to extract facial features: geometric feature-based methods and appearance-based methods. Geometric features present the shape and locations of facial components, which are extracted to form a feature vector that represents the face geometry. Recently Valstar et al. have demonstrated that geometric feature-based methods provide similar or better performance than appearance-based approaches in Action Unit recognition. However, the geometric feature-based methods usually require accurate and reliable facial feature detection and tracking, which is difficult to accommodate in many situations. With appearance-based methods, image filters, such as Gabor wavelets, are applied to either the whole-face or specific face-regions to extract the appearance changes of the face. Due to their superior performance, the major works on appearance-based methods have focused on using Gabor-wavelet representations. However, it is both time and memory intensive to convolve face images with a bank of Gabor filters to extract multi-scale and multi-orientational coefficients.

Emotion is a mental state associated with the nervous system associated with feeling, perceptions, behavioral reactions, and a degree of gratification or displeasure. One of the current applications of artificial intelligence (AI) using neural networks is the recognition of faces in images and videos for various applications. Most techniques process visual data and search for general pattern present in human faces in images or videos. Face detection can be used for surveillance purposes by law enforcers as well as in crowd management. In this paper, we present a method for identifying five emotions such as anger, disgust, happy, neutral, and sadness using facial images. Previous research used deep-learning technology to create models of facial expressions based on emotions to identify emotions. The typical human computer interaction (HCI) lacks users emotional state and loses a great deal of information during the process of interaction. Comparatively, users are more efficient and desired by emotion sensitive HCI systems. Now a days, interest in emotional computing has increased with the increasing demands of leisure, commerce, physical and psychological well-being, and education related applications. Because of this, several products of emotionally sensitive HCI systems have been developed over the past several years, although the ultimate solution for this research field has not been suggested.

1.1. Digital Image Processing

Digital Image Processing (DIP) is a software which is used to manipulate the digital image by use of computer system. It is also used to enhance the images, to get some important information from it. Digital Image Processing is a software which is used in image processing. For example: computer graphics, signals, photography, camera mechanism, pixels, etc. Digital Image Processing provides a platform to perform various operations like image enhancing, processing of analogy and digital signals, image signals, voice signals etc. It provides images in different formats.

Signal processing is a discipline in electrical engineering and in mathematics that deals with analysis and processing of analog and digital signals, and deals with storing, filtering, and other operations on signals. These signals include transmission signals, sound or voice signals, image signals, and other signals etc.

Out of all these signals, the field that deals with the type of signals for which the input is an image, and the output is also an image is done in image processing. As its name suggests, it deals with the processing of images.

It can be further divided into analog image processing and digital image processing.

1.1.1 Analog image processing

Analog image processing is done on analog signals. It includes processing on two dimensional analog signals. In this type of processing, the images are manipulated by electrical means by varying the electrical signal. The common example include is the television image.

Digital image processing has dominated over analog image processing with the passage of time due its wider range of applications.

1.1.2 Digital image processing

The digital image processing deals with developing a digital system that performs operations on a digital image.

1.2. Phases Of Digital Image Processing

1.2.1 Image Acquisition:

In image processing, it is defined as the action of retrieving an image from some source, usually a hardware-based source for processing. It is the first step in the workflow sequence because, without an image, no processing is possible. The image that is acquired is completely unprocessed. In image acquisition using pre-processing such scaling is done.

1.2.2 Image Enhancement:

It is the process of adjusting digital images so that the results are more suitable for display or further image analysis. Usually in includes sharpening of images, brightness & contrast adjustment, removal of noise, etc. In image enhancement, we generally try to modify the image, so as to make it more pleasing to the eyes. It is subjective in nature as for example some people like high saturation images and some people like natural colour. That's why it is subjective in nature as it differs from person to person.

1.2.3 Image Restoration:

It is the process of recovering an image that has been degraded by some knowledge of degraded function H and the additive noise term. Unlike image enhancement, image restoration is completely objective in nature.

1.2.4 Color Image Processing:

This part handles the image processing of colored images either as indexed images or RGB images.

1.2.5 Wavelets and multiresolution processing:

• Wavelets are small waves of limited duration which are used to calculate wavelet transform which provides time-frequency information.

• Wavelets lead to multiresolution processing in which images are represented in various degrees of resolution

1.2.6 Compression:

Compression deals with the techniques for reducing the storage space required to save an image or the bandwidth required to transmit it. This is particularly useful for displaying images on the internet as if the size of the image is large, then it uses more bandwidth to display the image from the server.

1.2.7 Morphological Processing:

It deals with extracting image components that are useful in representation and description of shape. It includes basic morphological operations like erosion and dilation. As seen from the block diagram above that the outputs of morphological processing generally are image attributes.

1.2.8 Segmentation:

It is the process of partitioning a digital image into multiple segments. It is generally used to locate objects and boundaries in objects.

1.2.9 Representation and Description:

Representation deals with converting data into a suitable form for computer processing.

- Boundary representation: it is used when the focus is on external shape characteristics.
  e.g., corners
- regional representation: it is used when the focus is on internal properties e.g., texture.
- Description deals with extracting attributes that.
  o results in some quantitative information of interest
  o is used for differentiating one class of objects from others

1.2.10 Recognition:

It is the process that assigns a label (e.g., Face) to an object based on its description.

Image processing mainly include the following steps:

1.Importing the image via image acquisition tools;

2. Analyzing and manipulating the image;

3. Output in which result can be altered image or a report which is based on analyzing that image.

1.3. What is an image?

An image is defined as a two-dimensional function, f (X, Y) where x and y are spatial coordinates, and the amplitude of at any pair of coordinates (x,y) is called **intensity** the of that image at that point. When x, y, and amplitude values of are finite, we call it a digital image. In other words, an image can be defined by a two-dimensional array specifically arranged in rows and columns. Digital Image is composed of a finite number of elements, each of which elements have a particular value at a particular location. These elements are referred to as picture elements, image elements, and pixels. A Pixel is most widely used to denote the elements of a Digital Image.

1.4. Types of an image

1. **BINARY IMAGE**– The binary image as its name suggests contains only two-pixel elements i.e. 0 & 1, where 0 refers to black and 1 refers to white. This image is also known as Monochrome.
2. **BLACK AND WHITE IMAGE**– The image which consist of only black and white color is called BLACK AND WHITE IMAGE.
3. **8-bit COLOR FORMAT**– It is the most famous image format. It has 256 different shades of colors in it and is commonly known as Grayscale Image. In this format, 0 stands for Black, and 255 stands for white, and 127 stands for gray.
4. **16-bit COLOR FORMAT**– It is a color image format. It has 65,536 different colors in it. It is also known as High Color Format. In this format the distribution of color is not as same as Grayscale image.

A 16-bit format is divided into three further formats which are Red, Green, and Blue. That famous RGB format.

Image as a Matrix

As we know, images are represented in rows and columns we have the following syntax in which images are represented:

$$f = \begin{bmatrix} f(1,1) & f(1,2) & \cdots & f(1,N) \\ f(2,1) & f(2,2) & \cdots & f(2,N) \\ \vdots & \vdots & & \vdots \\ f(M,1) & f(M,2) & \cdots & f(M,N) \end{bmatrix}$$

The right side of this equation is digital image. Every element of this matrix is called image element, picture element, or pixel.

1.5. Digital Image Representation in MATLAB:

In MATLAB the start index is from 1 instead of 0. Therefore, f(1,1) = f(0,0).

$$f(x,y) = \begin{bmatrix} f(0,0) & f(0,1) & f(0,2) & \ldots & f(0,N-1) \\ f(1,0) & f(1,1) & f(1,2) & \ldots & f(1,N-1) \\ . & . & . & & . \\ . & . & . & & . \\ . & . & . & & . \\ f(M-1,0) & f(M-1,1) & f(M-1,2) & \ldots & f(M-1,N-1) \end{bmatrix}$$

henceforth the two representations of image are identical, except for the shift in origin.

In MATLAB, matrices are stored in a variable i.e X,x,input_image , and so on. The variables must be a letter as same as other programming languages.

OVERLAPPING FIELDS WITH IMAGE PROCESSING

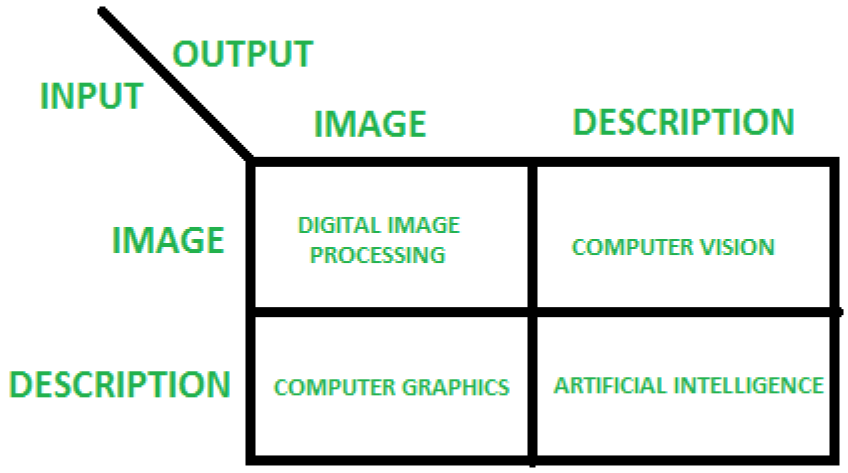


Figure1.1: Overlapping Fields with Image Processing

According to block 1, if input is an image and we get out image as a output, then it is termed as Digital Image Processing.

According to block 2, if input is an image and we get some kind of information or description as a output, then it is termed as Computer Vision.

According to block 3, if input is some description or code and we get image as an output, then it is termed as Computer Graphics.

According to block 4, if input is description or some keywords or some code and we get description or some keywords as a output, then it is termed as Artificial Intelligence

1.6. Advantages of Digital Image Processing:

- Improved image quality: Digital image processing algorithms can improve the visual quality of images, making them clearer, sharper, and more informative.
- Automated image-based tasks: Digital image processing can automate many image-based tasks, such as object recognition, pattern detection, and measurement.
- Increased efficiency: Digital image processing algorithms can process images much faster than humans, making it possible to analyze large amounts of data in a short amount of time.
- Increased accuracy: Digital image processing algorithms can provide more accurate results than humans, especially for tasks that require precise measurements or quantitative analysis.

1.7. Disadvantages of Digital Image Processing:

- High computational cost: Some digital image processing algorithms are computationally intensive and require significant computational resources.
- Limited interpretability: Some digital image processing algorithms may produce results that are difficult for humans to interpret, especially for complex or sophisticated algorithms.
- Dependence on quality of input: The quality of the output of digital image processing algorithms is highly dependent on the quality of the input images. Poor quality input images can result in poor quality output.
- Limitations of algorithms: Digital image processing algorithms have limitations, such as the difficulty of recognizing objects in cluttered or poorly lit scenes, or the inability to recognize objects with significant deformations or occlusions.
- Dependence on good training data: The performance of many digital image processing algorithms is dependent on the quality of the training data used to develop the algorithms.

1.8. Applications of Digital Image Processing

- **Image sharpening and restoration:** The common applications of Image sharpening and restoration are zooming, blurring, sharpening, grayscale conversion, edges detecting, Image recognition, and Image retrieval, etc.
- **Medical field:** The common applications of medical field are Gamma-ray imaging, PET scan, X-Ray Imaging, Medical CT, UV imaging, etc.
- **Remote sensing:** It is the process of scanning the earth by the use of satellite and acknowledges all activities of space.
- **Machine/Robot vision:** It works on the vision of robots so that they can see things, identify them, etc

Chapter - 2
Deep learning

Deep learning is a subset of machine learning, which is essentially a neural network with three or more layers. These neural networks attempt to simulate the behavior of the human brain—albeit far from matching its ability—allowing it to “learn” from large amounts of data.

A group of machine learning techniques known as "deep learning" use numerous layers to gradually extract higher-level characteristics from the input's raw data. In image processing, for instance, lower layers could recognize borders, while higher layers might identify things that are important to people, like numbers, letters, or faces. From a different perspective, deep learning is the process of "computer-simulating" or "automating" human learning processes from a source to an item that has been learnt. As a result, the idea of "deeper" or "deepest" learning makes sense. The most in-depth learning occurs when an item is learned entirely automatically from a source. Hence, a deeper learning refers to a blended learning process that involves both human and computer learning: learning from a source to a learned semi-object, then learning from the learned semi-object to the final learned object.

2.1 Introduction to Deep Learning

Deep learning is a branch of machine learning which is completely based on artificial neural network, as neural network is going to mimic the human brain so deep learning is also a kind of mimic of human brain. In deep learning, we don’t need to explicitly program everything. The concept of deep learning is not new. It has been around for a couple of years now. It’s on hype nowadays because earlier we did not have that much processing power and a lot of data. As in the last 20 years, the processing power increases exponentially, deep learning and machine learning came in the picture. A formal definition of deep learning is- neurons.

Deep Learning is a subset of Machine Learning that is based on artificial neural networks (ANNs) with multiple layers, also known as deep neural networks (DNNs). These neural networks are inspired by the structure and function of the human brain, and they are designed to learn from large amounts of data in an unsupervised or semi-supervised manner.

Deep Learning models can automatically learn features from the data, which makes them well-suited for tasks such as image recognition, speech recognition, and natural language processing. The most widely used architectures in deep learning are feedforward neural networks, convolutional neural networks (CNNs), and recurrent neural networks (RNNs).

2.2 Feedforward neural networks:

(FNNs) are the simplest type of ANN, with a linear flow of information through the network. FNNs have been widely used for tasks such as image classification, speech recognition, and natural language processing.

Feedforward Neural Networks, also known as Deep feedforward Networks or Multi-layer Perceptron's, are the focus of this article. For example, Convolutional and Recurrent Neural Networks (which are used extensively in computer vision applications) are based on these networks. We'll do our best to grasp the key ideas in an engaging and hands-on manner without having to delve too deeply into mathematics. Search engines, machine translation, and mobile applications all rely on deep learning technologies. It works by stimulating the human brain in terms of identifying and creating patterns from various types of input.

A feedforward neural network is a key component of this fantastic technology since it aids software developers with pattern recognition and classification, non-linear regression, and function approximation.

Layers Of Feed Forward Neural Network

The following are the components of a feedforward neural network:

Layer of input

It contains the neurons that receive input. The data is subsequently passed on to the next tier. The input layer’s total number of neurons is equal to the number of variables in the dataset.

Hidden layer

This is the intermediate layer, which is concealed between the input and output layers. This layer has a large number of neurons that perform alterations on the inputs. They then communicate with the output layer.

Output layer

It is the last layer and depends on the model’s construction. Additionally, the output layer is the expected feature, as you are aware of the desired outcome.

Cost Function of feedforward neural network

The cost function is an important factor of a feedforward neural network. Generally, minor adjustments to weights and biases have little effect on the categorized data points. Thus, to determine a method for improving performance by making minor adjustments to weights and biases using a smooth cost function.

2.2.1 Applications of Feedforward Neural Network

These neural networks are utilized in a wide variety of applications. Several of them are denoted by the following area units:

- Physiological feedforward system: Here, feedforward management is exemplified by the usual preventative control of heartbeat prior to exercise by the central involuntary system.
- Gene regulation and feedforward: Throughout this, a theme predominates throughout the famous networks, and this motif has been demonstrated to be a feedforward system for detecting non-temporary atmospheric alteration.
- Automating and managing machines
- Parallel feedforward compensation with derivative: This is a relatively recent approach for converting the non-minimum component of an open-loop transfer system into the minimum part.

Convolutional Neural Networks:

(CNNs) are a special type of FNNs designed specifically for image and video recognition tasks. CNNs can automatically learn features from the images, which makes them well-suited for tasks such as image classification, object detection, and image segmentation. The convolution neural network is explained below.

2.3 Recurrent Neural Networks:

(RNNs) are a type of neural networks that are able to process sequential data, such as time series and natural language. RNNs are able to maintain an internal state that captures information about the previous inputs, which makes them well-suited for tasks such as speech recognition, natural language processing, and language translation.

2.3.1 working

The working of an RNN can be understood with the help of the below example: Suppose there is a deeper network with one input layer, three hidden layers, and one output layer. Then like other neural networks, each hidden layer will have its own set of weights and biases, let's say, for hidden layer 1 the weights and biases are (w1, b1), (w2, b2) for the second hidden layer, and (w3, b3) for the third hidden layer. This means that each of these layers is independent of the other they do not memorize the previous outputs.

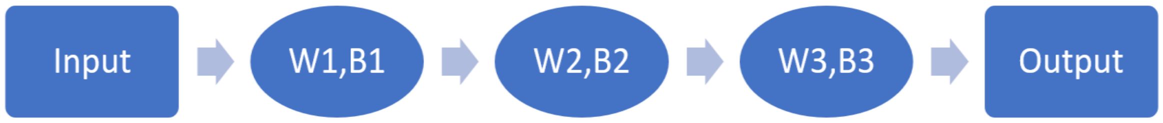


Figure 2.1: Block Diagram of Recurrent Neural Network

Now the RNN will do the following:

- RNN converts the independent activations into dependent activations by providing the same weights and biases to all the layers, thus reducing the complexity of increasing parameters and memorizing each previous output by giving each output as input to the next hidden layer.
- Hence these three layers can be joined together such that the weights and bias of all the hidden layers are the same, in a single recurrent layer.

2.3.2 Training through RNN

1. A single-time step of the input is provided to the network.
2. Then calculate its current state using a set of current input and the previous state.
3. The current ht becomes ht-1 for the next time step.
4. One can go as many time steps according to the problem and join the information from all the previous states.
5. Once all the time steps are completed the final current state is used to calculate the output.
6. The output is then compared to the actual output i.e the target output and the error is generated.
7. The error is then back-propagated to the network to update the weights and hence the network (RNN) is trained.

2.3.3 Advantages of Recurrent Neural Network

- An RNN remembers each piece of information through time. It is useful in time series prediction only because of the feature to remember previous inputs as well. This is called Long Short-Term Memory.
- Recurrent neural networks are even used with convolutional layers to extend the effective pixel neighborhood.

2.3.4 Disadvantages of Recurrent Neural Network.

- Gradient vanishing and exploding problems.
- Training an RNN is a very difficult task.
- It cannot process very long sequences if using tanh or relu as an activation function.

2.3.5 Applications of Recurrent Neural Network

- Language Modelling and Generating Text
- Speech Recognition
- Machine Translation
- Image Recognition
- Face detection
- Time series Forecasting

Deep Learning models are trained using large amounts of labeled data and require significant computational resources. With the increasing availability of large amounts of data and computational resources, deep learning has been able to achieve state-of-the-art performance in a wide range of applications such as image and speech recognition, natural language processing, and more.

Deep learning is a particular kind of machine learning that achieves great power and flexibility by learning to represent the world as a nested hierarchy of concepts, with each concept defined in relation to simpler concepts, and more abstract representations computed in terms of less abstract ones.

In the human brain approximately 100 billion neurons all together this is a picture of an individual neuron and each neuron is connected through thousands of their neighbors. The question here is how we recreate these neurons in a computer. So, we create an artificial structure called an artificial neural net where we have nodes or neurons. We have some neurons for input value and some for-output value and in between, there may be lots of neurons interconnected in the hidden layer.

2.4 Architectures of Deep learning:

1. **Deep Neural Network** – It is a neural network with a certain level of complexity (having multiple hidden layers in between input and output layers). They are capable of modeling and processing non-linear relationships.
2. **Deep Belief Network (DBN)** – It is a class of Deep Neural Network. It is multi-layer belief networks.

   Steps for performing DBN: a. Learn a layer of features from visible units using Contrastive Divergence algorithm. b. Treat activations of previously trained features as visible units and then learn features of features. c. Finally, the whole DBN is trained when the learning for the final hidden layer is achieved.
3. **Recurrent** (perform same task for every element of a sequence) **Neural Network** – Allows for parallel and sequential computation. Similar to the human brain (large feedback network of connected neurons). They are able to remember important things about the input they received and hence enables them to be more precise.

2.5 Working of deep learning.

First, we need to identify the actual problem in order to get the right solution and it should be understood, the feasibility of the Deep Learning should also be checked (whether it should fit Deep Learning or not). Second, we need to identify the relevant data which should correspond to the actual problem and should be prepared accordingly. Third, Choose the Deep Learning Algorithm appropriately. Fourth, Algorithm should be used while training the dataset. fifth final testing is done on dataset.

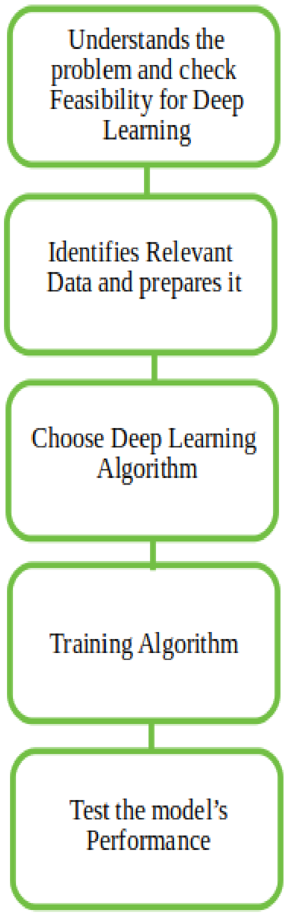

Understands the problem and check Feasibility for Deep Learning
Identifies Relevant Data and prepares it
Choose Deep Learning Algorithm
Training Algorithm
Test the model's Performance

Figure2.2: Working of Deep Learning

Tools used

Anaconda, Jupyter, Pycharm,Matlab etc.

Languages used :

R, Python, Matlab, CPP, Java, Julia, Lisp, Java Script, etc.

Overview

However, they can also include propositional formulas or latent variables organised layer-wise in deep generative models, such as the nodes in deep belief networks and deep Boltzmann machines. The majority of contemporary deep learning models are based on artificial neural networks, specifically convolutional neural networks (CNNs).

Each degree of deep learning learns how to change the incoming data into a tad more abstract and composite representation. In an application for image recognition, the initial input could be a matrix of pixels; the first representational layer could abstract the pixels and encode edges; the second layer could compose and encode arrangements of edges; the third layer could encode a nose and eyes; and the fourth layer could recognise that the image contains a face. Significantly, a deep learning process may learn which characteristics to ideally arrange at which level on its own. This does not eliminate the necessity for manual adjustment; for instance, various layer counts and widths might result in different levels of abstraction.

In "deep learning," the word "deep" refers to the number of layers through which the data is changed. More specifically, deep learning systems have a significant depth of credit assignment paths (CAP). The series of conversions from input to output is known as the CAP. CAPs define the relationships that could exist between input and output. The number of hidden layers plus one determines the depth of the CAPs for a feedforward neural network (as the output layer is also parameterized). The CAP depth for recurrent neural networks, in which a signal may pass through a layer more than once, is theoretically limitless. [13] Deep learning incorporates CAP, but there is no commonly recognised threshold of depth that distinguishes it from shallow learning. It has been demonstrated that the approximator of depth 2 is a universal approximator, capable of simulating any function. After that, adding more layers does not improve the network's capacity to approximate functions. Deep models (CAP > 2) are able to extract better features than shallow models and so, further layers aid in learning the features efficiently.

Unsupervised learning problems can be handled by deep learning algorithms. Since that unlabeled data are more prevalent than labelled data, this is a significant advantage. Deep belief networks are an example of deep structures that can be learned unsupervised.

2.6 Convolution neural network

A Convolutional Neural Network (CNN) is a type of Deep Learning architecture commonly used for image classification and recognition tasks. It consists of multiple layers, including Convolutional layers, Pooling layers, and fully connected layers. The Convolutional layer applies filters to the input image to extract features, the Pooling layer down samples the image to reduce computation, and the fully connected layer makes the final prediction. The network learns the optimal filters through backpropagation and gradient descent.

It is assumed that the reader knows the concept of Neural networks. When it comes to Machine Learning, Artificial neural network performs well. Artificial Neural Networks are used in various classification tasks like image, audio, words. Different types of Neural Networks are used for different purposes, for example for predicting the sequence of words we use Recurrent Neural Networks more precisely an LSTM, similarly for image classification we use Convolution Neural networks. In this blog, we are going to build a basic building block for CNN.

Before diving into the Convolution Neural Network, let us first revisit some concepts of Neural Network.

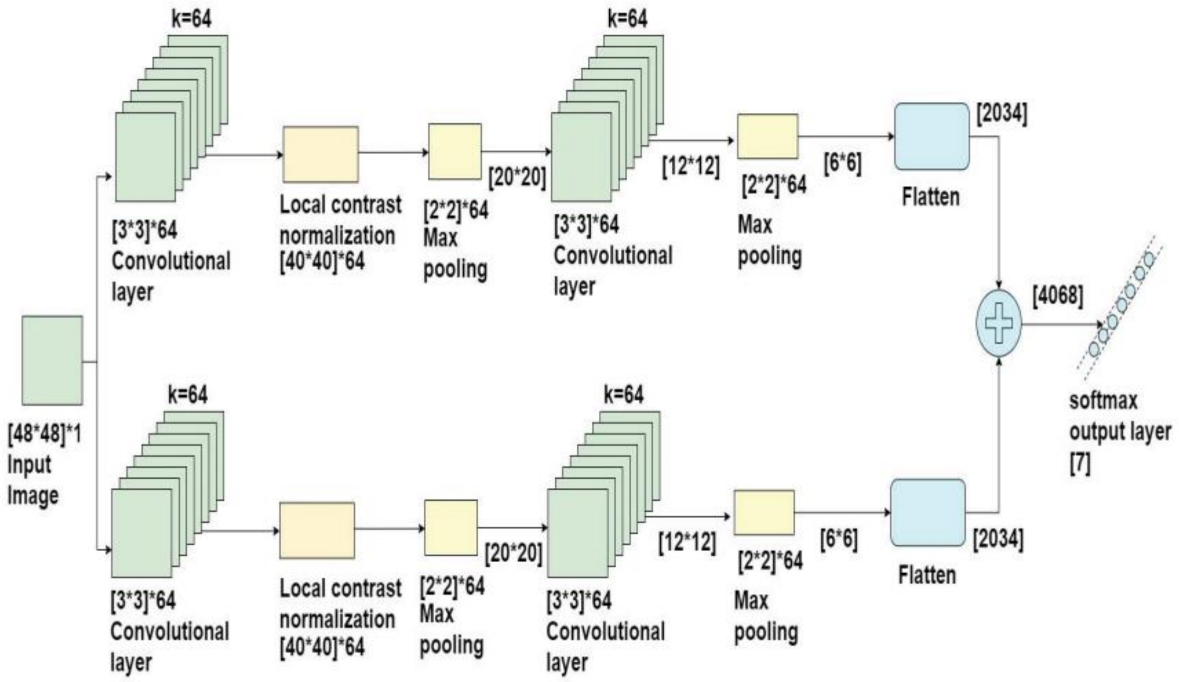


Figure 2.3: Internal Layers of Convolution Neural Networks

In a regular Neural Network, there are three types of layers:

1. **Input Layers:** It's the layer in which we give input to our model. The number of neurons in this layer is equal to the total number of features in our data (number of pixels in the case of an image).
2. **Hidden Layer:** The input from the Input layer is then feed into the hidden layer. There can be many hidden layers depending upon our model and data size. Each hidden layer can have different numbers of neurons which are generally greater than the number of features. The output from each layer is computed by matrix multiplication of output of the previous layer with learnable weights of that layer and then by the addition of learnable biases followed by activation function which makes the network nonlinear.

3. **Output Layer:** The output from the hidden layer is then fed into a logistic function like sigmoid or softmax which converts the output of each class into the probability score of each class.

The data is then fed into the model and output from each layer is obtained this step is called feedforward, we then calculate the error using an error function, some common error functions are cross-entropy, square loss error, etc. After that, we backpropagate into the model by calculating the derivatives. This step is called Backpropagation which basically is used to minimize the loss.

2.6.1 Working of CNN

Convolution Neural Networks or covnets are neural networks that share their parameters. Imagine you have an image. It can be represented as a cuboid having its length, width (dimension of the image), and height (as images generally have red, green, and blue channels).

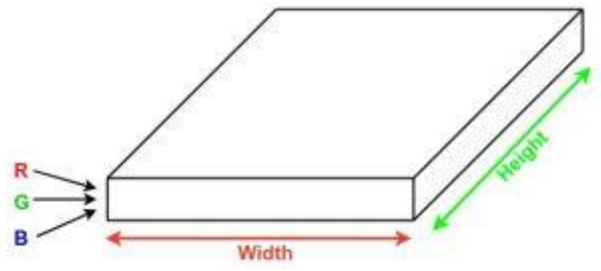


Figure 2.4: Basic Steps Of CNN

Now imagine taking a small patch of this image and running a small neural network on it, with say, k outputs and represent them vertically. Now slide that neural network across the whole image, as a result, we will get another image with different width, height, and depth. Instead of just R, G, and B channels now we have more channels but lesser width and height. This operation is called Convolution. If the patch size is the same as that of the image it will be a regular neural network. Because of this small patch, we have fewer weights.

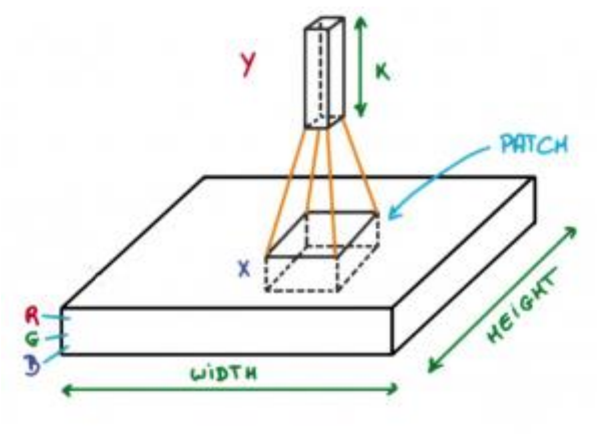


Figure 4.5: Convolution layer

Now let’s talk about a bit of mathematics that is involved in the whole convolution process.

- Convolution layers consist of a set of learnable filters (a patch in the above image). Every filter has small width and height and the same depth as that of input volume (3 if the input layer is image input).
- For example, if we have to run convolution on an image with dimension 34x34x3. The possible size of filters can be axax3, where ‘a’ can be 3, 5, 7, etc but small as compared to image dimension.
- During forward pass, we slide each filter across the whole input volume step by step where each step is called stride (which can have value 2 or 3 or even 4 for high dimensional images) and compute the dot product between the weights of filters and patch from input volume.

- As we slide our filters we’ll get a 2-D output for each filter and we’ll stack them together and as a result, we’ll get output volume having a depth equal to the number of filters. The network will learn all the filters.

Layers used to build ConvNets

A covnets is a sequence of layers, and every layer transforms one volume to another through a differentiable function.

Types of layers:

Let’s take an example by running a covnets on of image of dimension 32 x 32 x 3.

1. **Input Layer:** This layer holds the raw input of the image with width 32, height 32, and depth 3.
2. **Convolution Layer:** This layer computes the output volume by computing the dot product between all filters and image patches. Suppose we use a total of 12 filters for this layer we’ll get output volume of dimension 32 x 32 x 12.
3. **Activation Function Layer:** This layer will apply an element-wise activation function to the output of the convolution layer. Some common activation functions are RELU: max(0, x), Sigmoid: 1/(1+e^-x), Tanh, Leaky RELU, etc. The volume remains unchanged hence output volume will have dimension 32 x 32 x 12.
4. **Pool Layer:** This layer is periodically inserted in the covnets and its main function is to reduce the size of volume which makes the computation fast reduces memory and also prevents overfitting. Two common types of pooling layers are **max pooling** and **average pooling**. If we use a max pool with 2 x 2 filters and stride 2, the resultant volume will be of dimension 16x16x12.

5. **Fully Connected Layer:** This layer is a regular neural network layer that takes input from the previous layer and computes the class scores and outputs the 1-D array of size equal to the number of classes.

2.6.2 Advantages of Convolutional Neural Networks (CNNs):

1. Good at detecting patterns and features in images, videos and audio signals.
2. Robust to translation, rotation, and scaling invariance.
3. End-to-end training, no need for manual feature extraction.
4. Can handle large amounts of data and achieve high accuracy.

2.6.3 Disadvantages of Convolutional Neural Networks (CNNs):

1. Computationally expensive to train and require a lot of memory.
2. Can be prone to overfitting if not enough data or proper regularization is used.
3. Requires large amount of labeled data.
4. Interpretability is limited, it’s hard to understand what the network has learned.

2.7 Introduction to Alexnet

The Alexnet has eight layers with learnable parameters. The model consists of five layers with a combination of max pooling followed by 3 fully connected layers, and they use Relu activation in each of these layers except the output layer.

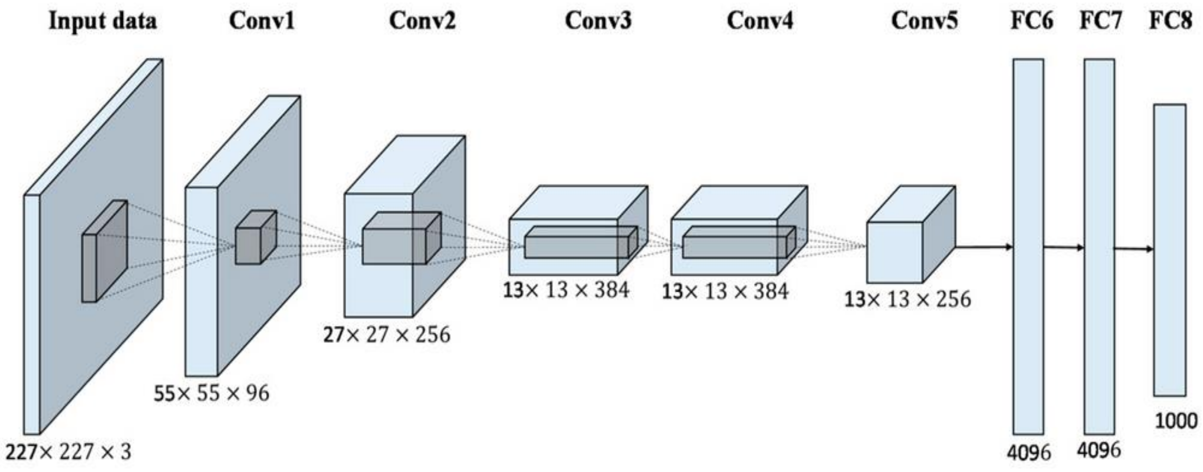


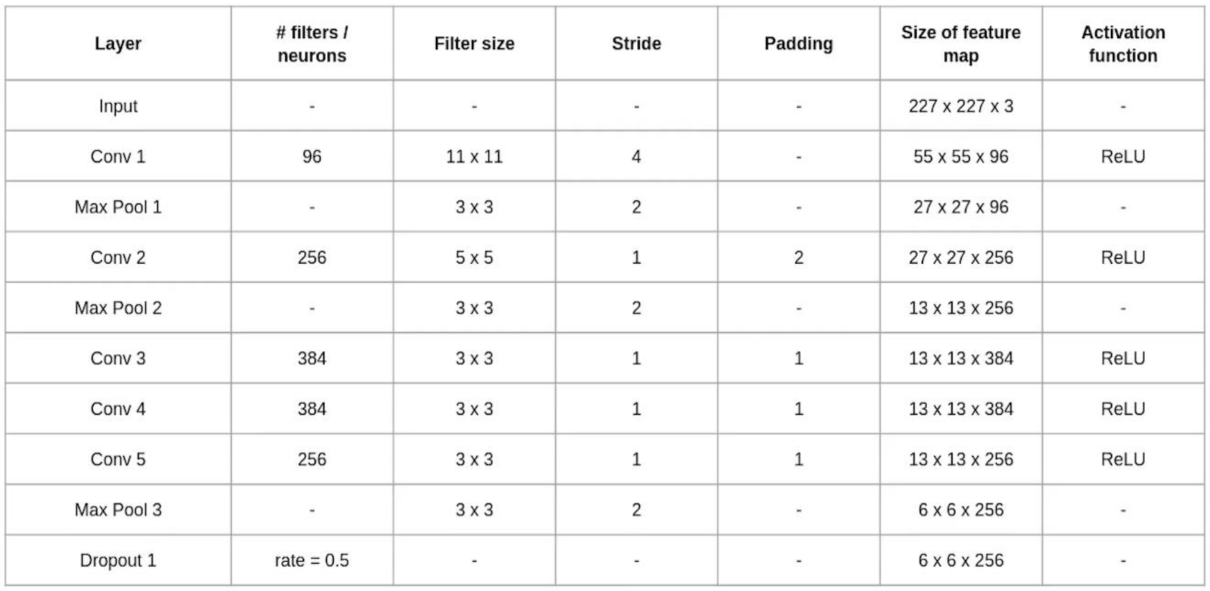
Figure 4.6 Architecture of Alexnet

They found out that using the relu as an activation function accelerated the speed of the training process by almost six times. They also used the dropout layers, that prevented their model from overfitting. Further, the model is trained on the Imagenet dataset. The Imagenet dataset has almost 14 million images across a thousand classes

Let's see the architectural details in this article.

2.7.1 Alexnet Architecture

One thing to note here, since Alexnet is a deep architecture, the authors introduced padding to prevent the size of the feature maps from reducing drastically. The input to this model is the images of size 227X227X3.



Layer	# filters / neurons	Filter size	Stride	Padding	Size of feature map	Activation function
Input	-	-	-	-	227 x 227 x 3	-
Conv 1	96	11 x 11	4	-	55 x 55 x 96	ReLU
Max Pool 1	-	3 x 3	2	-	27 x 27 x 96	-
Conv 2	256	5 x 5	1	2	27 x 27 x 256	ReLU
Max Pool 2	-	3 x 3	2	-	13 x 13 x 256	-
Conv 3	384	3 x 3	1	1	13 x 13 x 384	ReLU
Conv 4	384	3 x 3	1	1	13 x 13 x 384	ReLU
Conv 5	256	3 x 3	1	1	13 x 13 x 256	ReLU
Max Pool 3	-	3 x 3	2	-	6 x 6 x 256	-
Dropout 1	rate = 0.5	-	-	-	6 x 6 x 256	-

Table 2.1: Stages of Alexnet

2.7.2 Convolution and Maxpooling Layers

Then we apply the first convolution layer with 96 filters of size 11X11 with stride 4. The activation function used in this layer is relu. The output feature map is 55X55X96. In case, you are unaware of how to calculate the output size of a convolution layer.

output= ((Input-filter size)/ stride) +1

Also, the number of filters becomes the channel in the output feature map.

Next, we have the first Maxpooling layer, of size 3X3 and stride 2. Then we get the resulting feature map with the size 27X27X96.

After this, we apply the second convolution operation. This time the filter size is reduced to 5X5, and we have 256 such filters. The stride is 1 and padding 2. The activation function used is again relu. Now the output size we get is 27X27X256.

Again, we applied a max-pooling layer of size 3X3 with stride 2. The resulting feature map is of shape 13X13X256.

Now we apply the third convolution operation with 384 filters of size 3X3 stride 1 and also padding 1. Again, the activation function used is relu. The output feature map is of shape 13X13X384.

Then we have the fourth convolution operation with 384 filters of size 3X3. The stride along with the padding is 1. On top of that activation function used is relu. Now the output size remains unchanged i.e 13X13X384.

After this, we have the final convolution layer of size 3X3 with 256 such filters. The stride and padding are set to one also the activation function is relu. The resulting feature map is of shape 13X13X256.

So, if you look at the architecture till now, the number of filters is increasing as we are going deeper. Hence it is extracting more features as we move deeper into the architecture. Also, the filter size is reducing, which means the initial filter was larger and as we go ahead the filter size is decreasing, resulting in a decrease in the feature map shape.

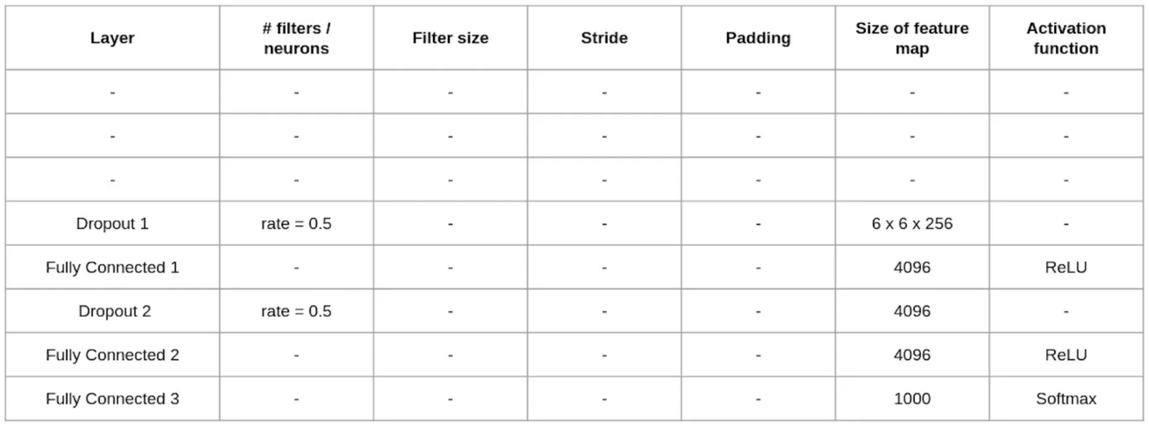
Next, we apply the third max-pooling layer of size 3X3 and stride 2. Resulting in the feature map of the shape 6X6X256.

2.7.3 Fully Connected and Dropout Layers

After this, we have our first dropout layer. The drop-out rate is set to be 0.5.

Then we have the first fully connected layer with a relu activation function. The size of the output is 4096. Next comes another dropout layer with the dropout rate fixed at 0.5.

This followed by a second fully connected layer with 4096 neurons and relu activation.



Layer	# filters / neurons	Filter size	Stride	Padding	Size of feature map	Activation function
-	-	-	-	-	-	-
-	-	-	-	-	-	-
-	-	-	-	-	-	-
Dropout 1	rate = 0.5	-	-	-	6 x 6 x 256	-
Fully Connected 1	-	-	-	-	4096	ReLU
Dropout 2	rate = 0.5	-	-	-	4096	-
Fully Connected 2	-	-	-	-	4096	ReLU
Fully Connected 3	-	-	-	-	1000	Softmax

Table 2.2. Fully Connected and Dropout Layers

Finally, we have the last fully connected layer or output layer with 1000 neurons as we have 10000 classes in the data set. The activation function used at this layer is Softmax.

Pooling

The pooling operation involves sliding a two-dimensional filter over each channel of feature map and summarising the features lying within the region covered by the filter. For a feature map having dimensions $\mathbf{n_h \times n_w \times n_c}$, the dimensions of output obtained after a pooling layer is.

$$(n_h - f + 1) / s \times (n_w - f + 1) / s \times n_c$$

where,

-> $\mathbf{n_h}$ - height of feature map

-> $\mathbf{n_w}$ - width of feature map

-> $\mathbf{n_c}$ - number of channels in the feature map

-> **f -** size of filter

-> **s -** stride length

A common CNN model architecture is to have a number of convolution and pooling layers stacked one after the other.

2.7.4 Why to use Pooling Layers?

Pooling layers are used to reduce the dimensions of the feature maps. Thus, it reduces the number of parameters to learn, and the amount of computation performed in the network.

The pooling layer summarizes the features present in a region of the feature map generated by a convolution layer. So, further operations are performed on summarized features instead of precisely positioned features generated by the convolution layer. This makes the model more robust to variations in the position of the features in the input image.

2.7.5 Types of Pooling Layers:

1. Max pooling
2. Average polling

Max Pooling

Max pooling is a pooling operation that selects the maximum element from the region of the feature map covered by the filter. Thus, the output after max-pooling layer would be a feature map containing the most prominent features of the previous feature map.

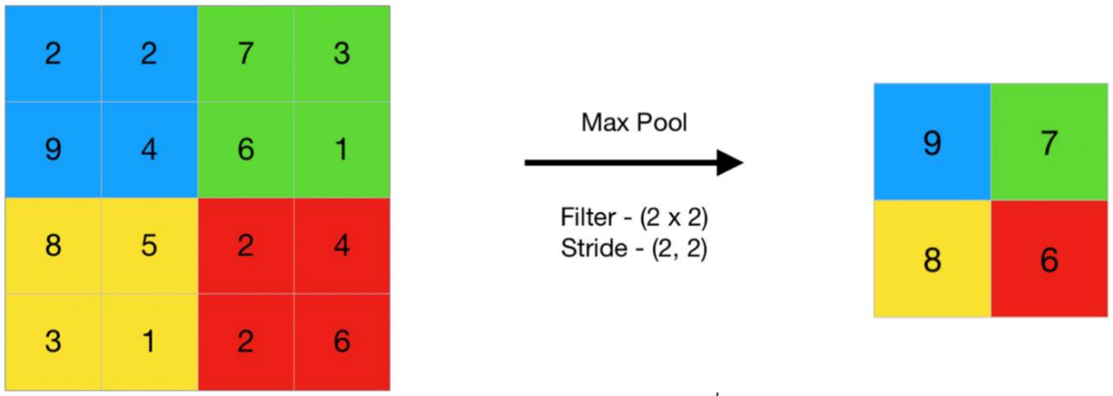
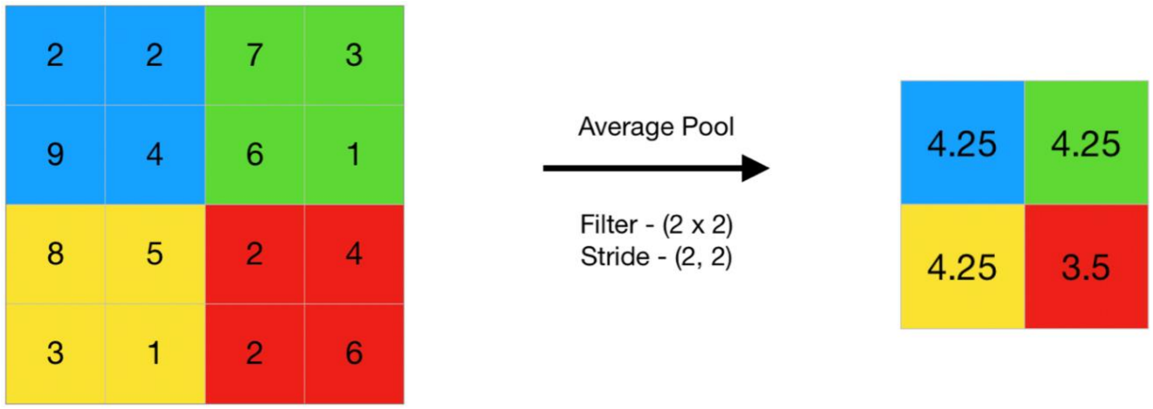


Figure 2.7: Max Pooling

Average Pooling

Average pooling computes the average of the elements present in the region of feature map covered by the filter. Thus, while max pooling gives the most prominent feature in a particular patch of the feature map, average pooling gives the average of



features present in a patch.

Figure 2.8: Average Pooling

2.7.5 Pros of AlexNet

1. AlexNet is considered as the milestone of CNN for image classification.
2. Many methods, such as the conv+pooling design, dropout, GPU, parallel computing, ReLU, are still the industrial standard for computer vision.
3. The unique advantage of AlexNet is the direct image input to the classification model.
4. The convolution layers can automatically extract the edges of the images and fully connected layers learning these features.
5. Theoretically the complexity of visual pattern scan be effective extracted by adding more conv layer.

2.7.6 Cons of AlexNet

1. AlexNet is NOT deep enough compared to the later model, such as VGGNet, GoogLENet, and ResNet.
2. The use of large convolution filters (5*5) is not encouraged shortly after that.
3. Use normal distribution to initiate the weights in the neural networks, cannot effectively solve the problem of gradient vanishing, replaced by the Xavier method later.
4. The performance is surpassed by more complex models such as GoogLENet (6.7%), and ResNet (3.6%)

Cascade object detector

The Computer Vision Toolbox cascade object detector can detect object categories whose aspect ratio does not vary significantly. Objects whose aspect ratio remains fixed include faces, stop signs, and cars viewed from one side.

The vision CascadeObjectDetector System object detects objects in images by sliding a window over the image. The detector then uses a cascade classifier to decide whether the window contains the object of interest. The size of the window varies to detect objects at different scales, but its aspect ratio remains fixed. The detector is very sensitive to out-of-plane rotation, because the aspect ratio changes for most 3-D objects. Thus, you need to train a detector for each orientation of the object. Training a single detector to handle all orientations will not work.

How Does the Cascade Classifier Work?

The cascade classifier consists of stages, where each stage is an ensemble of weak learners. The weak learners are simple classifiers called decision stumps. Each stage is trained using a technique called boosting. Boosting provides the ability to train a highly accurate classifier by taking a weighted average of the decisions made by the weak learners.

Each stage of the classifier labels the region defined by the current location of the sliding window as either positive or negative. *Positive* indicates that an object was found and negative indicates no objects were found. If the label is negative, the classification of this region is complete, and the detector slides the window to the next location. If the label is positive, the classifier passes the region to the next stage. The detector reports an object found at the current window location when the final stage classifies the region as positive.

The stages are designed to reject negative samples as fast as possible. The assumption is that the vast majority of windows do not contain the object of interest. Conversely, true positives are rare and worth taking the time to verify.

- A true positive occurs when a positive sample is correctly classified.
- A false positive occurs when a negative sample is mistakenly classified as positive.
- A false negative occurs when a positive sample is mistakenly classified as negative.

To work well, each stage in the cascade must have a low false negative rate. If a stage incorrectly labels an object as negative, the classification stops, and you cannot correct the mistake. However, each stage can have a high false positive rate. Even if the detector incorrectly labels a nonobject as positive, you can correct the mistake in subsequent stages.

The overall false positive rate of the cascade classifier is f^s, where f is the false positive rate per stage in the range (0 1), and s is the number of stages. Similarly, the overall true positive rate is t^s, where t is the true positive rate per stage in the range (0 1]. Thus, adding more stages reduces the overall false positive rate, but it also reduces the overall true positive rate.

Chapter-3
Literature Survey

The process of emotion recognition involves the processing images and detecting the face then extracting the facial feature. There are different methods that are used for Face Expression Recognition in the last 20 years but usually they are separated into two main methods: Conventional or Traditional FER Approach and Deep Learning-based approach.

- In [1] Deep neural networks are being used more often to train discriminative representations for automated facial expression recognition (FER) as the field moves from controlled lab settings to the more difficult situations of the real world. Current deep FER systems often concentrate on two key problems: overfitting brought on by a dearth of training data and expression-unrelated variables including lighting, head posture, and identification bias. In this article, we present a thorough overview of deep FER, complete with datasets and techniques that shed light on these fundamental issues. The usual pipeline of a deep FER system is first described, along with background information and ideas for practical implementations for each level. Then, we introduce the widely-used datasets that are currently available.in the literature and offer generally agreed guidelines for choosing and assessing the data in these datasets. Reviewing current innovative deep neural networks and associated training methods created for FER based on both static pictures and dynamic image sequences, we analyse the benefits and drawbacks of each. This is the state of the art in deep FER. This section also summarizes competitive results on frequently recognized benchmarks. Afterwards, we expand our poll to cover more relevant problems and usage scenarios. Finally, we discuss the field's remaining difficulties, associated possibilities, and potential future paths for the development of reliable deep FER systems.

- In [2] one of the biggest issues with computer vision is the identification of human emotion. Human emotions are made up of a number of sub-emotions that are challenging to categorise. In the suggested study, we have attempted to categorise human emotions into six main categories: joyful, sad, disgusted, fearful, surprised, and angry. We have developed a deep learning framework that combines CNN, ResNet, and an attention block to offer the network visual perceptibility. The suggested approach is more practical for identifying facial expressions of emotion in daily life. For the FER dataset, the suggested model produced good results and demonstrated effectiveness.

- In [3] this research,The majority of computerised expression analysis programmes try to identify a limited number of prototypical expressions (e.g. happiness and anger). However these prototypic phrases don't happen very often. Changes in one or two distinct face characteristics are more frequently used to convey human emotions and intentions. Based on both permanent face characteristics (brows, eyes, and mouth) and transitory facial features (deepening of facial furrows) in a nearly frontal picture series, we create an autonomous method to evaluate small changes in facial emotions. Our approach, in contrast to most others, makes an effort to identify subtle changes in facial expression based on the six fundamental facial expressions of the Facial Action Coding System (FACS) action units (AUs) (e.g. happiness and anger). It is suggested to use multi-state face and facial component models for tracking and modelling various facial aspects, such as lips, eyes, brows, cheeks, and associated facial furrows and wrinkles. The tracking findings are then transformed into in-depth parametric descriptions of the face characteristics. 11 lower face action units (AUs) and 7 upper face AUs are detected by a neural network algorithm using these attributes as inputs. For lower face AUs, the recognition rate is 96.7%, while for upper face AUs, it is 95%. The results of recognition show that our system can recognise action units whether they appear individually or in combinations.

- In [4] Charles Darwin (1872-1965) maintained the claim that emotion displays are developed and adaptable (at least at one time in the past) and serve a significant communication purpose in his book The Expression of the Emotions in Man and Animals. His book's concepts had a significant influence on the subject and gave rise to several fruitful areas of study. This article first outlines Darwin's three guiding principles in this field before going into a few of the current study areas that stem from his theoretical outlook. Focus is placed on five topics in particular: (a) the question of what emotions are expressed by various expressions of emotion; (b) the idea that some expressions of emotion are universal; (c) the issue of emotion prototypes; and (d) the problem of animal emotions.

- In [5] Face expressions are a key component of nonverbal communication, and as we go towards digitalization, human-computer interactions will become increasingly important. Expressions vary as a result of emotional changes. This study describes the construction and training of a deep convolutional neural network model using tf.keras. The goal is to use open CV and one of its classifiers to classify facial images into one of the seven face detection classifiers in order to draw a border box around the face and identify the appropriate expression. We utilised 48x48 greyscale photos from Kaggle's ICMP 2013-Fecial Expression Recognition (FER) dataset to train the CNN models.The FER dataset is separated into two folders, test and train, each of which has a distinct folder containing one of the seven FER dataset types. Minority classes are generated using the augmentation approach in order to comprehend the distribution of the class data. Dropout and batch normalisation are employed to reduce over-fitting of the models. Because this is a multiclass classification issue, we are utilising the atom optimizer and softmax activation function. We are training a categorical cross entropy matrix for accuracy based on the parameters to assess the effectiveness of the created CNN model by examining the training epoch history.

- In [6] one The Cohn-Kanade (CK) database was made public in 2000 with the intention of encouraging research into the automatic recognition of certain facial expressions. Since then, one of the most popular test-beds for algorithm creation and assessment is the CK database. Three restrictions have become clear throughout this time: Although AU codes are thoroughly vetted, emotion labels are not since they correspond to what was requested rather than what was really done. Other issues include the absence of a standard performance criterion for assessing new algorithms, as well as the lack of standard protocols for common databases. As both AU and emotion identification have been done using the CK database (even though the labels for the latter have not been confirmed), comparison with benchmark methods is possible.

  Meta-analyses are challenging because of missing data and the usage of random selections from the original database. We introduce the Expanded Cohn-Kanade (CK+) database to solve these and other issues. The number of patients increases by 27%, whereas the number of sequences increases by 22%. Each sequence's target expression has a complete FACS code, and the emotion labels have been updated and verified. Moreover, non-posed sequences for a number of different grins have been provided, along with the information that goes with them. For both AU and emotion recognition for the posed data, we show baseline findings using Active Appearance Models (AAMs) and a linear support vector machine (SVM) classifier utilising a leave-one-out subject cross-validation. The extended picture data, tracked landmarks, and emotion and AU labels will all be made public in July 2010.

- In [7] backpropagation-of-error algorithm (backprop), which has been the subject of much debate, is thought to represent a model of how the brain learns. The typical temporal analogue of backprop in recurrent neural networks for machine learning is backpropagation-through-time (BPTT), although there is even less agreement on whether BPTT is related to the brain. Even in machine learning, several difficult temporal credit assignment (TCA) tasks that the brain is known to be capable of handling have shown that the usage of BPTT in traditional neural network topologies is inadequate. Nevertheless, new memory- and attention-based architectures and algorithms, some of which are brain-inspired, have been used in recent machine learning research to make headway in addressing challenging TCA issues. It's significant that these modern machine learning techniques were created in BPTT's position as a helpful normative framework for considering temporal credit assignment in both artificial and biological systems is hence strengthened in the context of, and with reference to, BPTT.

- In [8] The primary focus of this article is facial expression recognition using face parsing's components (FP). A suggestion is made to recognize facial expression utilizing components that are active in expression disclosure in light of the drawback that various portions of the face contain varied amounts of information for facial expression and the weighted function is not the same for different faces. Deep belief networks are used to train the face parsing detectors, and logistic regression is used to fine-tune them. The detectors identify faces first, then nose, eyes, and mouth in a hierarchical order. With the help of the focused characteristics of components that have been recognized, a deep architecture pretrained using stacked autoencoder is used to recognize face expressions. Images do not need to be aligned or in any other way prepared for expression recognition since the parsing components eliminate the unnecessary information artificial medical care. The efficacy and reliability of this technique are demonstrated by experimental results on the Japanese Female Facial Expression database and expanded Cohn-Kanade dataset, which outperform previous approaches.

- In [9] To categorize the 1.2 million high-resolution photographs entered in the ImageNet LSVRC-2010 contest into the 1000 separate classes, we trained a large, deep convolutional neural network. Our top-1 and top-5 error rates on the test data were 37.5% and 17.0%, respectively, which is significantly better than the prior state-of-the-art. The neural network, which has 650,000 neurons and 60 million parameters, is made up of three fully connected layers, a final 1000-way softmax, three convolutional layers, some of which are followed by max-pooling layers, and five convolutional layers. We employed non-saturating neurons and a very effective GPU version of the convolution process to speed up training. We used the "dropout" regularisation technique, a recently discovered regularisation technique, to significantly minimise overfitting in the fully linked layers.We also submitted a different version of this model to the ILSVRC-2012 competition, where it won with a top-5 test error rate of 15.3% as opposed to the second-best entry's 26.2%.

- In [10] An increase in sophisticated and reliable models and algorithms for indexing, retrieving, organising, and interacting with photos and multimedia data may result from the explosion of image data on the Internet. Here, we introduce "ImageNet," a brand-new database that is a sizable ontology of pictures constructed on top of the WordNet framework. The bulk of WordNet's 80,000 synsets will be filled with an average of 500–1000 crisp, full-resolution pictures thanks to ImageNet. The outcome will be tens of millions of annotated photographs arranged according to WordNet's semantic hierarchy. This paper provides a thorough evaluation of ImageNet as it is today: 5247 synsets and 3.2 million in 12 subtrees pictures overall. We demonstrate that compared to the present picture databases, ImageNet is significantly more accurate, diverse, and big in scale. Building a database of this size is a difficult endeavour. We outline the method of data collecting using Amazon Mechanical Turk. Finally, we provide three straightforward applications in automated object clustering, picture classification, and object identification to demonstrate the utility of ImageNet. We anticipate that ImageNet's size, precision, variety, and hierarchical structure will present academics working in the field of computer vision and beyond with unmatched opportunities.

Chapter - 4
Proposed model

In the facial emotion recognition, we are using Taiwanese Facial Expression Image Database (TFEID) dataset for training the network. the project mainly depends one two vital steps they are training and testing. Without these steps we cannot get the output as effective.

4.1. Training

The training process follows these steps.

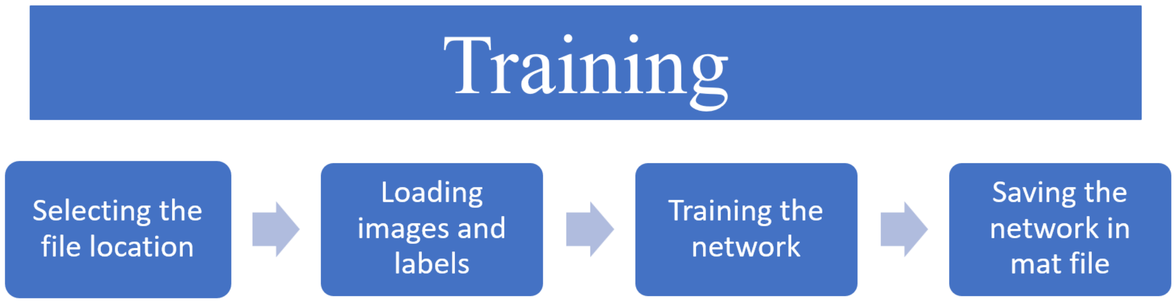

Training
Selecting the file location
Loading images and labels
Training the network
Saving the network in mat file

Figure4.1: Block Diagram of Training

4.1.1. Selecting the file location

Use an Image Datastore object to manage a collection of image files, where each individual image fits in memory, but the entire collection of images does not necessarily fit. You can create an ImageDatastore object using the ImageDatastore function, specify its properties, and then import and process the data using object functions.
Files or folders included in the datastore, specified as a FileSet object, as file paths, or as a DsFileSet object.

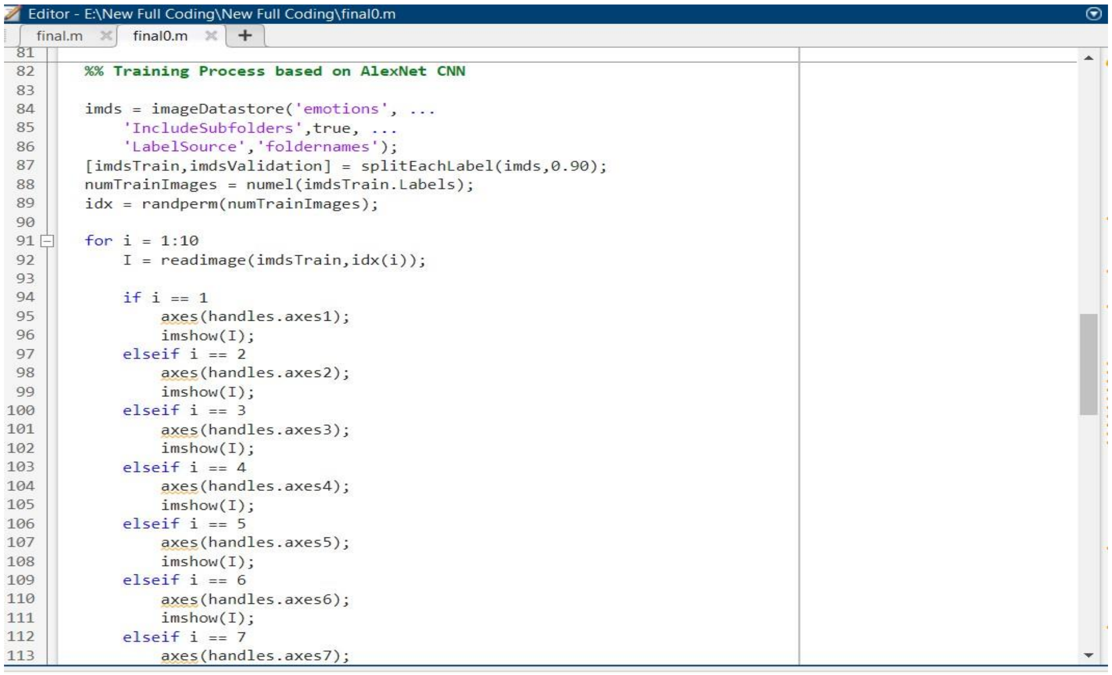
- FileSet object — You can specify location as a FileSet object. Specifying the location as a FileSet object leads to a faster construction time for datastores compared to specifying a path or DsFileSet object.

- File path — You can specify a single file path as a character vector or string scalar. You can specify multiple file paths as a cell array of character vectors or a string array.
- DsFileSet object — You can specify a DsFileSet object.

Files or folders may be local or remote:

- Local files or folders — Specify local paths to files or folders. If the files are not in the current folder, then specify full or relative paths. Files within subfolders of the specified folder are not automatically included in the datastore. You can use the wildcard character (*) when specifying the local path. This character specifies that the datastore include all matching files or all files in the matching folders.
- Remote files or folders — Specify full paths to remote files or folders as a uniform resource locator (URL) of the form.

When you specify a folder, the datastore includes only files with supported file formats and ignores files with any other format.



```
%% Training Process based on AlexNet CNN

imds = imageDatastore('emotions', ...
    'IncludeSubfolders',true, ...
    'LabelSource','foldernames');
[imdsTrain,imdsValidation] = splitEachLabel(imds,0.90);
numTrainImages = numel(imdsTrain.Labels);
idx = randperm(numTrainImages);

for i = 1:10
    I = readimage(imdsTrain,idx(i));

    if i == 1
        axes(handles.axes1);
        imshow(I);
    elseif i == 2
        axes(handles.axes2);
        imshow(I);
    elseif i == 3
        axes(handles.axes3);
        imshow(I);
    elseif i == 4
        axes(handles.axes4);
        imshow(I);
    elseif i == 5
        axes(handles.axes5);
        imshow(I);
    elseif i == 6
        axes(handles.axes6);
        imshow(I);
    elseif i == 7
        axes(handles.axes7);
```

Figure 4.2: code for selecting image.

To specify a custom list of file extensions to include in your datastore, see the FileExtensions property.

The ImageDatastore function supports files that have an imformats format.

Example: "file1.jpg"

Example: "../dir/data/file1.png"

Example: ["C:\dir\data\file1.tif","C:\dir\data\file2.tif"]

4.1.2. Loading images and labels

By selecting the dataset location, we have to load all folders which contains images and its folder name. The names and images of the folder as follows

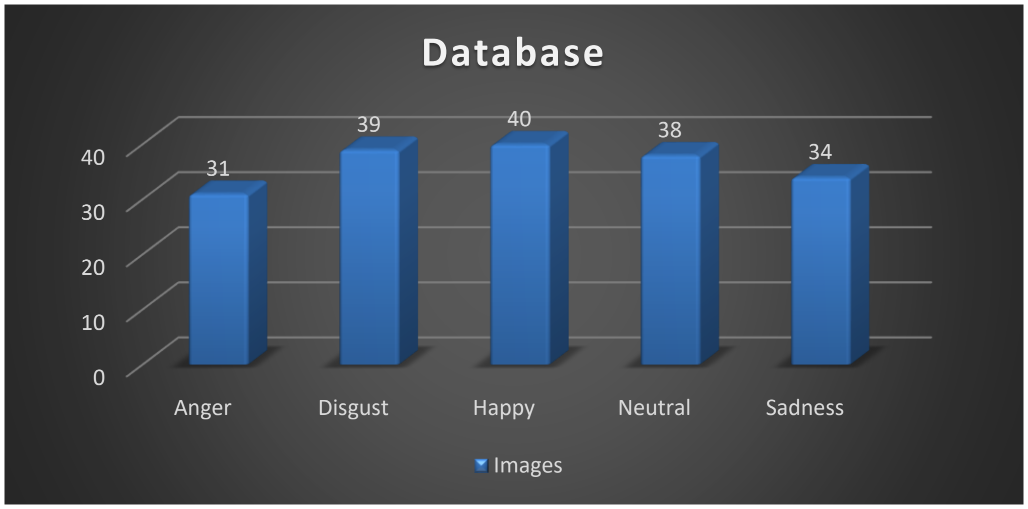


Figure 4.3: Bar Graph Representation of Database

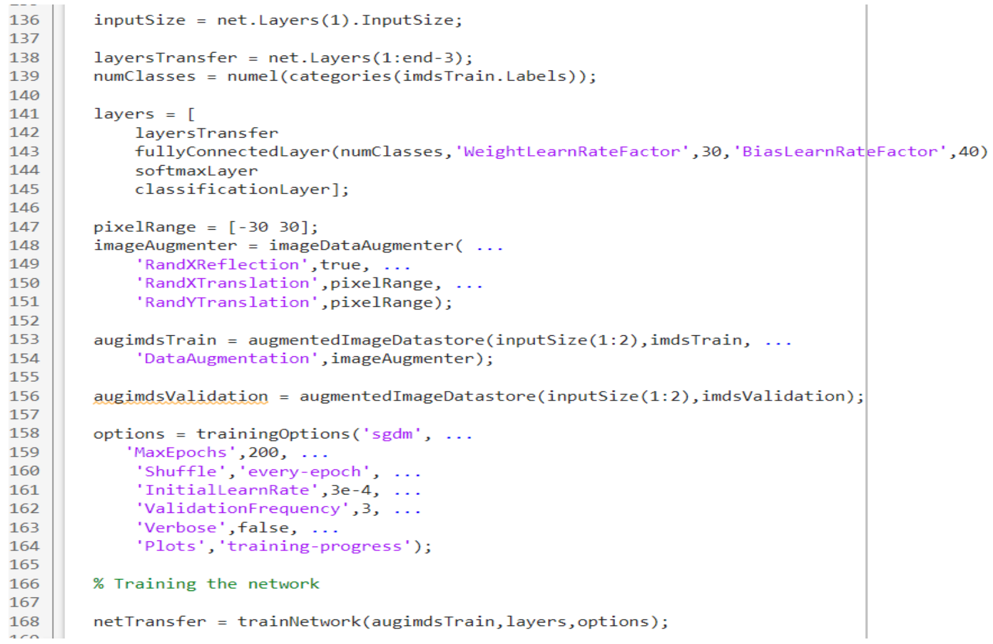
4.1.3. Training the network

The network requires input images of size 227x227x3, but the images in the image datastores have different sizes. Use an augmented image datastore to automatically resize the training images. Specify additional augmentation operations to perform on the training images: randomly flip the training images along the vertical axis, and randomly translate them up to 30 pixels horizontally and vertically. Data augmentation helps prevent the network from overfitting and memorizing the exact details of the training images. To automatically resize the validation images without performing further data augmentation, use an augmented image datastore without specifying any additional pre-processing operations.

Specify the training options. For transfer learning, keep the features from the early layers of the pretrained network (the transferred layer weights). To slow down learning in the transferred layers, set the initial learning rate to a small value. In the previous step, you increased the learning rate factors for the fully connected layer to speed up learning in the new final layers. This combination of learning rate settings results in fast learning only in the new layers and slower learning in the other layers. When performing transfer learning, you do not need to train for as many epochs. An epoch is a full training cycle on

the entire training data set. Specify the mini-batch size and validation data. The software validates the network every ValidationFrequency iterations during training.

Train the network that consists of the transferred and new layers. By default, trainNetwork uses a GPU if one is available, otherwise, it uses a CPU. Training on a GPU requires Parallel Computing Toolbox™ and a supported GPU device.



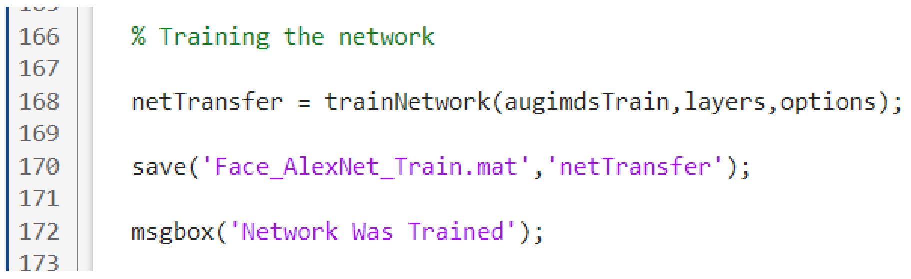
```
inputSize = net.Layers(1).InputSize;

layersTransfer = net.Layers(1:end-3);
numClasses = numel(categories(imdsTrain.Labels));

layers = [
    layersTransfer
    fullyConnectedLayer(numClasses,'WeightLearnRateFactor',30,'BiasLearnRateFactor',40)
    softmaxLayer
    classificationLayer];

pixelRange = [-30 30];
imageAugmenter = imageDataAugmenter( ...
    'RandXReflection',true, ...
    'RandXTranslation',pixelRange, ...
    'RandYTranslation',pixelRange);

augimdsTrain = augmentedImageDatastore(inputSize(1:2),imdsTrain, ...
    'DataAugmentation',imageAugmenter);

augimdsValidation = augmentedImageDatastore(inputSize(1:2),imdsValidation);

options = trainingOptions('sgdm', ...
   'MaxEpochs',200, ...
    'Shuffle','every-epoch', ...
    'InitialLearnRate',3e-4, ...
    'ValidationFrequency',3, ...
    'Verbose',false, ...
    'Plots','training-progress');

% Training the network

netTransfer = trainNetwork(augimdsTrain,layers,options);
```

Figure 4.4: Code for Training the Network

For information on supported devices. You can also specify the execution environment by using the 'Execution Environment' name-value pair argument of trainingOptions.

4.1.4. Saving the network into mat file

The trained network is saved into mat file. The extinction of the MATLAB file is .m where all the trained network is saved into the MATLAB file.



```
% Training the network

netTransfer = trainNetwork(augimdsTrain,layers,options);

save('Face_AlexNet_Train.mat','netTransfer');

msgbox('Network Was Trained');
```

Figure 4.5: Saving the Network.

4.2. Testing

The testing process follows these steps.

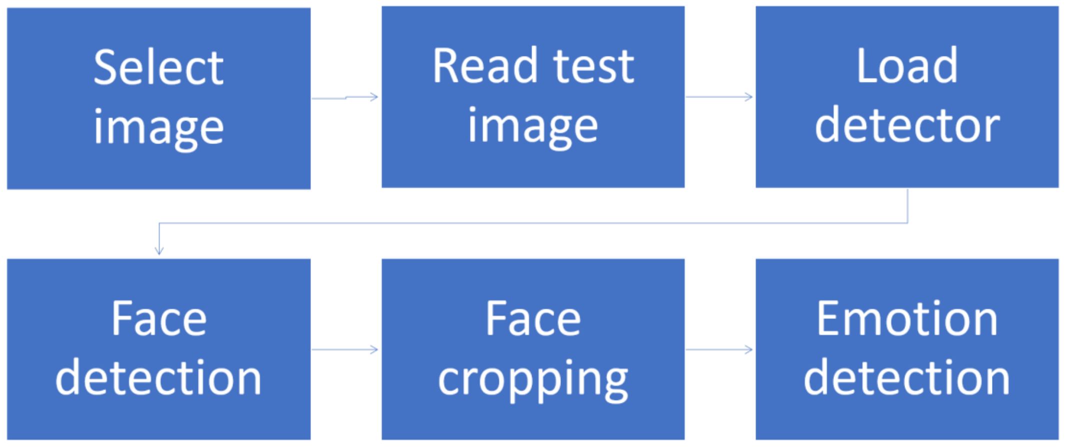
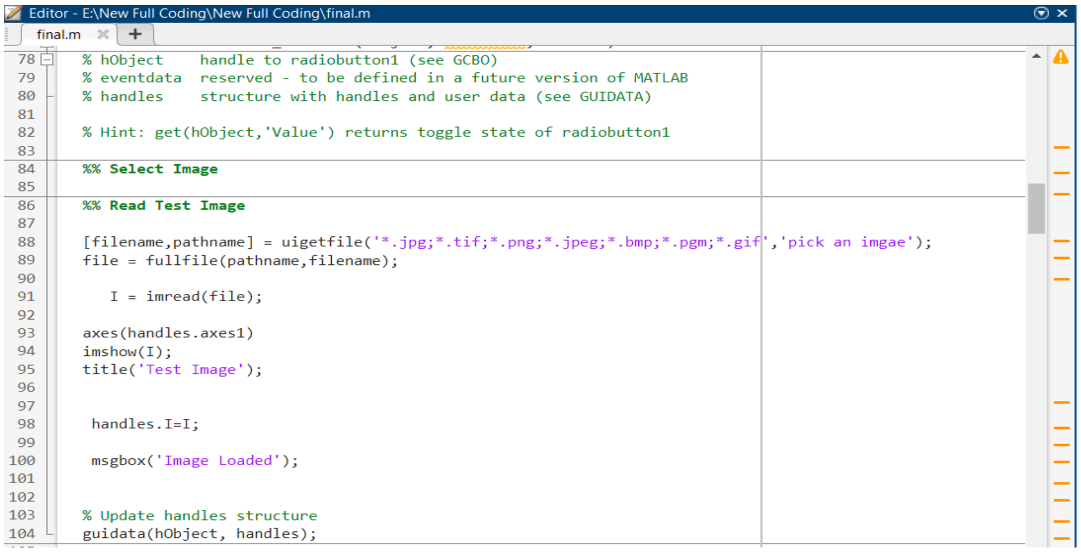


Figure 4.6: Block Diagram of Testing

4.2.1. Select image.

- To select the image, we need to call the dialog box so that's why use 'uigetfile'
- It used to open a model dialog box that lists files in the current folder.
- It enables a user to select or enter the name of a file. If the file exists and is valid, uigetfile returns the file name when the user clicks **Open**.
- If the user clicks Cancel or the window close button (X), uigetfile returns 0.



```matlab
% hObject    handle to radiobutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of radiobutton1

%% Select Image

%% Read Test Image

[filename,pathname] = uigetfile('*.jpg;*.tif;*.png;*.jpeg;*.bmp;*.pgm;*.gif','pick an imgae');
file = fullfile(pathname,filename);

   I = imread(file);

axes(handles.axes1)
imshow(I);
title('Test Image');


 handles.I=I;

 msgbox('Image Loaded');


% Update handles structure
guidata(hObject, handles);
```

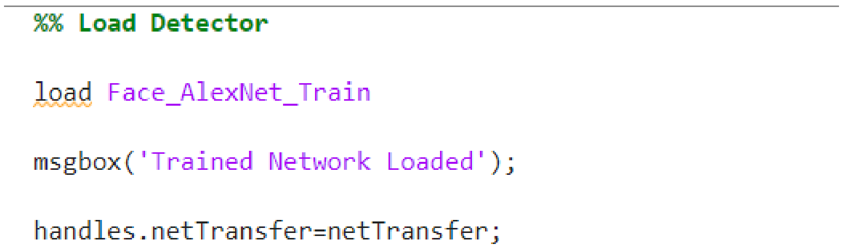
Figure 4.7: selecting and reading the test image.

4.2.2. Read test image.

- To read the test image we need to use the keywork 'imread'
- An imread(filename) reads the image from the file specified by filename, inferring the format of the file from its contents. If filename is a multi-image file, then imread reads the first image in the file.
- After the reading we need to store it in a variable for the further use.

4.2.3. Load Detector

We must load the previous trained image the which is stored in math file (.m file). That happen by calling the saved file by the help of the file name.



```
%% Load Detector

load Face_AlexNet_Train

msgbox('Trained Network Loaded');

handles.netTransfer=netTransfer;
```

Figure 4.8: loading the trained data.

4.2.4. Face Detection

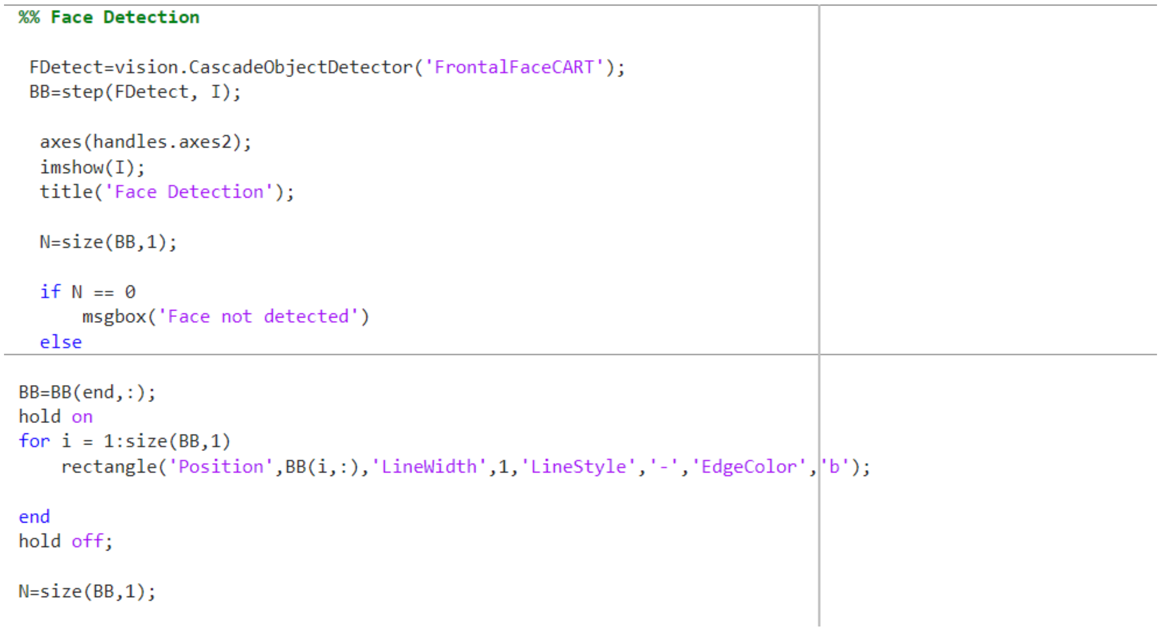
Face detection involves separating image windows into two classes; one containing faces (turning the background (clutter). It is difficult because although commonalities exist between faces, they can vary considerably in terms of age, skin colour and facial expression. The problem is further complicated by differing lighting conditions, image qualities and geometries, as well as the possibility of partial occlusion and disguise. An ideal face detector would therefore be able to detect the presence of any face under any set of lighting conditions, upon any background. The face detection task can be broken down into two steps. The first step is a classification task that takes some arbitrary image as input and outputs a binary value of yes or no, indicating whether there are any faces present in the image. The second step is the face localization task that aims to take an image as input and output the location of any face or faces within that image as some bounding box with (x, y, width, height).

The face detection system can be divided into the following steps: -

1. Pre-Processing: To reduce the variability in the faces, the images are processed before they are fed into the network. All positive examples that is the face images are obtained by cropping images with frontal faces to include only the front view. All the cropped images are then corrected for lighting through standard algorithms.

2. Classification: Neural networks are implemented to classify the images as faces or nonfaces by training on these examples. We use both our implementation of the neural network and the Matlab neural network toolbox for this task. Different network configurations are experimented with to optimize the results.

3. Localization: The trained neural network is then used to search for faces in an image and if present localize them in a bounding box. Various Feature of Face on which the work has done on: - Position Scale Orientation Illumination



```
%% Face Detection

 FDetect=vision.CascadeObjectDetector('FrontalFaceCART');
 BB=step(FDetect, I);

  axes(handles.axes2);
  imshow(I);
  title('Face Detection');

  N=size(BB,1);

  if N == 0
      msgbox('Face not detected')
  else
BB=BB(end,:);
hold on
for i = 1:size(BB,1)
    rectangle('Position',BB(i,:),'LineWidth',1,'LineStyle','-','EdgeColor','b');

end
hold off;

N=size(BB,1);
```

Figure 4.9: Face detection

4.2.5. Face Cropping

Once the face has been detected by the Cascade object detector, a simple MATLAB routine was written to crop the face image by detecting the coordinates of the top-left corner, the height and width of the face enclosing rectangle. The cropped image is then saved in a predefined folder to be used in the facial expression recognition stage.

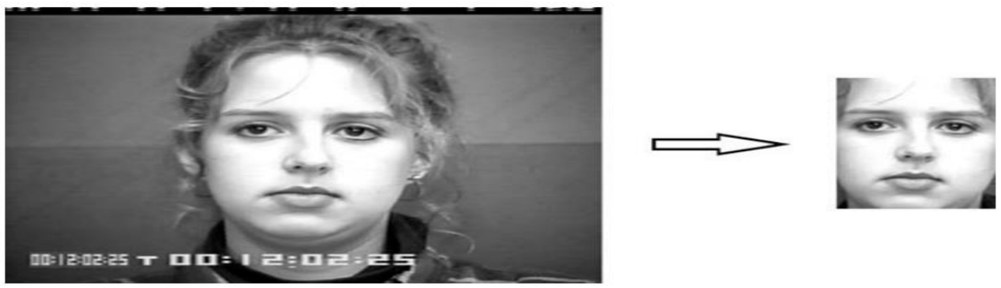
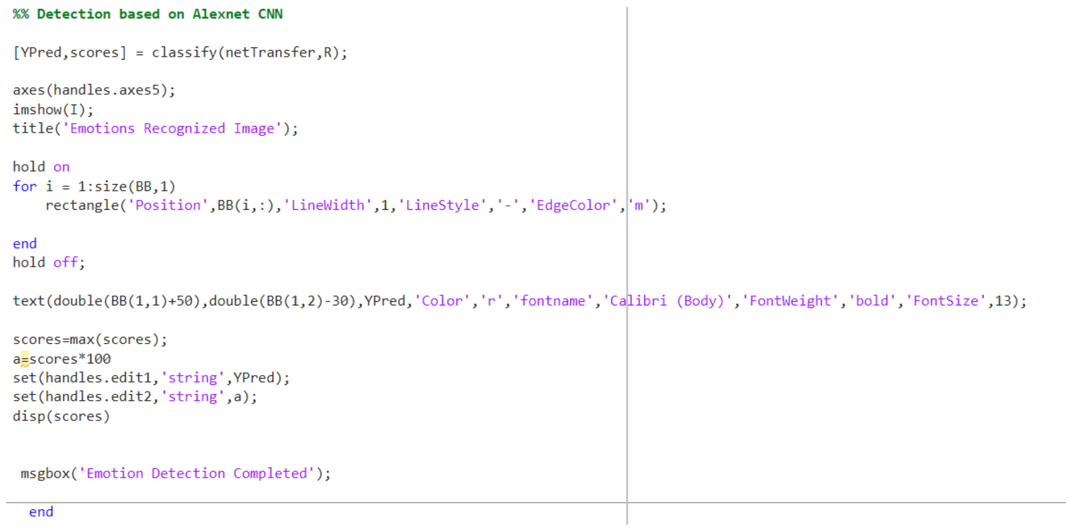


Figure 4.10: Face Cropping

4.2.6. Emotion detection

For emotion detection we used classify. This classifies used data using trained deep learning neural network. Classify(sample,training,group) classifies each row of the data in sample into one of the groups to which the data in training belongs. The groups for training are specified by group. The function returns class, which contains the assigned groups for each row of sample.



```
%% Detection based on Alexnet CNN

[YPred,scores] = classify(netTransfer,R);

axes(handles.axes5);
imshow(I);
title('Emotions Recognized Image');

hold on
for i = 1:size(BB,1)
    rectangle('Position',BB(i,:),'LineWidth',1,'LineStyle','-','EdgeColor','m');

end
hold off;

text(double(BB(1,1)+50),double(BB(1,2)-30),YPred,'Color','r','fontname','Calibri (Body)','FontWeight','bold','FontSize',13);

scores=max(scores);
a=scores*100
set(handles.edit1,'string',YPred);
set(handles.edit2,'string',a);
disp(scores)


 msgbox('Emotion Detection Completed');

  end
```

Figure 4.11: Code for Emotion Detection

CHAPTER - 5

SOFTWARE TOOLS

5.1. Introduction To MATLAB

The name MATLAB stands for Matrix Laboratory. MATLAB was written originally to provide easy access to matrix software developed by the LINPACK (linear system package) and EISPACK (Eigen system package) projects. MATLAB is a high-performance language for technical computing. It integrates computation, visualization, and programming environment. MATLAB has many advantages compared to conventional computer languages (e.g., C, FORTRAN) for solving technical problems. MATLAB is an interactive system whose basic data element is an array that does not require dimensioning. Specific applications are collected in packages referred to as toolbox. There are toolboxes for signal processing, symbolic computation, control theory, simulation, optimization, and several other fields of applied science and engineering.

5.2. MATLAB's Power of Computational Mathematics

MATLAB is used in every facet of computational mathematics. Following are some commonly used mathematical calculations where it is used most commonly:

• Dealing with Matrices and Arrays
• 2-D and 3-D Plotting and graphics
• Linear Algebra
• Algebraic Equations
• Non-linear Functions
• Statistics
• Data Analysis
• Calculus and Differential Equations Numerical Calculations
• Integration
• Transforms
• Curve Fitting

5.3. Features of MATLAB

The following are the basic features of MATLAB.

It is a high-level language for numerical computation, visualization, 5and application development.

• It also provides an interactive environment for iterative exploration, design and problem solving.

• It provides vast library of mathematical functions for linear algebra, statistics, Fourier analysis, filtering, optimization, numerical integration and solving ordinary differential equations.

• It provides built-in graphics for visualizing data and tools for creating custom plots.

• MATLAB's programming interface gives development tools for improving code quality, maintainability, and maximizing performance.

• It provides tools for building applications with custom graphical interfaces.

• It provides functions for integrating MATLAB based algorithms with external applications and languages such as C, Java, .NET and Microsoft Excel.

5.4. Uses of MATLAB

MATLAB is widely used as a computational tool in science and engineering encompassing the fields of physics, chemistry, math and all engineering streams. It is used in a range of applications including:

• signal processing and Communications

• image and video Processing

• control systems

• test and measurement

• computational finance

• computational biology

5.5. MATLAB Simulink

Simulink is a simulation and model-based design environment for dynamic and embedded systems, integrated with MATLAB. Simulink, also developed by MathWorks, is a data flow graphical programming language tool for modelling, simulating and analysing multi-domain dynamic systems. It is basically a graphical block diagramming tool with customizable set of block libraries.

It allows you to incorporate MATLAB algorithms into models as well as export the simulation results into MATLAB for further analysis.

Simulink supports −

- system-level design
- simulation
- automatic code generation
- testing and verification of embedded systems

There are several other add-on products provided by MathWorks and third-party hardware and software products that are available for use with Simulink.

The following list gives brief description of some of them −

- **Stateflow** allows developing state machines and flow charts.
- **Simulink Coder** allows the generation of C source code for real-time implementation of systems automatically.
- **xPC Target** together with **x86-based real-time systems** provide an environment to simulate and test Simulink and Stateflow models in real-time on the physical system.
- **Embedded Coder** supports specific embedded targets.
- **HDL Coder** allows to automatically generate synthesizable VHDL and Verilog.
- **SimEvents** provides a library of graphical building blocks for modelling queuing systems.

Chapter - 6
Results Explanation

6.1. Training results

Before the training it shows the random sample images from the given dataset.by using this process we can recheck the dataset we inserted correctly or not. This is indicating like the given below.

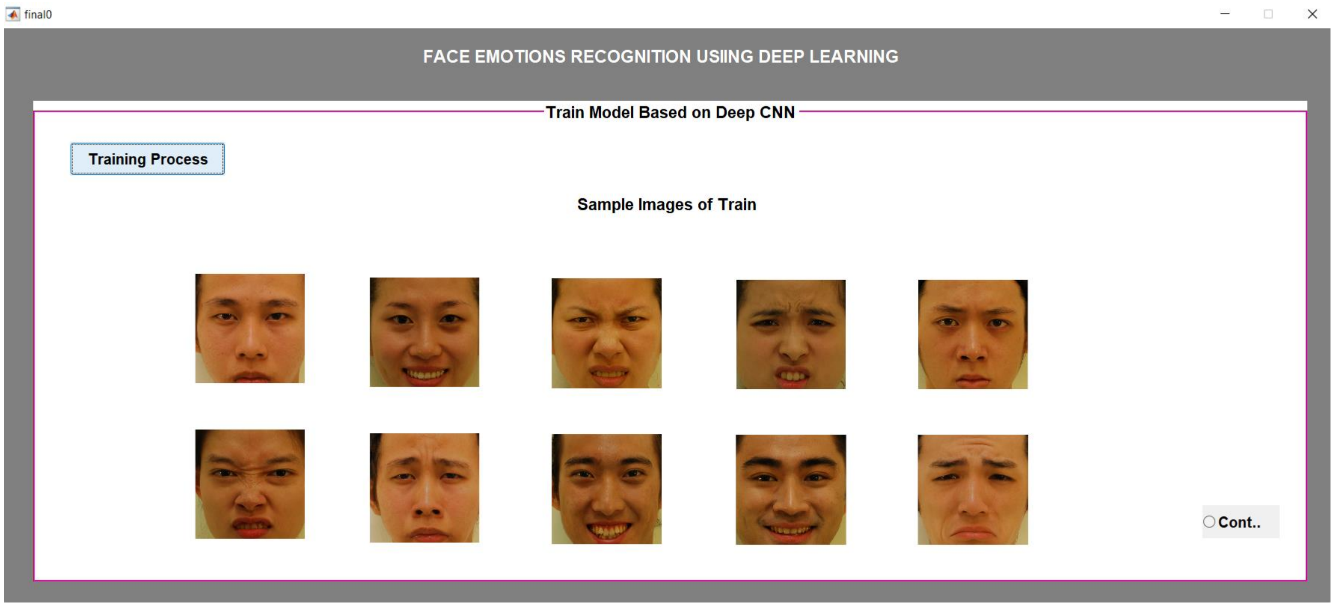
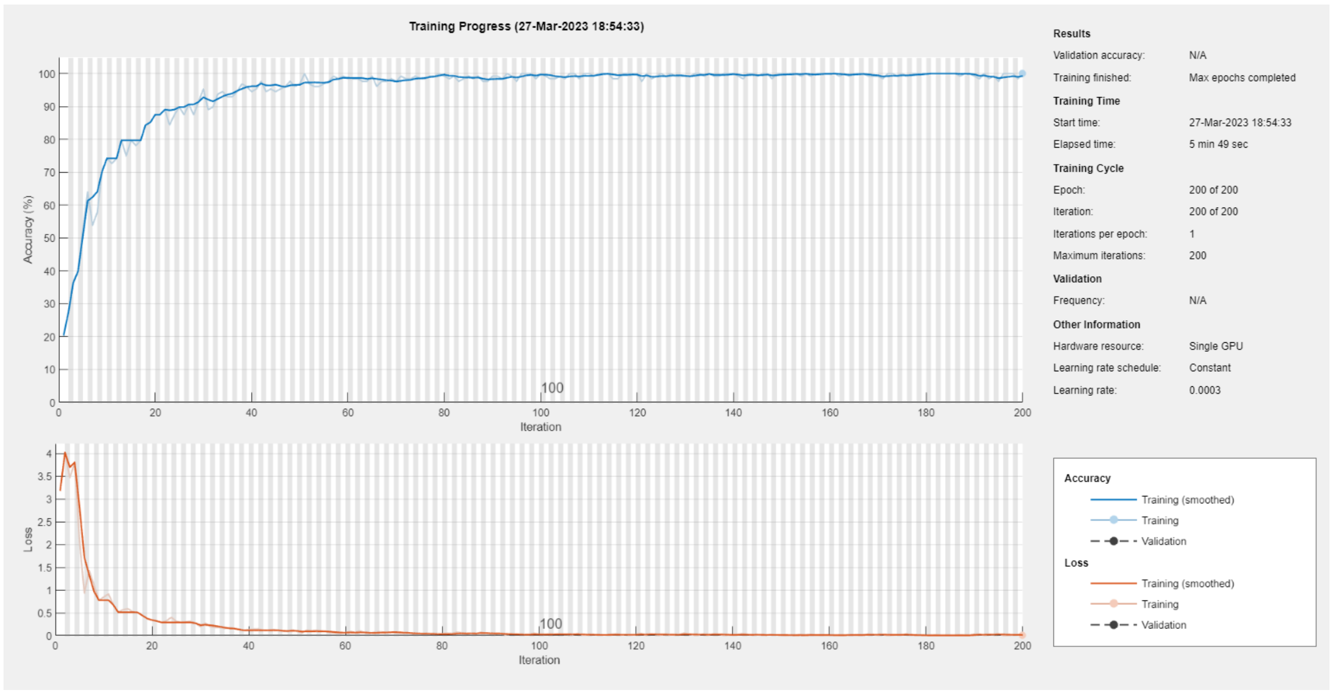

final0
FACE EMOTIONS RECOGNITION USIING DEEP LEARNING
Train Model Based on Deep CNN
Training Process
Sample Images of Train
Cont..

Figure 6.1: Getting Random Images

After getting the sample images of the training dataset then the actual training process starts by clicking the continues button that is indicated on the right side which is seen in the above image. In this training process starts iterating each and every image in the dataset. Then it indicates the total accuracy and loss of the total images in that dataset in the form of graphs. And it also indicates the starting time and ending time of the total training process.in that way we know that how much time takes for total training process. The graphical representation of accuracy and loss is shown in the below figure.



Results

Validation accuracy:	N/A
Training finished:	Max epochs completed

Training Time

Start time:	27-Mar-2023 18:54:33
Elapsed time:	5 min 49 sec

Training Cycle

Epoch:	200 of 200
Iteration:	200 of 200
Iterations per epoch:	1
Maximum iterations:	200

Validation

Frequency:	N/A

Other Information

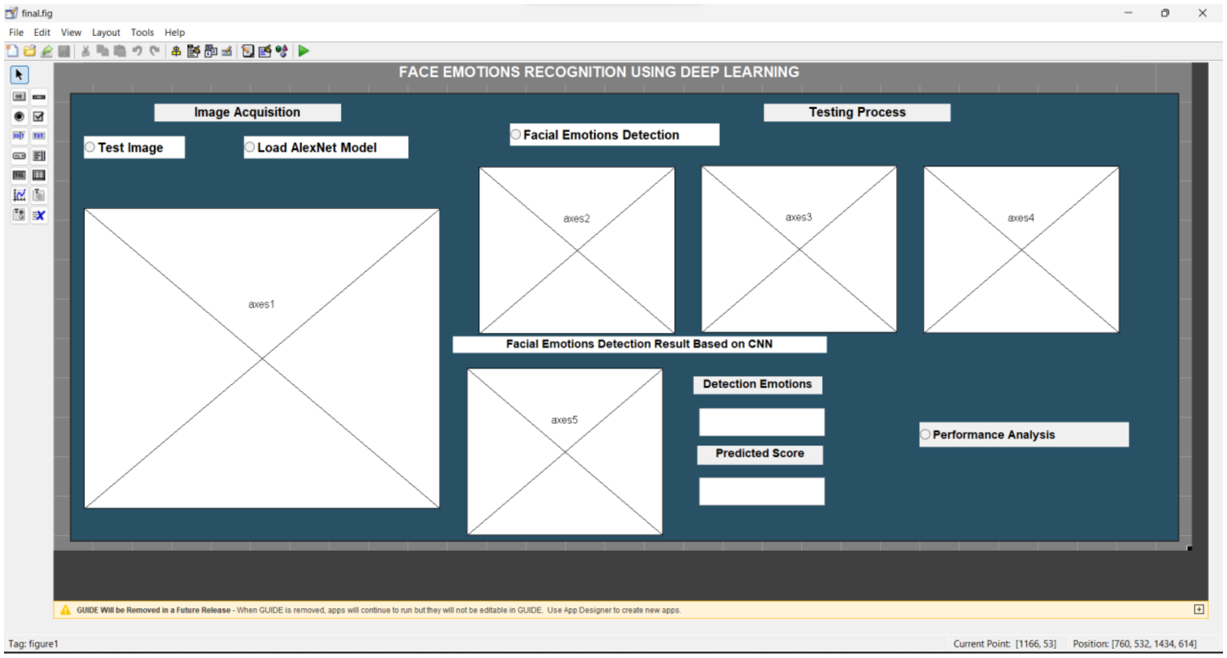
Hardware resource:	Single GPU
Learning rate schedule:	Constant
Learning rate:	0.0003

Figure 6.2: Training Output

The final output of the training is saved in MATLAB file extension known as mat file (.m file)

6.2. Testing result

In this testing process we use graphical user interface which helps for better understanding purpose. This GUI consists of radio buttons and axes. The axes are used for the image representation and the radio button is used for starting and stop by step format. Here we create our own GUI image which is shown in given below.


final.fig
File Edit View Layout Tools Help
FACE EMOTIONS RECOGNITION USING DEEP LEARNING
Image Acquisition
Test Image
Load AlexNet Model
Testing Process
Facial Emotions Detection
axes1
axes2
axes3
axes4
Facial Emotions Detection Result Based on CNN
axes5
Detection Emotions
Predicted Score
Performance Analysis
Tag: figure1
Current Point: [1166, 53]
Position: [760, 532, 1434, 614]

GUIDE Will be Removed in a Future Release - When GUIDE is removed, apps will continue to run but they will not be editable in GUIDE. Use App Designer to create new apps.

Figure 6.3: Graphical User Interface Image

In this testing process we use three radio buttons that

1. Test image
2. Load alexnet model
3. Facial emotion detection
4. Performance analysis

By clicking the first radio button it will open a pop-up window where we can select the test image.

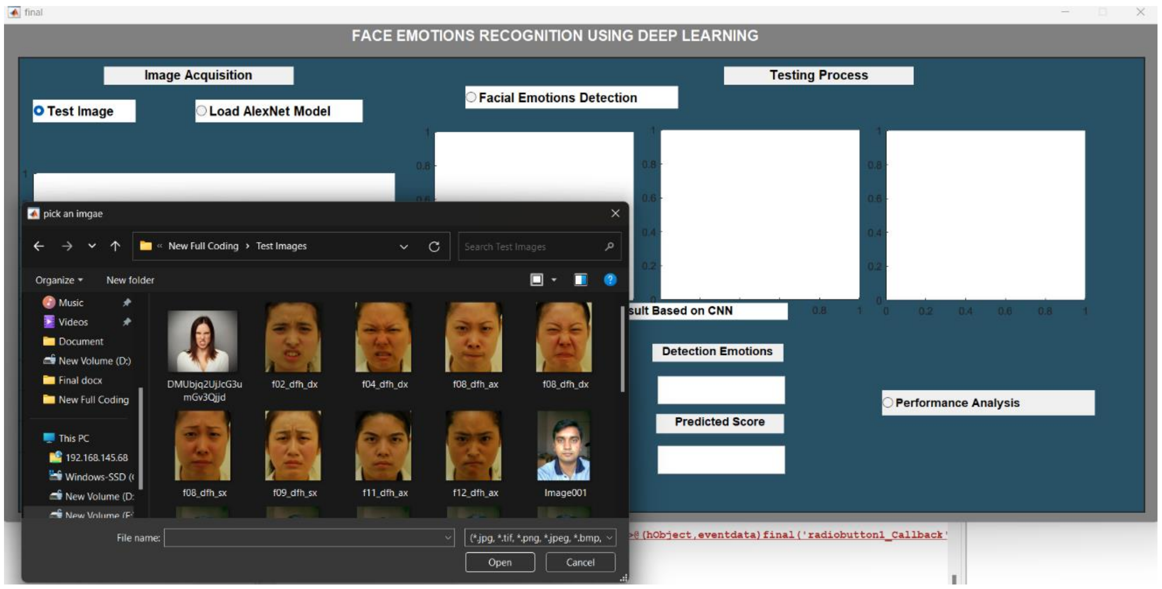


Figure 6.4: Selecting the Image.

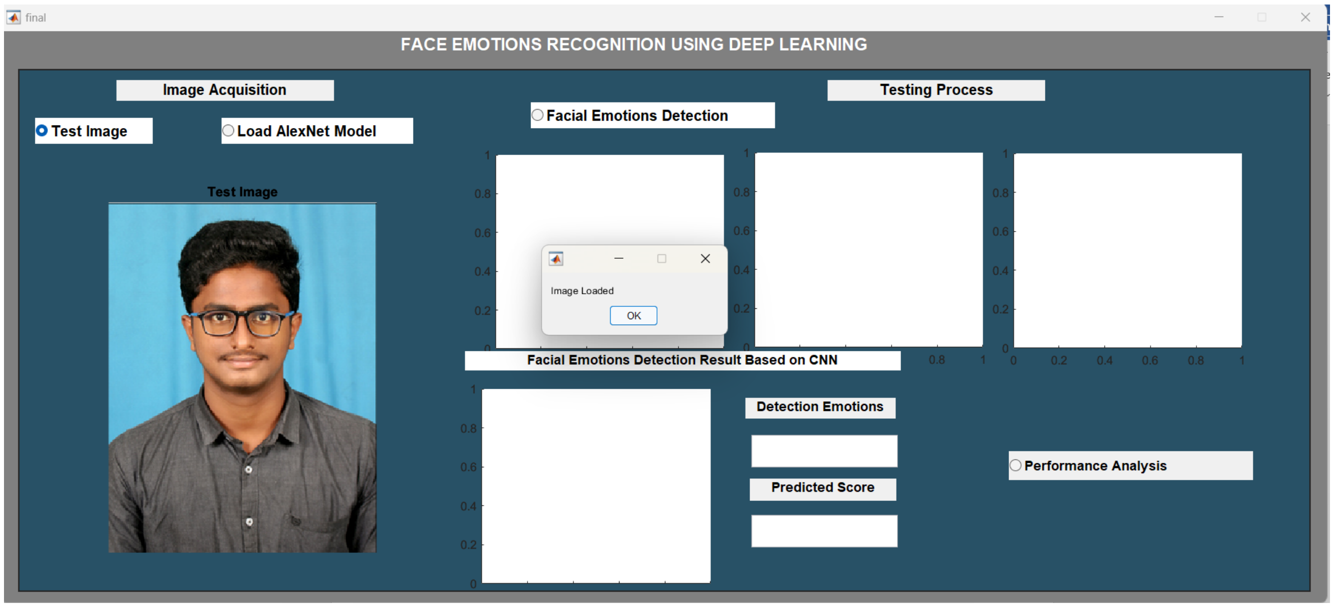


Image Loaded

Figure 6.5: Message of Image Loaded

After the image is loaded it gets a pop-up message window that have image is loaded.

In the second step we load the previously saved matlab file by clicking the second radio button named load alexnet model. It will directly load the trained file. After completion of the loading process, it gives the message box.

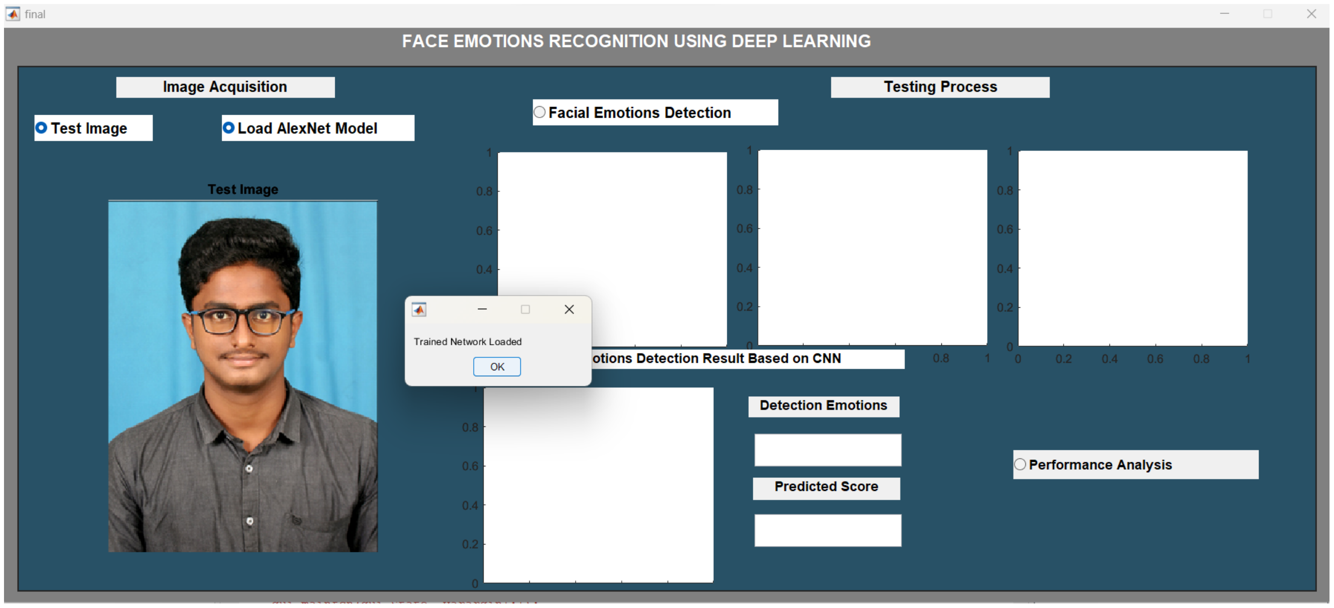


Figure 6.6: Loaded the Trained Network

After loading the previous network. The third radio button plays a key role in our project that is to detect emotion. This process explanation is already given previously.

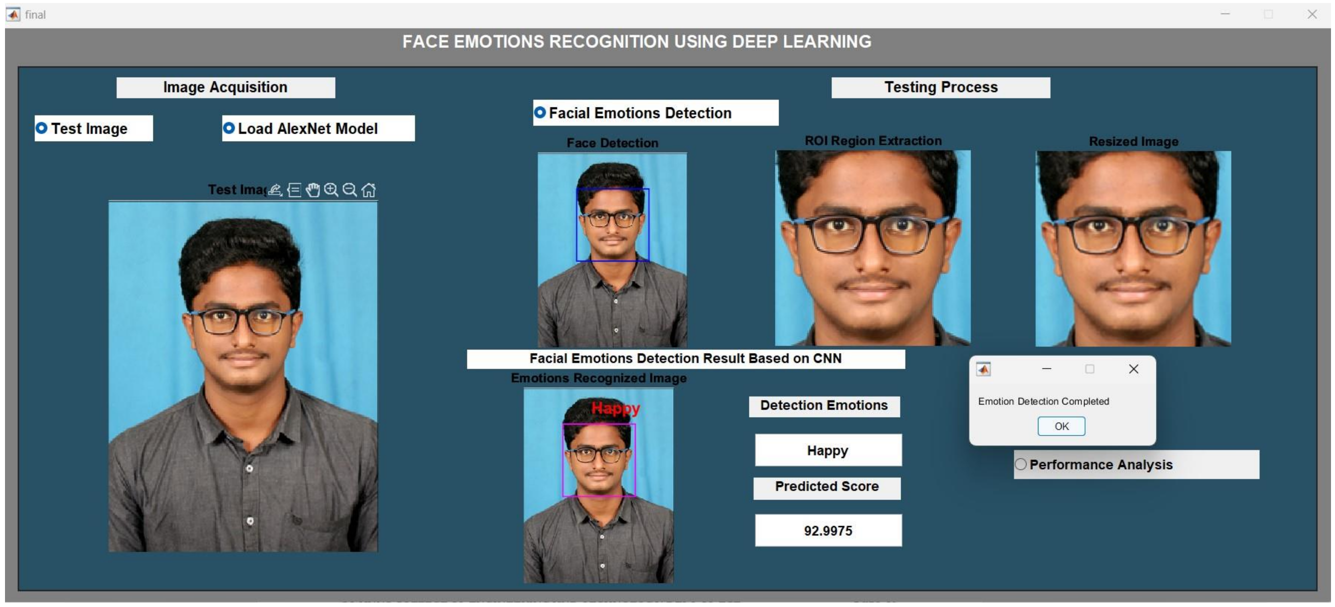
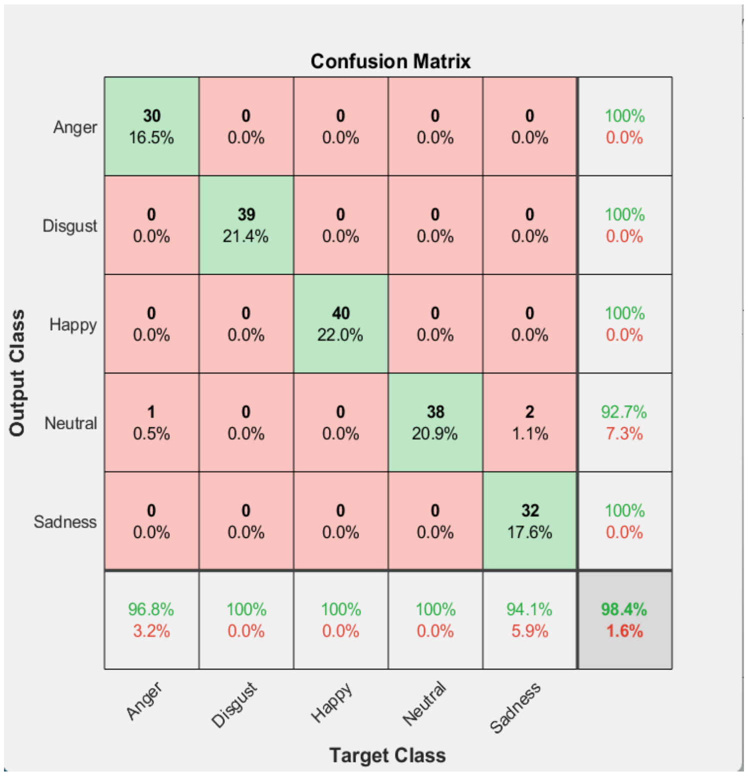


Figure 6.7: Emotion Detection

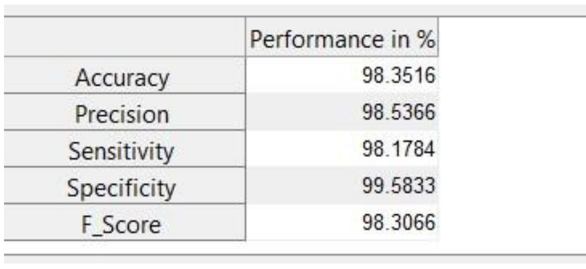
By the use of the third radio button, we detect the emotion and also know the accuracy of the detection in the name of predicted score and it notify the detection by the help of message which is shown in the above image.

In the fourth radio button we know the overall performance analysis of the project which is indicated in the form of confusion matrix by the help of confusion matrix we calculate accuracy, precision, sensitivity, specificity, and F-score by the help of this we may get the total performance of the dataset.The are shown in the below figures.



Confusion Matrix

Output Class \ Target Class	Anger	Disgust	Happy	Neutral	Sadness	
Anger	**30** 16.5%	**0** 0.0%	**0** 0.0%	**0** 0.0%	**0** 0.0%	100% 0.0%
Disgust	**0** 0.0%	**39** 21.4%	**0** 0.0%	**0** 0.0%	**0** 0.0%	100% 0.0%
Happy	**0** 0.0%	**0** 0.0%	**40** 22.0%	**0** 0.0%	**0** 0.0%	100% 0.0%
Neutral	**1** 0.5%	**0** 0.0%	**0** 0.0%	**38** 20.9%	**2** 1.1%	92.7% 7.3%
Sadness	**0** 0.0%	**0** 0.0%	**0** 0.0%	**0** 0.0%	**32** 17.6%	100% 0.0%
	96.8% 3.2%	100% 0.0%	100% 0.0%	100% 0.0%	94.1% 5.9%	**98.4%** **1.6%**



	Performance in %
Accuracy	98.3516
Precision	98.5366
Sensitivity	98.1784
Specificity	99.5833
F_Score	98.3066

Figure 6.8: Performance Analysis

Chapter - 7
CONCLUSION AND FUTURE SCOPE

7.1 Conclusion

In this project, we have proposed a deep learning based facial emotion detection method from image. We discuss our proposed model using Taiwanese Facial Expression Image Database (TFEID). The performance evaluation of the proposed facial emotion detection model is carried out in terms of validation accuracy, precision, sensitivity, specificity and F- score. We analyzed our proposed model using trained and test sample images, and evaluate their performance compared to previous existing model. Results of the experiment show that the model proposed is better in terms of the results of emotion detection to previous models reported in the literature. The experiments show that the proposed model is producing state-of-the-art effects on this dataset.

7.2 Future scope

Our aim in this work were to go in detail in every step of Deep Learning, starting from finding the dataset and ending up analyzing the results. We separated the work into two main parts: Dataset and the construction of the CNN. As a future work we would suggest using the same dataset but trying a different approach by taking into consideration “Action Units” to detect as features the movement of the muscles of the face and then to feed the CNN. To have fast results and to perform many experiments with different parameters, a machine with proper parameters (especially GPU) is recommended.

Chapter - 8
References

1)S. Li and W. Deng, “Deep facial expression recognition: A survey,” arXiv preprint arXiv:1804.08348, 2018

2)E. Correa, A. Jonker, M.Ozo, andR.Stolk, “Emotion recognition using deep convolutional neural networks,” Tech. Report IN4015, 2016.

3)Y. I. Tian, T. Kanade, and J.F. Cohn, “Recognizing action units for facial expression analysis,” IEEE Transactions on pattern analysis and machine intelligence, vol. 23, no. 2, pp. 97–115, 2001.

4)C. R. Darwin. The expression of the emotions in man and animals. John Murray, London, 1872.

5)Kaggle. Challenges in representation learning: Facial expression recognition challenge, 2013.

6)P. Lucey, J. F. Cohn, T. Kanade, J. Saragih, Z. Ambadar, and I. Matthews. The extended cohn-kanade dataset (ck+): A complete dataset for action unit and emotion-specified expression. In Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on, pages 94–101. IEEE, 2010

7)P. J. Werbos et al., “Backpropagation through time: what it does and how to do it,” Proceedings of the IEEE, vol. 78, no. 10, pp. 1550–1560, 1990.

8)Y. Lv, Z. Feng, and C. Xu. Facial expression recognition via deep learning. In Smart Computing (SMARTCOMP), 2014 International Conference on, pages 303–308. IEEE, 2014

9)A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems, pages 1097–1105, 2012

10)J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on, pages 248–255. IEEE, 2009

11)P. Ekman and W. V. Friesen. Constants across cultures in the face and emotion. Journal of personality and social psychology, 17(2):124, 1971.

12) J. Nicholson, K. Takahashi, and R. Nakatsu. Emotion recognition in speech using neural networks. Neural computing applications, 9(4): 290–296, 2000.

13) B. Fasel and J. Luettin. Automatic facial expression analysis: a survey. Pattern recognition, 36(1):259–275, 2003.

14) A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images, 2009.

15)] TFlearn. Tflearn: Deep learning library featuring a higher-level api for tensorflow.

16) Open Source Computer Vision Face detection using haar cascades.

17) R.M. Bell and Y. Koren. Lessons from the netflix prize challenge. ACM SIGKDD Explorations Newsletter, 9(2):75–79, 2007.

18) A. Berg, J. Deng, and L. Fei-Fei. Large scale visual recognition challenge 2010.

19) L. Breiman. Random forests. Machine learning, 45(1):5–32, 2001.

20) D. Cire¸san, U. Meier, and J. Schmidhuber. Multi-column deep neural networks for image classification. Arxiv preprint arXiv:1202.2745, 2012.

21) D.C. Cire¸san, U. Meier, J. Masci, L.M. Gambardella, and J. Schmidhuber. High-performance neural networks for visual object classification. Arxiv preprint arXiv:1102.0183, 2011.

22) J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In CVPR09, 2009.

23) J. Deng, A. Berg, S. Satheesh, H. Su, A. Khosla, and L. Fei-Fei. ILSVRC-2012, 2012. URL

24) A. Krizhevsky. Convolutional deep belief networks on cifar-10. Unpublished manuscript, 2010.

25) A. Krizhevsky and G.E. Hinton. Using very deep autoencoders for content-based image retrieval. In ESANN, 2011.x

Personal Profile

Name	:	Upputuri.Sairam
Father Name	:	Upputuri.Lakshmi Narayana
Class	:	B.Tech
Branch	:	Electronics and Communication Engineering
Roll No	:	19F01A04N6
Date Of Birth	:	11-07-2003
Marital Status	:	Unmarried
Mobile No	:	6281304637
Email Id	:	sairamupputuri@gmail.com
Permanent Address	:	Ramakrishna Puram
		Chirala
		Bapatla Dist