

AN OVERVIEW OF FACE RECOGNITION AND IT'S CHALLENGES USING CONVOLUTION NEURAL NETWORK.

Submitted in partial fulfillment of the requirements for the award
of Bachelor of Engineering Degree in Electronics and
Communication Engineering

by

**A.RAGHU SESHU(37130033)
ANWAR SADIQ (37130032)**



SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

Accredited with Grade "A" by NAAC

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI - 600 119

MARCH 2021



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)



Accredited with Grade “A” by NAAC

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI - 600
119

www.sathyabama.ac.in

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this project Report is the bonafide work of **A . RAGHU SESHU (37130033)** and **ANWAR SADIQ (37130032)** who carried out the project entitled “**AN OVERVIEW OF FACE RECOGNITION AND IT’S CHALLENGES USING CONVOLUTION NEURAL NETWORK.**” under our supervision from October 2020 to March 2021.

Internal Guide

M.A.MUTHIAH

Department of ECE

Head of the Department

Dr. T. RAVI, M.E., Ph.D.

Submitted for Viva voce examination held on _____

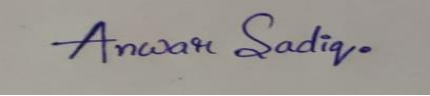
Internal Examiner

External Examiner

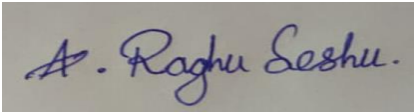
DECLARATION

We, A. RAGHU SESHU (37130033) and ANWAR SADIQ (37130032) hereby declare that the Project Report entitled “**AN OVERVIEW OF FACE RECOGNITION AND IT’S CHALLENGES USING CONVOLUTION NEURAL NETWORK.**” done by us under the guidance of M.A.MUTHIAH submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Electronics and Communication Engineering.

1)

A rectangular box containing a handwritten signature in blue ink that reads "Anwar Sadiq".

2)

A rectangular box containing a handwritten signature in blue ink that reads "A. Raghu Seshu".

DATE:

PLACE:

SIGNATURE OF THE CANDIDATES

ACKNOWLEDGEMENT

We are pleased to acknowledge our sincere thanks to Board of Management of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. We are grateful to them.

We convey our thanks to **Dr. N. M. NANDHITHA, M.E., Ph.D. Dean, School of Electrical and Electronics Engineering** and **Dr. T. RAVI, M.E., Ph.D. Head of the Department, Department of Electronics and Communication Engineering** for providing us necessary support and details at the right time during the progressive reviews.

We would like to express our sincere and deep sense of gratitude to our Project Guide **M.A MUTHIAH Department of Electronics and Communication Engineering** for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of the project work.

We wish to express our thanks to all teaching and Non-teaching staff members of the **Department of Electronics and Communication Engineering** who were helpful in many ways for the completion of the project.

We express our gratitude to our parents for their constant encouragement and support for the completion of the project.

ABSTRACT

Convolution Neural network is an algorithm widely used in Face Recognition in that architecture like alex net , dense net , google net , resnet ,squeeze net, vgg 16 and so many more are been used. This paper proposes a overview of face recognition of Convolution Neural Network (CNN) with some of famous architecture like alex net, dense net , google net , resnet ,squeeze net ,vgg 16 with same parameters and datasets are been tested with all the architecture .The number of iterations and the efficiency of the architecture show in the result in the proper manner , Datasets and parameters same are used for all architecture .The proposed system has designed by MATLAB Software of the version MATLAB 8.3 R2014a

TABLE OF CONTENTS

TITLE	PAGE NO
ABSTRACT	V
LIST OF FIGURES	IX
LIST OF ABBREVIATIONS	X

CHAPTER NO	TITLE	PAGE NO
1	INTRODUCTION	
	1.1 FACE RECOGNIZATION	1
	1.2 FACE DETECTION	2
	1.3 CONVOLUTION NEURAL NETWORK	3
2	LITERATURE SURVEY	
	2.1 LITERATURE REVIEW	7
3	AIM AND SCOPE OF THE PRESENT INVESTIGATION	
	3.1 ALEX NET	11
	3.2 DENSE NET	13
	3.3 GOOGLE NET	13
	3.4 RES NET	15
	3.5 SQUEEZE NET	16
	3.6 VGG NET	17
4	MATERIALS AND METHODS USED	
	4.1 MATLAB	18

4.1.1	MATLAB POWER OF COMPUTATIONAL MATHEMATICS.	18
4.1.2	FEATURES OF MATLAB	19
4.1.3	USES OF MATLAB	19
4.1.4	UNDERSTANDING THE MATLAB ENVIRONMENT	20
4.1.5	COMMONLY USED OPERATORS	22
4.1.6	COMMANDS	23
4.1.7	.M FILES	24
4.1.8	DATA TYPES	25
4.2	MICROSOFT WINDOWS	
4.2.1	MULTIPLE DESKTOP AND TASK VIEW	27
4.2.1.1	ACTION CENTER	27
4.2.1.2	TABLET MODE	28
4.2.1.3	SECURITY	28
5	RESULT AND DISCUSSION	
5.1	RESULT OF ALEX NET	31
5.2	RESULT OF DENSE NET	32
5.3	RESULT OF GOOGLE NET	34
5.4	RESULT OF RES NET	35
5.5	RESULT OF SQUEEZE NET	36

	5.6 RESULT OF VGG NET	37
6	CONCLUSION	38
7	References	44

LIST OF FIGURES

FIGURE NO.	TITLE	PAGE NO.
2	Photometric stereo image	2
3	Geometric facial recognition	2
4	Face detection algorithm	3
5	CNN architecture	3
6	Detail about CNN layer	4
7	CNN layers	6
8	AlexNet Architecture	12
9	GoogLeNet Architecture	14
10	RESSNet Architecture	15
11	Squeeze Architecture	16
12	Vgg16 Architecture	17
13	Matlab desktop environment	20
14	Current folder of Matlab	20
15	Command Window of Matlab	21
16	Workspace of Matlab	21
17	Command History	22
18	AlexNet Result	31
19	DenseNet Result	32
20	GoogLeNet Result	34
21	RESSNet Result	35
22	Squeeze Result	36
23	Vgg 16 Result	37

LIST OF ABBREVIATIONS

CNN	CONVOLUTION NEURAL NETWORK
PCA	PRINCIPAL COMPONENT ANALYSIS
ANN	ARTIFICIAL NEURAL NETWORK
ReLU	RECTIFYING LINEAR UNIT
FIP	FACE INDENTITY PRESERVING
LFW	LABELED FACE WILD
YTF	YOUTUBE FACE
NIN	NETWORK IN NETWORK
GUI	GRAPHICAL USER INTERFACE
VGG	VISUAL GEOMETRY GROUP
ResNet	RESIDUAL NEURAL NETWORK
IDE	INTEGRATED DEVELOPMENT ENVIRONMENT
MATLAB	MATRIX LABORATORY
LINPACK	LINER SYSTEM PACKAGE
EISPACK	EIGEN SYSTEM PACKAGE
LBP	LOCAL BINARY PATTERN
GTX	GIGA TEXEL SHADER EXTREME

CHAPTER-1

INTRODUCTION

Face recognition is the task of identifying an already detected object as a known or unknown face. Often the problem of face recognition is confused with the problem of face detection. Face Recognition on the other hand is to decide if the "face" is someone known, or unknown, using for this purpose a database of faces in order to validate this input face.

1.1 FACE RECOGNITION:

DIFFERENT APPROACHES OF FACE RECOGNITION:

There are two predominant approaches to the face recognition problem: Geometric (feature based) and photometric (view based). As researcher interest in face recognition continued, many different algorithms were developed, three of which have been well studied in face recognition literature.

Recognition algorithms can be divided into two main approaches:

1. **Geometric:** Is based on geometrical relationship between facial landmarks, or in other words the spatial configuration of facial features. That means that the main geometrical features of the face such as the eyes, nose and mouth are first located and then faces are classified on the basis of various geometrical distances and angles between features. (Figure 2)
2. **Photometric stereo:** Used to recover the shape of an object from a number of images taken under different lighting conditions. The shape of the recovered object is defined by a gradient map, which is made up of an array of surface normals (Zhao and Chellappa, 2006) (Figure 3)

Popular recognition algorithms include:

1. Principal Component Analysis using Eigenfaces, (PCA)
2. Linear Discriminate Analysis,
3. Elastic Bunch Graph Matching using the Fisherface algorithm,

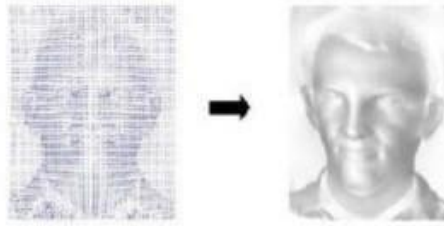


Figure 2 -Photometric stereo image.

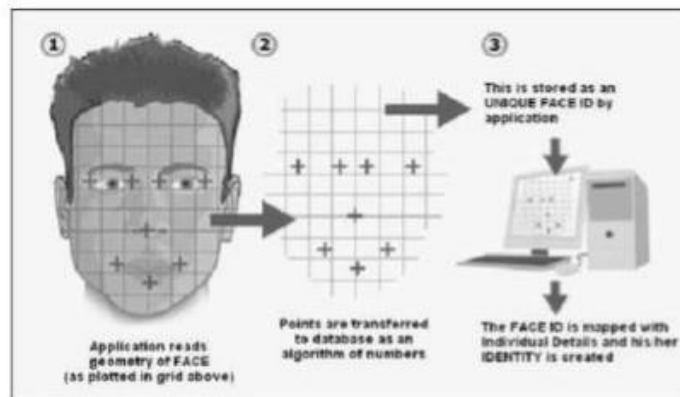


Figure 3 -Geometric facial recognition.

1.2 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

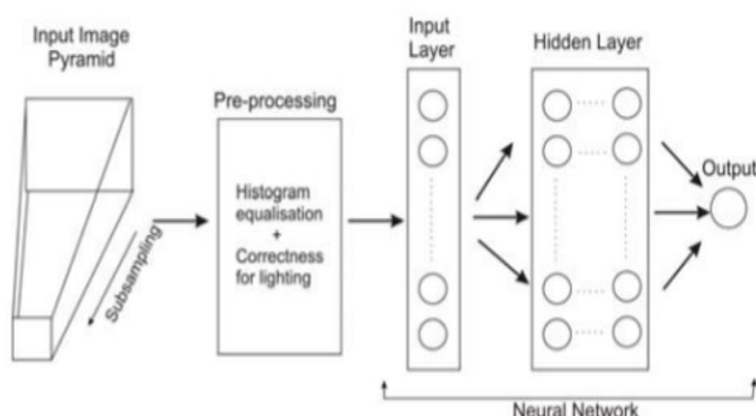


Fig: Face detection algorithm

Figure 4

1.3 CONVOLUTIONAL NETURAL NETWORK

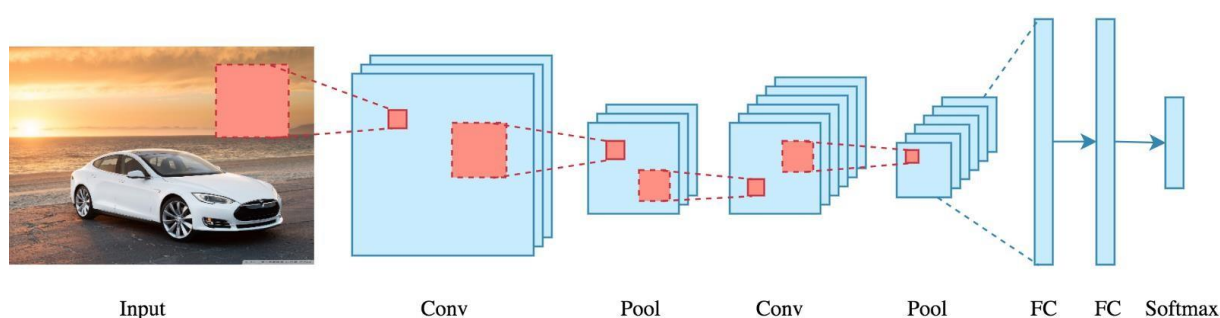


Figure 5: CNN architecture

Convolutional neural network (CNN) is a type of neural network architecture specially made to deal with visual data. In this article we will discuss the architecture of CNN and implement it on CIFAR-10 dataset in part-2. The main benefit of using a CNN over simple ANN on visual data is that CNN's are constrained to deal with image data exclusively. Two main features of CNNs are

1. Weight sharing

2. Feature extractors

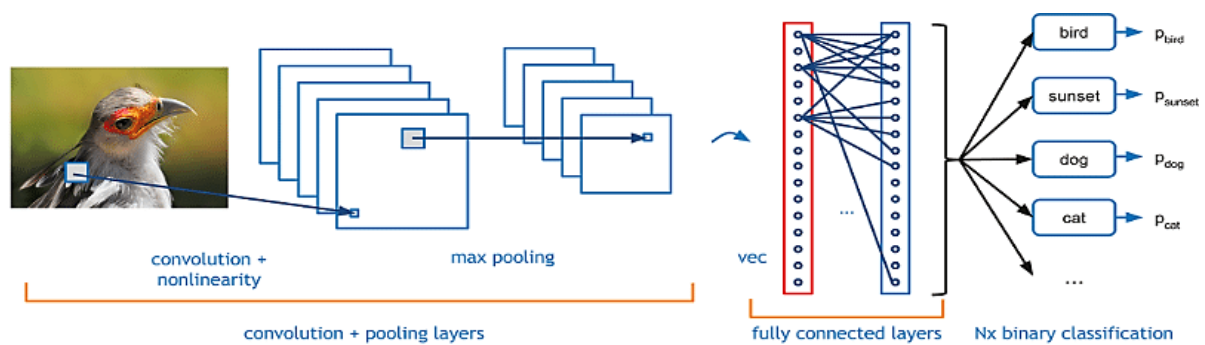


Figure 6 Detail about CNN layer

As we described above, a simple ConvNet is a sequence of layers, and every layer of a ConvNet transforms one volume of activations to another through a differentiable function. We use three main types of layers to build ConvNet architectures: Convolutional Layer, Pooling Layer, and Fully-Connected Layer (exactly as seen in regular Neural Networks). We will stack these layers to form a full ConvNet architecture.

Several new layers are introduced in CNNs to extract the useful features from our image or reducing the size of image without using the original representation.

Convolution Layer -

Convolutional layer apply convolution operation on the input layer, passing the results to next layer. A convolution operation is basically computing a dot product between their weights and a small region they are connected (currently overlapping) to in the input volume. This will change the dimensions depending on the filter size used and number of filters used.

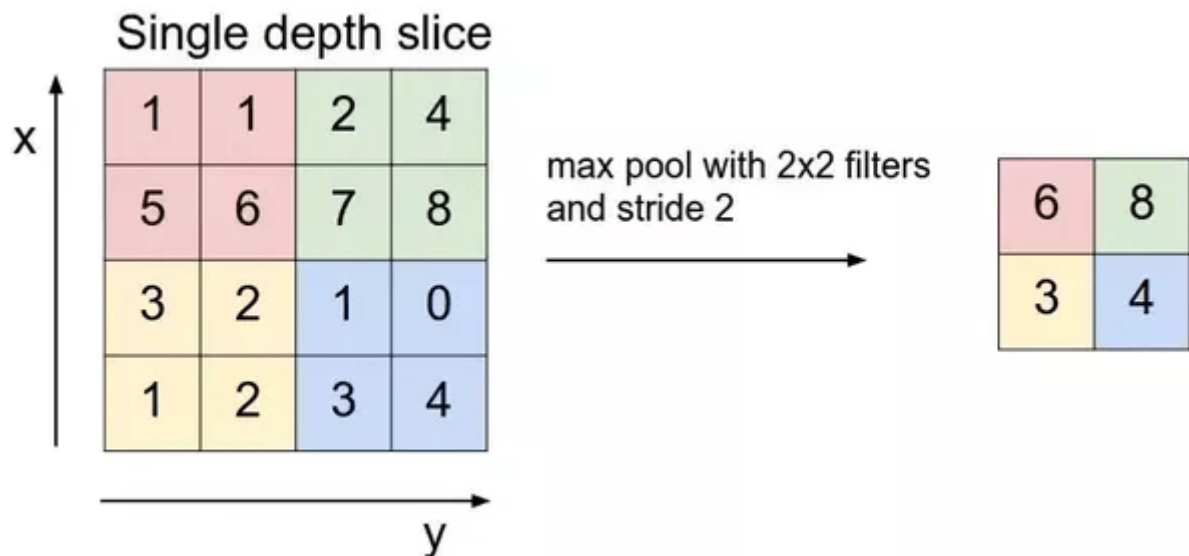
We can compute the spatial size of the output volume as a function of the input volume size (W), the receptive field size of the Conv Layer neurons (F), the stride with which they are applied (S), and the amount of zero padding used (P) on the border. You can convince yourself that the correct formula for calculating how many neurons "fit" is given by $(W - F + 2P) / S + 1$. For example for a 7×7 input and a 3×3 filter with stride 1 and pad 0 we would get a 5×5 output. With stride 2 we would get a 3×3 output.

ReLU Layer

Rectifying Linear Unit (ReLU) layer applies the relu activation element-wise. It is a mathematical function, which returns a positive value or 0 in place of previous negative values, It does not change the dimensions of the previous layer.

Pooling Layer

Pooling layer will perform a down-sampling operation along the width and resulting in the reduction of the dimensions. The sole purpose of pooling is to reduce spatial dimensions. There are various types of pooling in which the most common is Max Pooling, i.e taking the maximum element from the window.

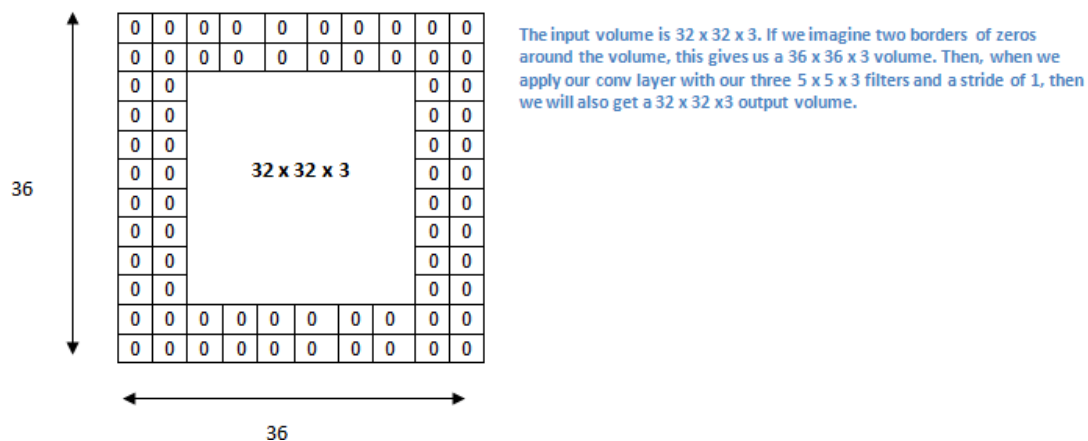


Stride

Stride decides by how much we move our window ,when we have a stride of one we move across and down a single pixel. With higher stride values, we move large number of pixels at a time and hence produce smaller output volumes.

Padding

Padding is used to preserve the boundary information , since without padding they are only traversed once.



Flattening Layer

This layer will convert the 3-dimensions (height,width,depth) into a single long vector to feed it to the fully connected layer or Dense layer. It connects every neuron in one layer to every neuron in another layer.

Fully Connected Layer and Output Layer

Fully connected layers or dense layers are the same hidden layers consisting of defined number of neurons connected with elements of another layer that we discussed in simple ANN. However the output layer is also the same but the number of neurons depend on our task. For instance in CIFAR-10 dataset we have 10 classes hence we will have 10 neurons in the outer layer.

Summary

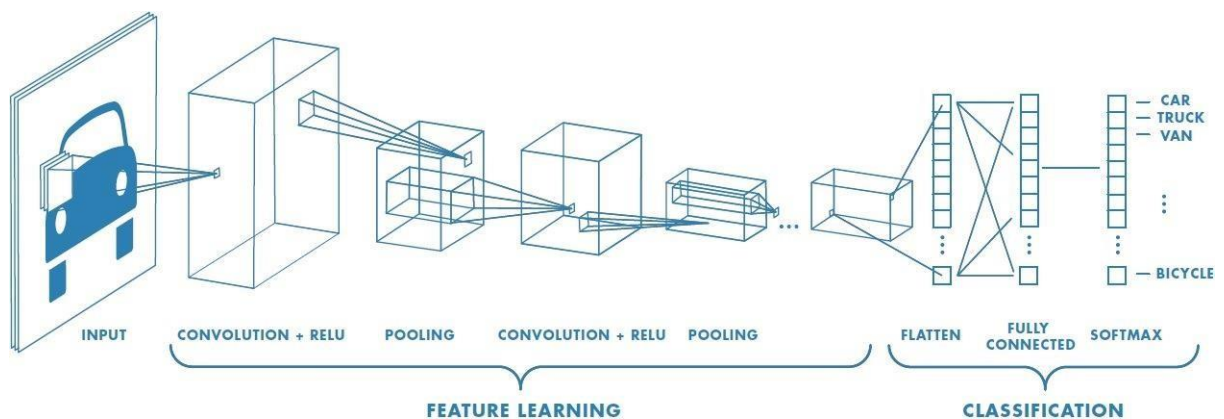


Figure 7 CNN LAYERS

In summary, the architecture of CNN, we can simply understand that it consists of an input layer followed by a Conv layer. The dimensions of the conv layer depend on the data and problem, hence changing the dimensions accordingly. After the Conv Layer, there is an activation layer, usually ReLU, since it gives better results. After some conv and ReLU combination, a pooling layer is used to reduce the size. Then, after some combination of the previously defined architecture, a flattening layer is used to flatten the input for the fully connected layer. Next to these layers, the last layer is the output layer.

CHAPTER 2

LITERATURE SURVEY

1. Face Identification with Deep Learning and Filter Algorithm

Author : Mohammed Alghaili, Hamdi A. R. Ali

Although significant advances have been made recently in the field of face recognition, these have some limitations, especially when faces are in different poses or have different levels of illumination, or when the face is blurred. In this study, we present a system that can directly identify an individual under all conditions by extracting the most important features and using them to identify a person. Our method uses a deep convolutional network that is trained to extract the most important features. A filter is then used to select the most significant of these features by finding features greater than zero, storing their indices, and comparing the features of other identities with the same indices as the original image. Finally, the selected features of each identity in the dataset are subtracted from features of the original image to find the minimum number that refers to that identity. This method gives good results, as we only extract the most important features using the filter to recognize the face in different poses. We achieve state-of-the-art face recognition performance using only half of the 128 bytes per face. The system has an accuracy of 99.27% on the Labeled Faces in the Wild dataset and 94.02% on YouTube Faces DB.

2. Deep learning Identity-preserving face

spaceAuthors: Z. Zhu, P. Luo, X. Wang, X.Tang

Face recognition with large pose and illumination variations is a challenging problem in computer vision. This paper addresses this challenge by proposing a new learning based face representation: the face identity-preserving (FIP) features. Unlike conventional face descriptors, the FIP features can significantly reduce intra-identity variances, while maintaining discriminative ness between identities. Moreover, the FIP features extracted from an image under any pose and illumination can be used to reconstruct its face image in the canonical view. This property makes it possible to improve the performance of traditional descriptors, such as LBP [2] and Gabor [31], which can be extracted from our reconstructed images in the canonical view to eliminate variations. In order to learn the FIP features, we carefully design a deep network that combines the feature extraction layers and the reconstruction layer. The former encodes a face image into the FIP features, while the latter transforms them to an image in the canonical view. Extensive experiments on the large MultiPIE face database [7] demonstrate that it significantly outperforms the state-of-the-art face recognition methods.

3. Hybrid deep learning for face verification

Authors: yi sun, xiaogang wang ,xiaouu tang

This paper proposes a hybrid convolutional network (ConvNet)-Restricted Boltzmann Machine (RBM) model for face verification. A key contribution of this work is to learn high-level relational visual features with rich identity similarity information. The deep ConvNets in our model start by extracting local relational visual features from two face images in

comparison, which are further processed through multiple layers to extract high-level and global relational features. To keep enough discriminative information, we use the last hidden layer neuron activations of the ConvNet as features for face verification instead of those of the output layer. To characterize face similarities from different aspects, we concatenate the features extracted from different face region pairs by different deep ConvNets. The resulting high-dimensional relational features are classified by an RBM for face verification. After pre-training each ConvNet and the RBM separately, the entire hybrid network is jointly optimized to further improve the accuracy. Various aspects of the ConvNet structures, relational features, and face verification classifiers are investigated. Our model achieves the state-of-the-art face verification performance on the challenging LFW dataset under both the unrestricted protocol and the setting when outside data is allowed to be used for training.

4. Sphereface : deep hypersphere embedding for face recognition

Authors: W. Liu, Y. Wen, Z. Yu et al, Ming li, Bhiksha Raj, Le song.

This paper addresses deep face recognition (FR) problem under open-set protocol, where ideal face features are expected to have smaller maximal intra-class distance than minimal inter-class distance under a suitably chosen metric space. However, few existing algorithms can effectively achieve this criterion. To this end, we propose the angular softmax (A-Softmax) loss that enables convolutional neural networks (CNNs) to learn angularly discriminative features. Geometrically, A-Softmax loss can be viewed as imposing discriminative constraints on a hypersphere manifold, which intrinsically matches the prior that faces also lie on a manifold. Moreover, the size of angular margin can be quantitatively adjusted by a parameter m . We further derive specific m to approximate the ideal feature criterion. Extensive analysis and experiments on Labeled Face in the Wild (LFW), Youtube Faces (YTF) and MegaFace Challenge show the superiority of A-Softmax loss in FR tasks. The code has also been made publicly available.

5. Deep learning face representation from predicting 10,000 classes

Authors: Y. Sun, X. Wang, X. Tang

This paper proposes to learn a set of high-level feature representations through deep learning, referred to as Deep hidden IDentity features (DeepID), for face verification. We argue that DeepID can be effectively learned through challenging multi-class face identification tasks, whilst they can be generalized to other tasks (such as verification) and new identities unseen in the training set. Moreover, the generalization capability of DeepID increases as more face classes are to be predicted at training. DeepID features are taken from the last hidden layer neuron activations of deep convolutional networks (ConvNets). When learned as classifiers to recognize about 10,000 face identities in the training set and configured to keep reducing the neuron numbers along the feature extraction hierarchy, these deep ConvNets gradually form compact identity-related features in the top layers with only a small number of hidden neurons. The proposed features are extracted from various face regions to form complementary and over-complete representations. Any state-of-the-art classifiers can be learned based on these high-level representations for face verification. 97.45% verification accuracy on LFW is achieved with only weakly aligned faces.

6. Deep Learning Face Representation by Joint Identification-Verification

Authors: Y. Sun, X. Wang, X. Tang

The key challenge of face recognition is to develop effective feature representations for reducing intra-personal variations while enlarging inter-personal differences. In this paper, we show that it can be well solved with deep learning and using both face identification and verification signals as supervision. The Deep IDentification-verification features (DeepID2) are learned with carefully designed deep convolutional networks. The face identification task increases the inter-personal variations by drawing DeepID2 extracted from different identities apart, while the face verification task reduces the intra-personal variations by pulling DeepID2 extracted from the same identity together, both of which are essential to face recognition. The learned DeepID2 features can be well generalized to new identities unseen in the training data. On the challenging LFW dataset, 99.15% face verification accuracy is achieved. Compared with the best deep learning result on LFW, the error rate has been significantly reduced by 67%.

7. Web-scale training for face identification

Authors: Y. Taigman, M. Yang, M. A. Ranzato, L. Wolf

Scaling machine learning methods to very large datasets has attracted considerable attention in recent years, thanks to easy access to ubiquitous sensing and data from the web. We study face recognition and show that three distinct properties have surprising effects on the transferability of deep convolutional networks (CNN): (1) The bottleneck of the network serves as an important transfer learning regularizer, and (2) in contrast to the common wisdom, performance saturation may exist in CNN's (as the number of training samples grows); we propose a solution for alleviating this by replacing the naive random subsampling of the training set with a bootstrapping process. Moreover, (3) we find a link between the representation norm and the ability to discriminate in a target domain, which sheds lights on how such network represent faces. Based on these discoveries, we are able to improve face recognition accuracy on the widely used LFW benchmark, both in the verification (1:1) and identification (1:N) protocols, and directly compare, for the first time, with the state of the art Commercially-Off-The-Shelf system and show a sizable leap in performance.

8. Deeply learned face representations are sparse, selective, and robust

Authors: Y. Sun, X. Wang, X. Tang.

This paper designs a high-performance deep convolutional network (DeepID2+) for face recognition. It is learned with the identification-verification supervisory signal. By increasing the dimension of hidden representations and adding supervision to early convolutional layers, DeepID2+ achieves new state-of-the-art on LFW and YouTube Faces benchmarks. Through empirical studies, we have discovered three properties of its deep neural activations critical for the high performance: sparsity, selectiveness and robustness. (1) It is observed that neural activations are moderately sparse. Moderate sparsity maximizes the discriminative power of the deep net as well as the distance between images. It is surprising that DeepID2+ still can achieve high recognition accuracy even after the neural responses are binarized. (2) Its neurons in higher layers are highly

selective to identities and identity-related attributes. We can identify different subsets of neurons which are either constantly excited or inhibited when different identities or attributes are present. Although DeepID2+ is not taught to distinguish attributes during training, it has implicitly learned such high-level concepts. (3) It is much more robust to occlusions, although occlusion patterns are not included in the training set.

9. Learning face representation from scratch

Authors: D. Yi, Z. Lei, S. Liao, S. Z. Li

Pushing by big data and deep convolutional neural network (CNN), the performance of face recognition is becoming comparable to human. Using private large scale training datasets, several groups achieve very high performance on LFW, i.e., 97% to 99%. While there are many open source implementations of CNN, none of large scale face dataset is publicly available. The current situation in the field of face recognition is that data is more important than algorithm. To solve this problem, this paper proposes a semi-automatic way to collect face images from Internet and builds a large scale dataset containing about 10,000 subjects and 500,000 images, called CASIAWebFace. Based on the database, we use a 11-layer CNN to learn discriminative representation and obtain state-of-the-art accuracy on LFW and YTF. The publication of CASIAWebFace will attract more research groups entering this field and accelerate the development of face recognition in the wild.

CHAPTER 3

AIM AND SCOPE OF PRESENT INVESTIGATION

3.1 AlexNet

LeNet (LeCun et al. 1995) though, begin the history of deep CNNs, but at that time, CNN was limited to hand digit recognition tasks and didn't perform well to all classes of images. AlexNet (Krizhevsky et al. 2012) is considered as the first deep CNN architecture, which showed groundbreaking results for image classification and recognition tasks. AlexNet was proposed by Krizhevsky et al., which enhanced the learning capacity of the CNN by making it deeper and by applying several parameter optimizations strategies (Krizhevsky et al. 2012). The basic architectural design of AlexNet is shown in Fig. 5. In early 2000, hardware limitations curtailed the learning capacity of deep CNN architectures by restricting them to small size. In order to get the benefit of the representational capacity of deep CNNs, Alexnet was trained in parallel on two NVIDIA GTX 580 GPUs to overcome shortcomings of the hardware. In AlexNet, depth was extended from 5 (LeNet) to 8 layers to make CNN applicable for diverse categories of images. Despite the fact that generally, depth improves generalization for different resolutions of images but, the main drawback associated with an increase in depth is overfitting. To address this challenge, Krizhevsky et al. (2012) exploited the idea of Hinton (Dahl et al. 2013; Srivastava et al. 2014), whereby their algorithm randomly skips some transformational units during training to enforce the model to learn more robust features. In addition to this, ReLU was employed as a non-saturating activation function to improve the convergence rate by alleviating the problem of vanishing gradient to some extent (Hochreiter 1998; Nair and Hinton 2010). Overlapping subsampling and local response normalization were also applied to improve the generalization by reducing overfitting. Other adjustments made were the use of large size filters (11x11 and 5x5) at the initial layers, compared to previously proposed networks. Due to the efficient learning approach of AlexNet, it has significant importance in the new generation of CNNs and has started a new era of research in the architectural advancements of CNNs.

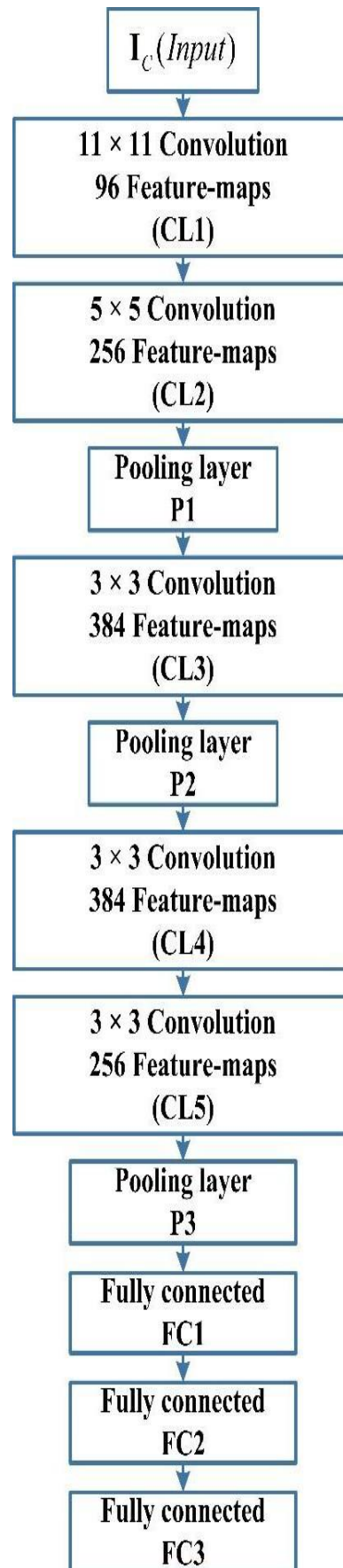


Fig. 8 Basic layout of AlexNet architecture showing its five convolution and three fully connected layers

3.2 DenseNet

Similar to Highway Networks and ResNet, DenseNet was proposed to solve the vanishing gradient problem (Srivastava et al. 2015a; He et al. 2015a; Huang et al. 2017). The problem with ResNet was that it explicitly preserves information through additive identity transformations due to which many layers may contribute very little or no information. To address this problem, DenseNet used cross-layer connectivity but, in a modified fashion. DenseNet connected each preceding layer to the next coming layer in a feed-forward fashion; thus, feature-maps of all previous layers were used as inputs into all subsequent layers as expressed in equation (10 & 11) Where the F_2^k and F_1^k are the resultant feature map of first and $i-1$ transformation layer and $gk(.)$ is a function which enables cross layer connection by concatenating proceeding layers information before assigning to new transformation layer l . this establishes $l(l+1)/2$ connections in DenseNet, as compared to l connections between a layer and its preceding layer in the traditional CNNs. It imprints the effect of cross-layer depthwise convolutions. As DenseNet concatenates the features of the previous layer instead of adding them, thus, the network may gain the ability to explicitly differentiate between information that is added to the network and information that is preserved. DenseNet has a narrow layer structure; however, it becomes parametrically expensive with an increase in a number of feature-maps. Information flow in the network improves by providing each layer direct access to the gradients through the loss function. Direct admittance to gradient incorporates a regularizing effect, which reduces overfitting on tasks with smaller training sets.

3.3 GoogleNet

GoogleNet was the winner of the 2014-ILSVRC competition and is also known as Inception-V1. The main objective of the GoogleNet architecture was to achieve high accuracy with a reduced computational cost (Szegedy et al. 2015). It introduced the new concept of inception block in CNN, whereby it incorporates multi-scale convolutional transformations using split, transform and merge idea. The architecture of the inception block is shown in Fig. 6. In GoogleNet, conventional convolutional layers are replaced in small blocks similar to the idea of substituting each layer with micro NN as proposed in Network in Network (NIN) architecture (Lin et al. 2013). This block encapsulates filters of different

sizes (1x1, 3x3, and 5x5) to capture spatial information at different scales, including both fine and coarse grain level. The exploitation of the idea of split, transform, and merge by GoogleNet, helped in addressing a problem related to the learning of diverse types of variations present in the same category of images having different resolutions. GoogleNet regulates the computations by adding a bottleneck layer of 1x1 convolutional filter, before employing large size kernels. In addition to it, it used sparse connections (not all the output feature-maps are connected to all the input feature-maps), to overcome the problem of redundant information and reduced cost by omitting feature-maps that were not relevant. Furthermore, connection's density was reduced by using global averagepooling at the last layer, instead of using a fully connected layer. These parameter tunings caused a significant decrease in the number of parameters from 138 million to 4 million parameters. Other regulatory factors applied were batch normalization and the use of RmsProp as an optimizer (Dauphin et al. 2015). GoogleNet also introduced the concept of auxiliary learners to speed up the convergence rate. However, the main drawback of the GoogleNet was its heterogeneous topology that needs to be customized from module to module. Another limitation of GoogleNet was a representation bottleneck that drastically reduces the feature space in the next layer and thus sometimes may lead to loss of useful information.

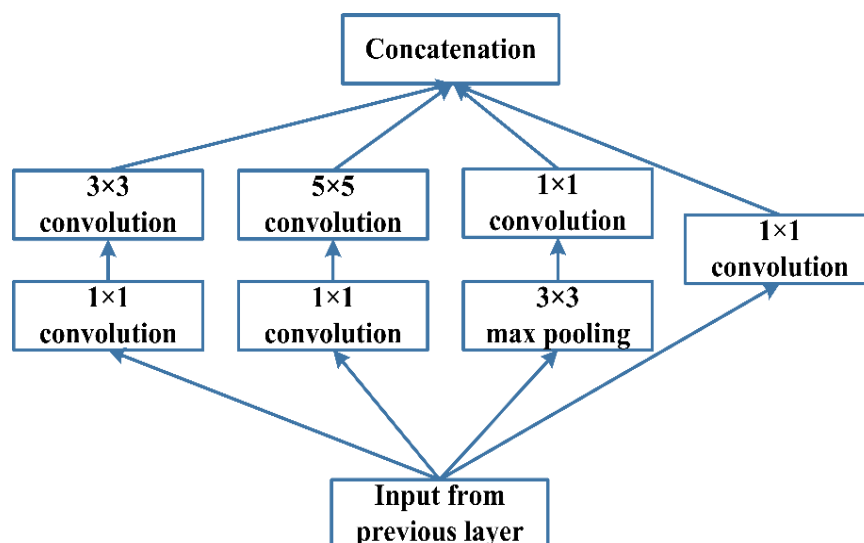


Fig. 9 Basic architecture of the inception block showing the split, transform, and merge concept.

3.4 ResNet

ResNet was proposed by He et al., which is considered as a continuation of deep networks (He et al. 2015a). ResNet revolutionized the CNN architectural race by introducing the concept of residual learning in CNNs and devised an efficient methodology for the training of deep networks. Similar to Highway Networks, it is also placed under the Multi-Path based CNNs; thus, its learning methodology is discussed in Section 4.3.2. ResNet proposed 152-layers deep CNN, which won the 2015-ILSVRC competition. The architecture of the residual block of ResNet is shown in Fig. 7. ResNet, which was 20 and 8 times deeper than AlexNet and VGG, respectively, showed less computational complexity than previously proposed networks (Krizhevsky et al. 2012; Simonyan and Zisserman 2015). He et al. empirically showed that ResNet with 50/101/152 layers has less error on image classification task than 34 layers plain Net. Moreover, ResNet gained a 28% improvement on the famous image recognition benchmark dataset named COCO (Lin et al. 2014). Good performance of ResNet on image recognition and localization tasks showed that representational depth is of central importance for many visual recognition tasks.

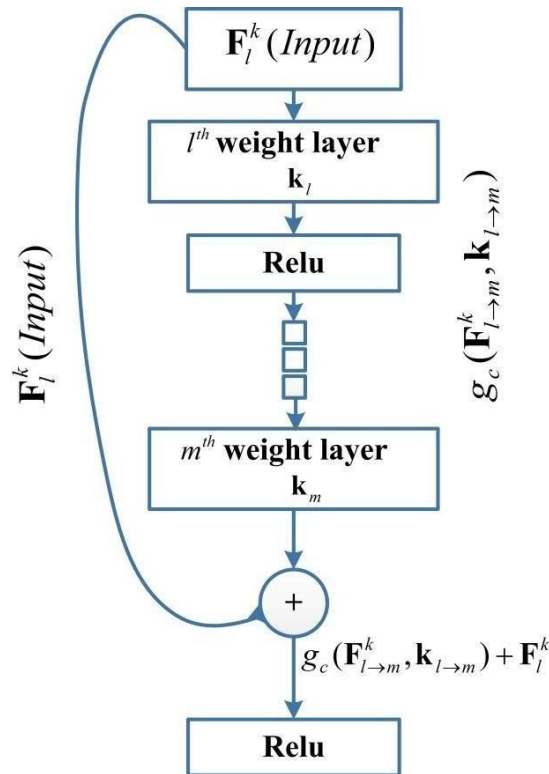


Fig. 10 Residual block as a basic structural unit of ResNet

3.5 Squeeze Network

Squeeze Network was reported by Hu et al. (Hu et al. 2018a). They proposed a new block for the selection of feature-maps (commonly known as channels) relevant to object discrimination. This new block was named as SE-block (shown in Fig. 10), which suppresses the less important feature-maps, but gives high weightage to the class specifying feature-maps. SE-Network reported a record decrease in error on the ImageNet dataset. SE-block is a processing unit that is designed generically and therefore, can be added in any CNN architecture before the convolution layer. The working of this block consists of two operations; squeeze and excitation. Convolution kernel captures information locally, but it ignores the contextual relation of features (correlation) that are outside of this receptive field (LeCun et al. 2015). Squeeze operation is performed to get a global view of feature-maps. The squeeze block generates feature-map wise statistics (also known as feature-map motifs or descriptors) by suppressing spatial information of the convolved input. As global average pooling has the potential to learn the extent of target object effectively (Lin et al. 2013; Zhou et al. 2016),

limit the model complexity and aid the generalization (LeCun 2007; Xu et al. 2015a). The output of the first hidden transformation in NN is preceded by the ReLU activation function, which inculcates non-linearity in motif responses. The gating mechanism is exploited in SE-block using the sigmoid activation function, which models the non-linear responses of the feature-maps and assigns a weight based on feature-map relevance (Zheng et al. 2017). SE-block adaptively recalibrates the feature-maps of each layer by multiplying convolved input with the motif responses.

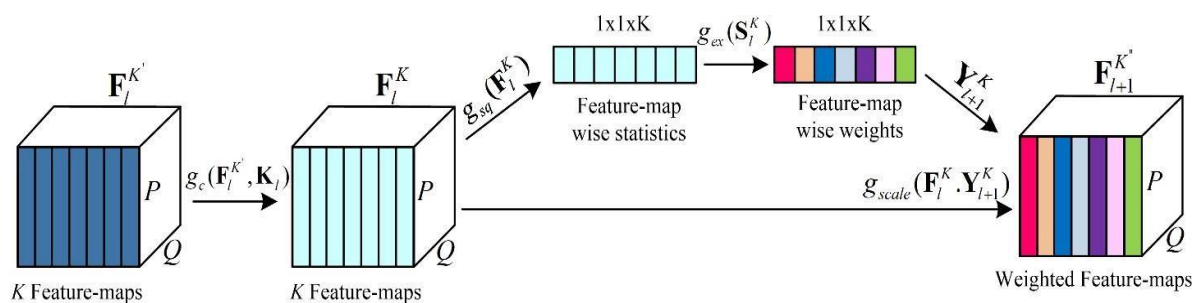


Fig. 11 Squeeze and Excitation block showing the computation of masks for the recalibration of feature-maps that are commonly known as channels in literature.

3.6 VGG NET

The successful use of CNNs in image recognition tasks has accelerated the research in architectural design. In this regard, Simonyan et al. proposed a simple and effective design principle for CNN architectures. Their architecture, named as VGG, was modular in layers pattern (Simonyan and Zisserman 2015). VGG was made 19 layers deep compared to AlexNet and ZfNet to simulate the relation of depth with the representational capacity of the network (Krizhevsky et al. 2012; Zeiler and Fergus 2013). ZfNet, which was a frontline network of 2013-ILSVRC competition, suggested that small size filters can improve the performance of the CNNs. Based on these findings, VGG replaced the 11x11 and 5x5 filters with a stack of 3x3 filters layer and experimentally demonstrated that concurrent placement of small size (3x3) filters could induce the effect of the large size filter (5x5 and 7x7). The use of the small size filters provides an additional benefit of low computational complexity by reducing the number of parameters. These findings set a new trend in research to work with smaller size filters in CNN. VGG regulates the complexity of a network by placing 1x1 convolutions in between the convolutional layers, which, besides, learn a linear combination of the resultant feature-maps.

For the tuning of the network, max-pooling is placed after the convolutional layer, while padding was performed to maintain the spatial resolution (Ranzato et al. 2007). VGG showed good results both for image classification and localization problems. VGG was at 2nd place in the 2014-ILSVRC competition but, got fame due to its simplicity, homogenous topology, and increased depth. The main limitation associated with VGG was the use of 138 million parameters, which make it computationally expensive and difficult to deploy it on low resource systems.

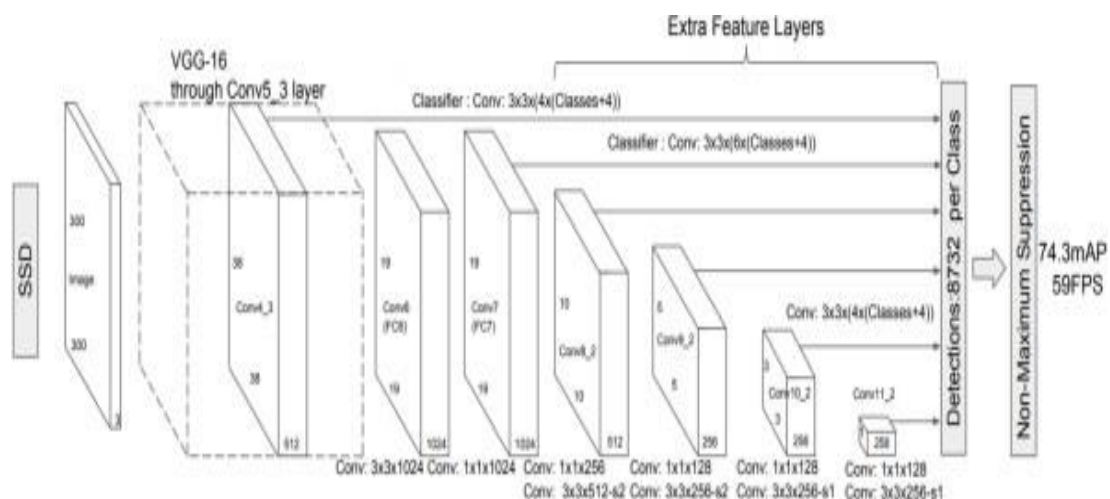


Figure 12 VGG16 architecture

CHAPTER 4

MATERIAL AND METHODS USED

4.1 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 tool boxes for signal processing, symbolic computation, control theory, simulation, optimization, and several other fields of applied science and engineering.

4.1.1 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
- Various other special functions

4.1.2 FEATURES OF MATLAB

Following are the basic features of MATLAB

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

4.1.3 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

4.1.4 UNDERSTANDING THE MATLAB ENVIRONMENT

MATLAB development IDE can be launched from the icon created on the desktop. The main working window in MATLAB is called the desktop. When MATLAB is started, the desktop appears in its default layout.

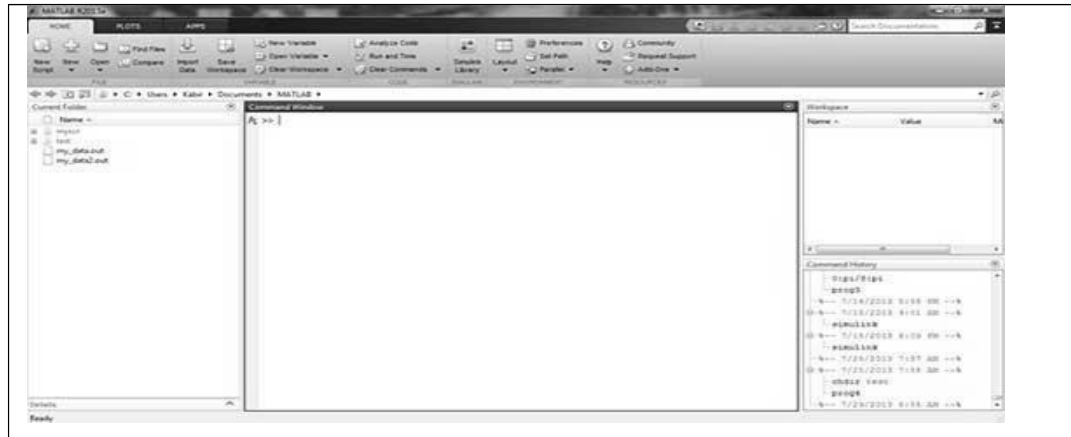


Fig.13. MATLAB desktop environment

The desktop has the following panels:

Current Folder - This panel allows you to access the project folders and files.

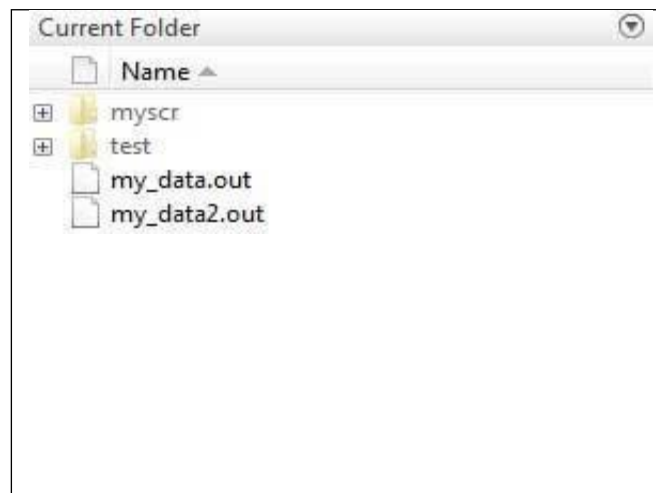


Fig.14. current folder

Command Window - This is the main area where commands can be entered at the command line. It is indicated by the command prompt (>>).

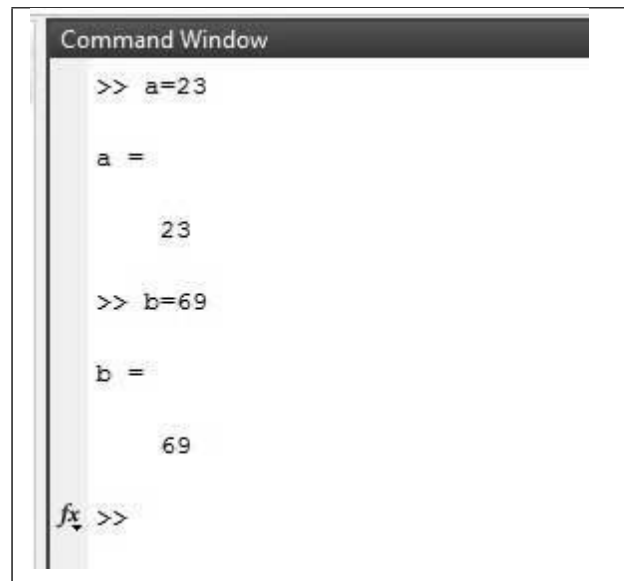


Fig 15 command window

Workspace - The workspace shows all the variables created and/or imported from files.

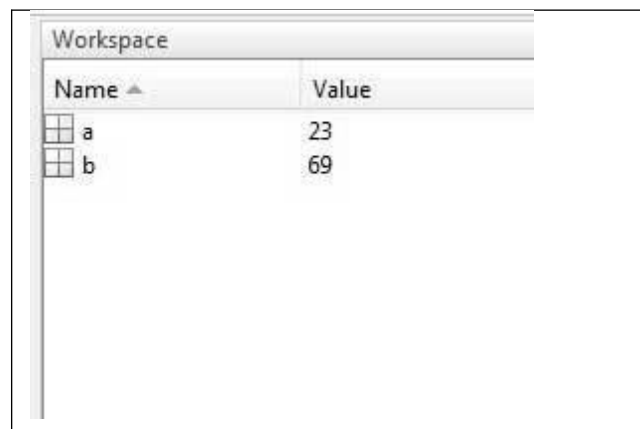


Fig.16 workspace

Command History - This panel shows or rerun commands that are entered at the command line.

```

-- 7/14/2013 5:58 PM --
-- 7/15/2013 9:01 AM --
simulink
-- 7/15/2013 6:09 PM --
simulink
-- 7/25/2013 7:57 AM --
-- 7/25/2013 7:58 AM --
chdir test
prog4
-- 7/29/2013 8:55 AM --
a=23
b=69

```

Fig.17 command history

4.1.5 COMMONLY USED OPERATORS AND SPATIAL CHARATERS

MATLAB supports the following commonly used operators and special characters:

Operator	Purpose
+	Plus; addition operator.
-	Minus, subtraction operator.
*	Scalar and matrix multiplication operator.
.*	Array and multiplication operator.
^	Scalar and matrix exponentiation operator.
.^	Array exponentiation operator.

\	Left-division operator.
/	Right-division operator.
.\	Array left-division operator.
./	Array right-division operator.

Table.4.5.6 MATLAB used operators and special characters.

4.1.6 COMMANDS

MATLAB is an interactive program for numerical computation and data visualization. You can enter a command by typing it at the MATLAB prompt '>>' on the Command Window.

Command for managing a session

MATLAB provides various commands for managing a session. The following table provides all

Commands	Purpose
Clc	Clear command window
Clear	Removes variables from memory
Exist	Checks for existence of file or variable.
Global	Declare variables to be global.
Help	Searches for help topics.
Look for	Searches help entries for a keyword.
Quit	Stops MATLAB.
Who	Lists current variable.
Whos	Lists current variables (Long Display).

Table.4.7.1 commands for managing a session

INPUT AND OUTPUT COMMAND

MATLAB provides the following input and output related commands:

Command	Purpose
Disp	Displays content for an array or string.
Fscanf	Read formatted data from a file.
Format	Control screen-display format.
Fprintf	Performs formatted write to screen or a file.
Input	Displays prompts and waits for input.
;	Suppresses screen printing.

Table.4.8 input and output commands

4.1.7 M FILES

MATLAB allows writing two kinds of program files:

Scripts:

script files are program files with .m extension. In these files, you write series of commands, which you want to execute together. Scripts do not accept inputs and do not return any outputs. They operate on data in the workspace.

Functions:

functions files are also program files with .m extension. Functions can accept inputs and return outputs. Internal variables are local to the function.

Creating and Running Script File:

To create scripts files, you need to use a text editor. You can open the MATLAB editor in two ways:

- Using the command prompt
- Using the IDE

You can directly type edit and then the filename (with .m extension).

Edit

or

edit<file name>

4.1.8 DATA TYPES AVAILABLE IN MATLAB

MATLAB provides 15 fundamental data types. Every data type stores data that is in the form of a matrix or array. The size of this matrix or array is a minimum of 0-by-0 and this can grow up to a matrix or array of any size.

The following table shows the most commonly used data types in MATLAB:

Datatype	Description
Int8	8-bit signed integer
Unit8	8-bit unsigned integer
Int16	16-bit signed integer
Unit16	16-bit unsigned integer
Int32	32-bit signed integer
unit32	32-bit unsigned integer
Int64	64-bit signed integer
Unit64	64-bit unsigned integer
Single	Single precision numerical data
Double	Double precision numerical data
Logical	Logical variables are 1or0,represent true &falserespectively
Char	Character data(strings are stored

	as vector of characters)
Call array	Array of indexed calls, each capable of storing array of a different dimension and datatype
Structure	C-like structure each structure having named fields capable of storing an array of a different dimension and datatype
Function handle	Pointer to a function
User classes	Object constructed from a user-defined class
Java classes	Object constructed from a java class

Table.4.10. data types in MATLAB.

4.2 Microsoft Windows

Microsoft Windows, commonly referred to as **Windows**, is a group of several proprietary graphical operating system families, all of which are developed and marketed by Microsoft. Each family caters to a certain sector of the computing industry. Active Microsoft Windows families include Windows NT and Windows IoT; these may encompass subfamilies, (e.g. Windows Server or Windows Embedded Compact) (Windows CE). Defunct Microsoft Windows families include Windows 9x, Windows Mobile, and Windows Phone.

Microsoft introduced an operating environment named *Windows* on November 20, 1985, as a graphical operating system shell for MS-DOS in response to the growing interest in graphical user interfaces (GUIs).^[5] Microsoft Windows came to dominate the world's personal computer (PC) market with over 90% market share, overtaking Mac OS, which had been introduced in 1984, while Microsoft has in 2020 lost its dominance of the consumer operating system market, with Windows down to 30%, lower than Apple's 31% mobile-only share (65% for desktop operating systems only, i.e. "PCs" vs. Apple's 28% desktop share) in its home market, the US, and 32% globally (77% for desktops), where Google's Android leads.

Apple came to see Windows as an unfair encroachment on their innovation in GUI development as implemented on products such as the Lisa and Macintosh (eventually settled in court in Microsoft's favor in 1993). On PCs, Windows is still the most popular operating system in all

countries.^{[6][7]} However, in 2014, Microsoft admitted losing the majority of the overall operating system market to Android,^[8] because of the massive growth in sales of Android smartphones. In 2014, the number of Windows devices sold was less than 25% that of Android devices sold. This comparison, however, may not be fully relevant, as the two operating systems traditionally target different platforms. Still, numbers for server use of Windows (that are comparable to competitors) show one third market share, similar to that for end user use.

As of October 2020, the most recent version of Windows for PCs, tablets and embedded devices is Windows 10, version 20H2. The most recent version for server computers is Windows Server, version 20H2.^[9] A specialized version of Windows also runs on the Xbox One video game console.^[10]

4.2.1 MULTIPLE DESKTOP AND TASK VIEW

Microsoft Edge

This new browser is designed to give Windows users a better experience on the Web. It's faster, more secure, and includes a lot of **new features**. Microsoft Edge is meant to replace Internet Explorer as your default web browser, but you'll still be able to use IE (or another browser) if you prefer.

Cortana

Similar to Siri and Google Now, you can talk to this **virtual assistant** with your computer's microphone. Cortana can answer questions like **What's the weather like today?**, perform simple tasks like **remind you to take out the trash**, and much more.

Multiple desktops and Task view

Instead of keeping everything open on the same desktop, you can move some of your windows to a **virtual** desktop to get them out of the way. And the new **Task view** feature makes it easy to manage all of your open windows.

Action Center

The new Action Center is pretty different from previous versions of Windows. For example, it's been expanded to let you access frequently used settings, such as Wi-Fi connectivity and tablet mode. It's also where you'll see important **notifications**, so if your computer receives an update you'll get a notification about it here.

4.2.1.2 Tablet mode

Unlike Windows 8, Windows 10 makes a clear distinction between desktops and tablets. If you're using a keyboard and mouse with Windows 10, you'll be in desktop mode by default. If your computer also has a touchscreen, you can go into tablet mode at any time. Tablet users can also switch back to desktop mode if they prefer.

4.2.1.3 Security:

Consumer versions of Windows were originally designed for ease-of-use on a single-user PC without a network connection, and did not have security features built in from the outset.^[69] However, Windows NT and its successors are designed for security (including on a network) and multi-user PCs, but were not initially designed with Internet security in mind as much, since, when it was first developed in the early 1990s, Internet use was less prevalent.

These design issues combined with programming errors (e.g. buffer overflows) and the popularity of Windows means that it is a frequent target of computer worm and virus writers. In June 2005, Bruce Schneier's *Counterpane Internet Security* reported that it had seen over 1,000 new viruses and worms in the previous six months. In 2005, Kaspersky Lab found around 11,000 malicious programs – viruses, Trojans, back-doors, and exploits written for Windows.

Microsoft releases security patches through its Windows Update service approximately once a month (usually the second Tuesday of the month), although critical updates are made available at shorter intervals when necessary. In versions of Windows after and including Windows 2000 SP3 and Windows XP, updates can be automatically downloaded and installed if the user selects to do so. As a result, Service Pack 2 for Windows XP, as well as Service Pack 1 for Windows Server 2003, were installed by users more quickly than it otherwise might have been.^[74]

While the Windows 9x series offered the option of having profiles for multiple users, they had no concept of access privileges, and did not allow concurrent access; and so were not true multi-user operating systems. In addition, they implemented only partial memory protection. They were accordingly widely criticised for lack of security.

The Windows NT series of operating systems, by contrast, are true multi-user, and implement absolute memory protection. However, a lot of the advantages of being a true multi-user operating system were nullified by the fact that, prior to Windows Vista, the first user account created during the setup process was an administrator account, which was also the default for new accounts.

Though Windows XP did have limited accounts, the majority of home users did not change to an account type with fewer rights – partially due to the number of programs which unnecessarily required administrator rights – and so most home users ran as administrator all the time.

Windows Vista changes this by introducing a privilege elevation system called User Account Control. When logging in as a standard user, a logon session is created and a token containing only the most basic privileges is assigned. In this way, the new logon session is incapable of making changes that would affect the entire system. When logging in as a user in the Administrators group, two separate tokens are assigned. The first token contains all privileges typically awarded to an administrator, and the second is a restricted token similar to what a standard user would receive. User applications, including the Windows shell, are then started with the restricted token, resulting in a reduced privilege environment even under an Administrator account. When an application requests higher privileges or "Run as administrator" is clicked, UAC will prompt for confirmation and, if consent is given (including administrator credentials if the account requesting the elevation is not a member of the administrators group), start the process using the unrestricted token.^[76]

Leaked documents published by WikiLeaks, codenamed Vault 7 and dated from 2013 to 2016, detail the capabilities of the CIA to perform electronic surveillance and cyber warfare, such as the ability to compromise operating systems such as Microsoft Windows.^[77]

In August 2019, computer experts reported that the Blue Keep security vulnerability, CVE-2019-0708, that potentially affects older unpatched Microsoft Windows versions via the program's Remote Desktop Protocol, allowing for the possibility of remote code execution, may now include related flaws, collectively named *Deja Blue*, affecting newer Windows versions (i.e., Windows 7 and all recent versions) as well.^[79] In addition, experts reported a Microsoft security vulnerability, CVE-2019-1162, based on legacy code involving Microsoft CTF and ctfmon (ctfmon.exe), that affects all Windows versions from the older Windows XP version to the most recent Windows 10 versions; a patch to correct the flaw is currently available.

CHAPTER 5

RESULT AND DISCUSSION

5.1 ALEX NET



True Class					
	Raghu	anwar	manoj	naveen	
	Raghu				
	anwar				
	manoj				
naveen					
	Predicted Class	Raghu	anwar	manoj	naveen

Confusion Matrix Data:

- Raghu (True Class) vs Raghu (Predicted Class): 2
- anwar (True Class) vs anwar (Predicted Class): 2
- manoj (True Class) vs manoj (Predicted Class): 1
- naveen (True Class) vs manoj (Predicted Class): 2
- manoj (True Class) vs naveen (Predicted Class): 1

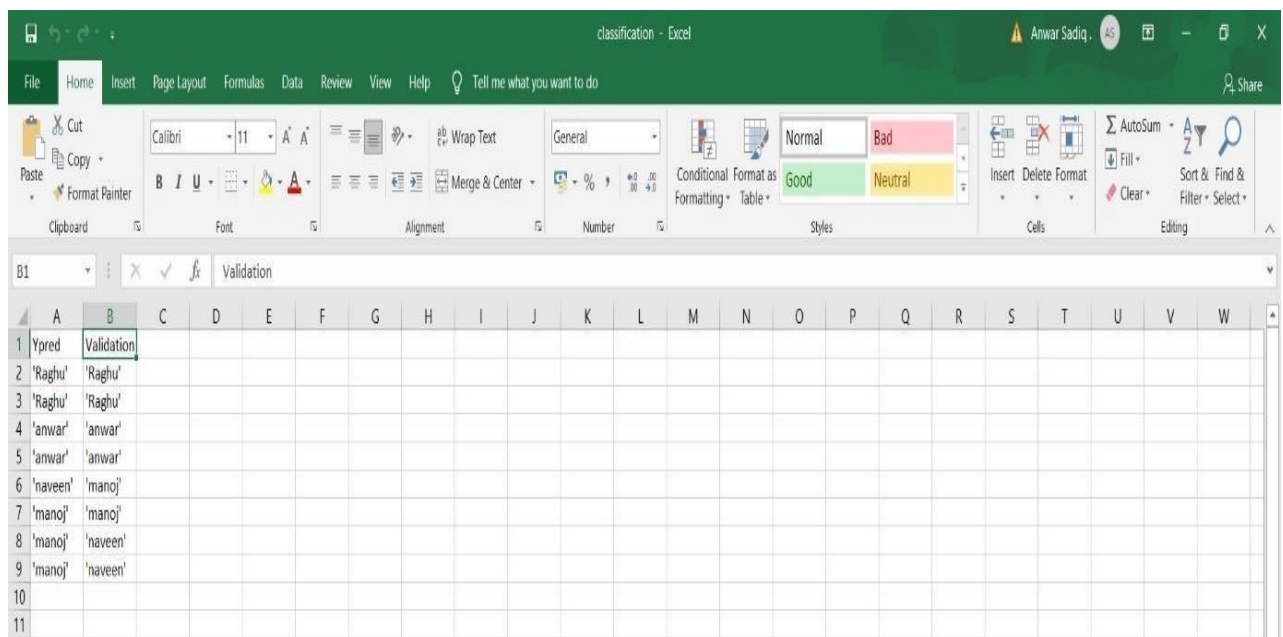


Figure18 alex net result

Training Progress (11-Mar-2021 11:24:48)

Training Progress (11-Mar-2021 11:24:48)

Results

- Validation accuracy: 75.00%
- Training finished: Reached final iteration

Training Time

- Start time: 11-Mar-2021 11:24:48
- Elapsed time: 1 min 19 sec

Training Cycle

- Epoch: 6 of 6
- Iteration: 6 of 6
- Iterations per epoch: 1
- Maximum iterations: 6

Validation

- Frequency: 1 iterations

Other Information

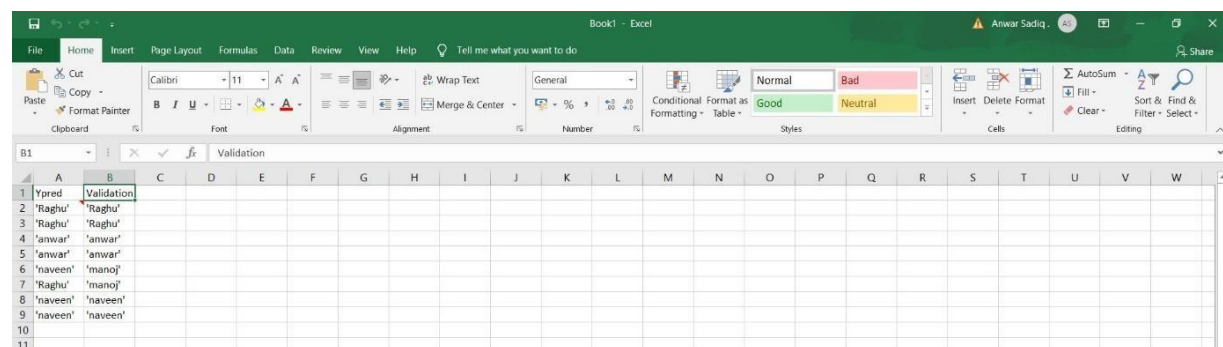
- Hardware resource: Single CPU
- Learning rate schedule: Constant
- Learning rate: 0.0003

[Learn more](#)

Accuracy

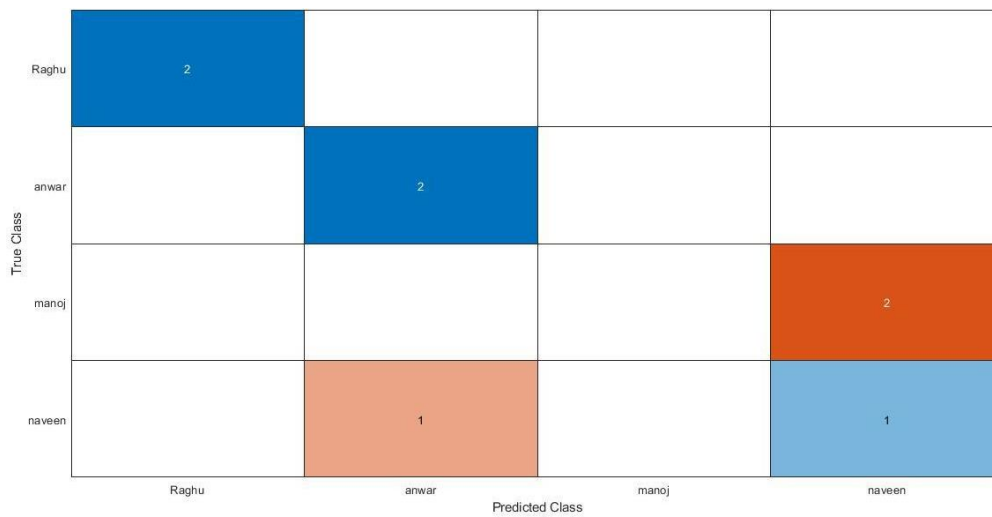
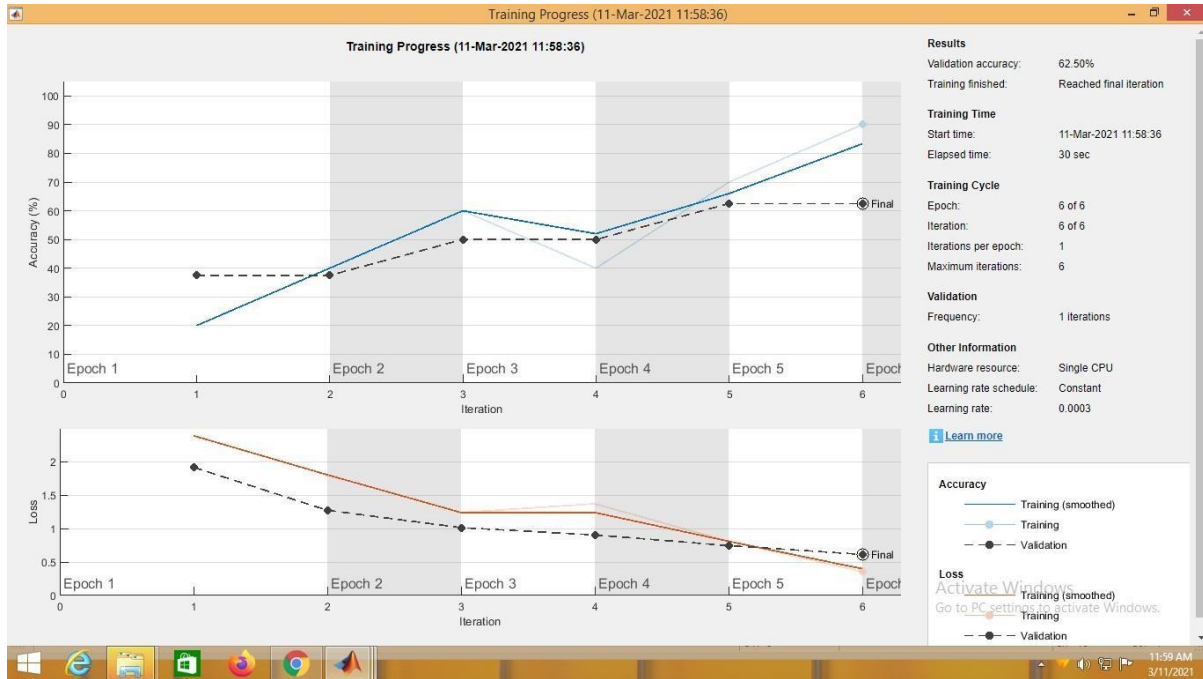
- Training (smoothed)
- Validation

Iteration	Epoch	Training Accuracy (%)	Validation Accuracy (%)	Training Loss	Validation Loss
1	1	40	40	1.5	1.5
2	2	30	50	1.3	1.3
3	3	60	75	1.0	1.0
4	4	90	75	0.8	0.8
5	5	100	65	0.6	0.6
6	6	100	65	0.2	0.2



Page 32

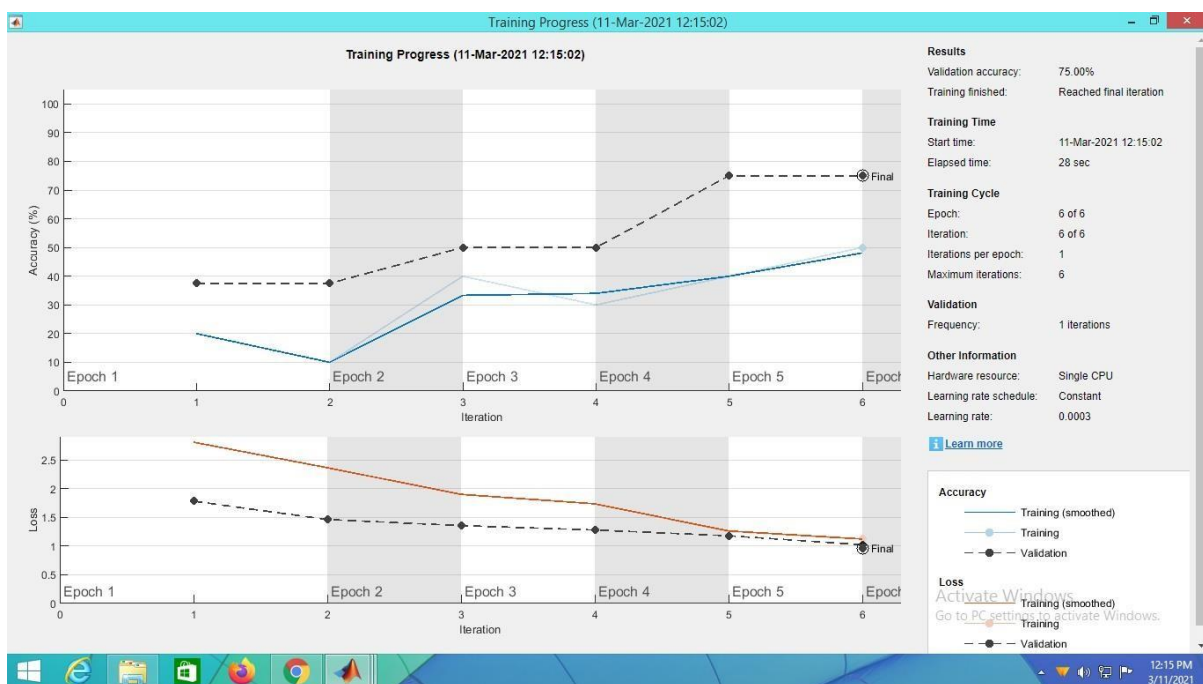
5.4 Google net

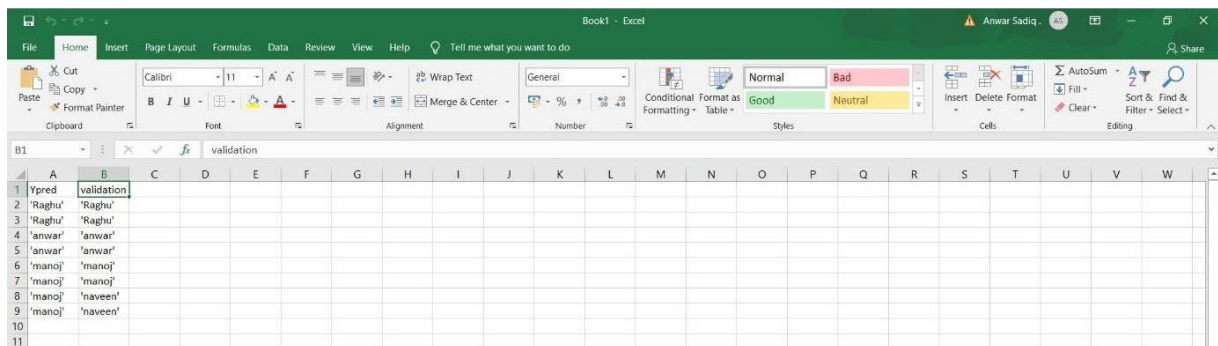


	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
1	ypred	validation																					
2	'Raghu'	'Raghu'																					
3	'Raghu'	'Raghu'																					
4	'anwar'	'anwar'																					
5	'anwar'	'anwar'																					
6	'naveen'	'manoj'																					
7	'naveen'	'manoj'																					
8	'anwar'	'naveen'																					
9	'naveen'	'naveen'																					
10																							
11																							

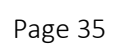
Figure20 Google net result

5.4 RESS NET





5.5 Squeez



True Class	Raghu	2			
	anwar		2		
	manoj			1	1
	naveen			1	1
		Raghu	anwar	manoj	naveen
		Predicted Class			

validation																						
1	Ypred																					
2	'Raghu'	'Raghu'																				
3	'Raghu'	'Raghu'																				
4	'anwar'	'anwar'																				
5	'anwar'	'anwar'																				
6	'naveen'	'manoj'																				
7	'manoj'	'manoj'																				
8	'naveen'	'naveen'																				
9	'manoj'	'naveen'																				
10																						
11																						

Figure22 Squeeze net result

Training Progress (11-Mar-2021 11:37:52)

Results

- Validation accuracy: 75.00%
- Training finished: Reached final iteration

Training Time

- Start time: 11-Mar-2021 11:37:52
- Elapsed time: 1 min 11 sec

Training Cycle

- Epoch: 6 of 6
- Iteration: 6 of 6
- Iterations per epoch: 1
- Maximum iterations: 6

Validation

- Frequency: 1 iterations

Other Information

- Hardware resource: Single CPU
- Learning rate schedule: Constant
- Learning rate: 0.0003

[Learn more](#)

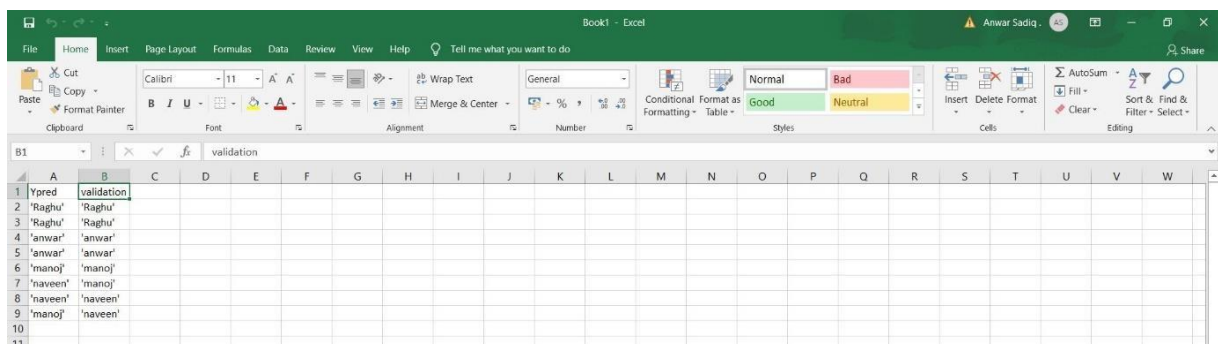
Accuracy

- Training (smoothed)

Loss

- Training (smoothed)

Go to PC settings to activate Windows.



Page 37

CHAPTER 6

CONCLUSION

In our project we have shown the overview of face recognition using convolution neural network algorithm Convolution Neural network is an algorithm widely used in Face Recognition in that architecture like alex net , dense net , google net , resnet , squeeze net, vgg 16 and so many more are been used. This paper proposes a overview of face recognition of Convolution Neural Network (CNN) with some of famous architecture like alex net, dense net , google net , resnet ,squeeze net ,vgg 16 with same parameters and datasets are been tested with all the architecture . The number of iterations and the efficiency of the architecture show in the result in the proper manner , Datasets and parameters same are used for all architecture . The proposed system has designed by MATLAB Software of the version MATLAB 8.3 R2014a and code is as follows

Alex net code

```
clear all;
close all;

imdsTrain = imageDatastore('C:\Users\admin\Documents\MATLAB\face\train', ...
    'IncludeSubfolders',true,'LabelSource','foldernames');

imdsValidation=imageDatastore('C:\Users\admin\Documents\MATLAB\face\test', ...
    'IncludeSubfolders',true,'LabelSource','foldernames');

net = alexnet;
analyzeNetwork(net)
net.Layers(1)
inputSize = net.Layers(1).InputSize;
if isa(net,'SeriesNetwork')
    lgraph = layerGraph(net.Layers);
else
    lgraph = layerGraph(net);
end
[learnableLayer,classLayer] = findLayersToReplace(lgraph);
[learnableLayer,classLayer]
numClasses = numel(categories(imdsTrain.Labels));
```

```

if isa(learnableLayer,'nnet.cnn.layer.FullyConnectedLayer')
    newLearnableLayer = fullyConnectedLayer(numClasses, ...
        'Name','new_fc', ...
        'WeightLearnRateFactor',10, ...
        'BiasLearnRateFactor',10);

elseif isa(learnableLayer,'nnet.cnn.layer.Convolution2DLayer')
    newLearnableLayer = convolution2dLayer(1,numClasses, ...
        'Name','new_conv', ...
        'WeightLearnRateFactor',10, ...
        'BiasLearnRateFactor',10);
end

lgraph = replaceLayer(lgraph,learnableLayer.Name,newLearnableLayer);
newClassLayer = classificationLayer('Name','new_classoutput');
lgraph = replaceLayer(lgraph,classLayer.Name,newClassLayer);
figure('Units','normalized','Position',[0.3 0.3 0.4 0.4]);
plot(lgraph)
ylim([0,10])
layers = lgraph.Layers;
connections = lgraph.Connections;

layers(1:10) = freezeWeights(layers(1:10));
lgraph = createLgraphUsingConnections(layers,connections);
pixelRange = [-30 30];
scaleRange = [0.9 1.1];
imageAugmenter = imageDataAugmenter( ...
    'RandXReflection',true, ...
    'RandXTranslation',pixelRange, ...

```

```

'RandYTranslation',pixelRange, ...
'RandXScale',scaleRange, ...
'RandYScale',scaleRange);
augimdsTrain = augmentedImageDatastore(inputSize(1:2),imdsTrain, ...
    'DataAugmentation',imageAugmenter);
augimdsValidation = augmentedImageDatastore(inputSize(1:2),imdsValidation);
miniBatchSize = 10;
valFrequency = floor(numel(augimdsTrain.Files)/miniBatchSize);
options = trainingOptions('sgdm', ...
    'MiniBatchSize',miniBatchSize, ...
    'MaxEpochs',6, ...
    'InitialLearnRate',3e-4, ...
    'Shuffle','every-epoch', ...
    'ValidationData',augimdsValidation, ...
    'ValidationFrequency',valFrequency, ...
    'Verbose',false, ...
    'Plots','training-progress');
net = trainNetwork(augimdsTrain,lgraph,options);
[YPred,probs] = classify(net,augimdsValidation);
accuracy = mean(YPred == imdsValidation.Labels)
t=table(YPred,imdsValidation.Labels)
cm=confusionmat(imdsValidation.Labels,YPred);
cm1=confusionchart(imdsValidation.Labels,YPred);

```

Dense net code

```

clear all;
close all;
imdsTrain = imageDatastore('C:\Users\admin\Documents\MATLAB\face\train', ...
    'IncludeSubfolders',true,'LabelSource','foldernames');
imdsValidation=imageDatastore('C:\Users\admin\Documents\MATLAB\face\test', ...

```

```

    'IncludeSubfolders',true,'LabelSource','foldernames');
net = densenet201;
analyzeNetwork(net)
net.Layers(1)
inputSize = net.Layers(1).InputSize;
if isa(net,'SeriesNetwork')
    lgraph = layerGraph(net.Layers);
else
    lgraph = layerGraph(net);
end
[learnableLayer,classLayer] = findLayersToReplace(lgraph);
[learnableLayer,classLayer]
numClasses = numel(categories(imdsTrain.Labels));

if isa(learnableLayer,'nnet.cnn.layer.FullyConnectedLayer')
    newLearnableLayer = fullyConnectedLayer(numClasses, ...
        'Name','new_fc', ...
        'WeightLearnRateFactor',10, ...
        'BiasLearnRateFactor',10);

elseif isa(learnableLayer,'nnet.cnn.layer.Convolution2DLayer')
    newLearnableLayer = convolution2dLayer(1,numClasses, ...
        'Name','new_conv', ...
        'WeightLearnRateFactor',10, ...
        'BiasLearnRateFactor',10);
end

lgraph = replaceLayer(lgraph,learnableLayer.Name,newLearnableLayer);
newClassLayer = classificationLayer('Name','new_classoutput');
lgraph = replaceLayer(lgraph,classLayer.Name,newClassLayer);

```

```

figure('Units','normalized','Position',[0.3 0.3 0.4 0.4]);
plot(lgraph)
ylim([0,10])
layers = lgraph.Layers;
connections = lgraph.Connections;

layers(1:10) = freezeWeights(layers(1:10));
lgraph = createLgraphUsingConnections(layers,connections);
pixelRange = [-30 30];
scaleRange = [0.9 1.1];
imageAugmenter = imageDataAugmenter( ...
    'RandXReflection',true, ...
    'RandXTranslation',pixelRange, ...
    'RandYTranslation',pixelRange, ...
    'RandXScale',scaleRange, ...
    'RandYScale',scaleRange);
augimdsTrain = augmentedImageDatastore(inputSize(1:2),imdsTrain, ...
    'DataAugmentation',imageAugmenter);
augimdsValidation = augmentedImageDatastore(inputSize(1:2),imdsValidation);
miniBatchSize = 10;
valFrequency = floor(numel(augimdsTrain.Files)/miniBatchSize);
options = trainingOptions('sgdm', ...
    'MiniBatchSize',miniBatchSize, ...
    'MaxEpochs',6, ...
    'InitialLearnRate',3e-4, ...
    'Shuffle','every-epoch', ...
    'ValidationData',augimdsValidation, ...
    'ValidationFrequency',valFrequency, ...
    'Verbose',false, ...
    'Plots','training-progress');

```

```
net = trainNetwork(augimdsTrain,lgraph,options);  
[YPred,probs] = classify(net,augimdsValidation);  
accuracy = mean(YPred == imdsValidation.Labels)  
t=table(YPred,imdsValidation.Labels)  
cm=confusionmat(imdsValidation.Labels,YPred);  
cm1=confusionchart(imdsValidation.Labels,YPred);
```

CHAPTER 7

REFERENCES

- [1] Mohammed Alghaili, Hamdi A. R. Ali, “Face Identification with Deep Learning and Filter Algorithm” received 21 November 2019; Accepted 2 April 2020; Published 1 August 2020.
- [2] Z. Zhu, P. Luo, X. Wang, and X. Tang, “Hybrid deep learning for face verification,” in Proceedings of the International Conference on Computer Vision, pp. 113–120, IEEE, Sydney, NSW, Australia, December 2013.
- [3] W. Liu, Y. Wen, Z. Yu et al., “Sphereface: deep hypersphere embedding for face recognition,” in Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR), July 2017.
- [4] Z. Zhu, P. Luo, X. Wang, and X. Tang, “Recover canonical view faces in the wild with deep neural networks,” pp. 1404–3543, 2014, <https://arxiv.org/abs/1404.3543>.
- [5] Y. Sun, X. Wang, and X. Tang, “Deep learning face representation from predicting 10,000 classes,” in Proceedings of the Conference on Computer Vision and Pattern Recognition, pp. 1891–1898, IEEE, Columbus, OH, USA, June 2014.
- [6] Y. Sun, Y. Chen, X. Wang, and X. Tang, “Deep learning face representation by joint identification-verification,” Advances in Neural Information Processing Systems, pp. 1988–1996, 2014, <https://arxiv.org/abs/1406.4773>.
- [7] Y. Taigman, M. Yang, M. A. Ranzato, and L. Wolf, “Web-scale training for face identification,” in Proceedings of the Conference on Computer Vision and Pattern Recognition, pp. 2746–2754, IEEE, Boston, MA, USA, June 2015.
- [8] Z. Zhu, P. Luo, X. Wang, and X. Tang, “Deep learning and disentangling face representation by multi-view perceptron,” in Proceedings of the NIPS, Quebec, Canada, December 2015.
- [9] K. Q. Weinberger and L. K. Saul, “Distance metric learning for large margin nearest neighbour classification,” Journal of Machine Learning Research, vol. 10, pp. 207–244, 2009.
- [10] Y. Sun, X. Wang, and X. Tang, “Deeply learned face representations are sparse,

selective, and robust,” in Proceedings of the Conference on Computer Vision and Pattern Recognition, pp. 2892–2900, IEEE, Boston, MA, USA, June 2015.

[11] D. Yi, Z. Lei, S. Liao, and S. Z. Li, “Learning face representation from scratch,” pp. 1411–7923, 2014, <https://arxiv.org/abs/1411.7923>.

[12] Y. Wang, J. Shen, S. Petridis, and M. Pantic, “A real-time and unsupervised face re-identification system for human-robot interaction,” Pattern Recognition Letters, vol. 128, 2018.

[13] Z. Liu, P. Luo, X. Wang, and X. Tang, “Deep learning face attributes in the wild,” in Proceedings of the International Conference on Computer Vision, pp. 3730–3738, IEEE, Santiago, Chile, December 2015.

[14] Z. Zhu, P. Luo, X. Wang, and X. Tang, “Deep learning identity-preserving face space,” in Proceedings of the IEEE International Conference on Computer Vision, pp. 1489–1496, IEEE, Sydney, NSW, Australia, December 2013.

ORIGINALITY REPORT

6%

SIMILARITY INDEX

3%

INTERNET SOURCES

4%

PUBLICATIONS

4%

STUDENT PAPERS

PRIMARY SOURCES

1	Yi Sun, Xiaogang Wang, Xiaoou Tang. "Deeply learned face representations are sparse, selective, and robust", 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015 Publication	1 %
2	arxiv.org Internet Source	1 %
3	Submitted to Mehran University of Eng. & Technology Student Paper	1 %
4	Asifullah Khan, Anabia Sohail, Umme Zahoora, Aqsa Saeed Qureshi. "A survey of the recent architectures of deep convolutional neural networks", Artificial Intelligence Review, 2020 Publication	1 %
5	Submitted to Coventry University Student Paper	1 %
6	Submitted to University of Wales Institute, Cardiff Student Paper	<1 %

7	Submitted to Shri Shankaracharya Technical Campus Student Paper	<1 %
8	wyliu.com Internet Source	<1 %
9	Sun, Yi, Xiaogang Wang, and Xiaoou Tang. "Hybrid Deep Learning for Face Verification", 2013 IEEE International Conference on Computer Vision, 2013. Publication	<1 %
10	Submitted to TAR University College Student Paper	<1 %
11	Submitted to CONACYT Student Paper	<1 %
12	export.arxiv.org Internet Source	<1 %
13	www.hitl.washington.edu Internet Source	<1 %
14	computer-trading.com Internet Source	<1 %
15	www.arxiv-vanity.com Internet Source	<1 %