Handwritten Bangla Character Recognition to Braille Pattern Conversion using Image Processing and Machine Learning

A Thesis Submitted to
The Department of Electrical and Electronic Engineering
Of BRAC University for partial fulfillment of the requirement for the degree of
Bachelors of Science in Electrical and Electronic Engineering by

Tawhidur Rahman Abir-13221017
Touseef Saleh Bin Ahmed-14221004
Md. Tausif Rahman- 14121031
Sumaiya Jafreen- 14121041

Fall 2018, BRAC University

Supervised by
Dr. Mohammed Belal Hossain Bhuiyan
Associate Professor
Department of Electrical and Electronics Engineering
BRAC University, Dhaka

Co-supervised by
Dr. Md Ashraful Alam
Assistant Professor
Department of Computer Science and Engineering
BRAC University, Dhaka

Handwritten Bangla Character Recognition to Braille Pattern Conversion using Image Processing and Machine Learning

A Thesis Submitted to
The Department of Electrical and Electronic Engineering
Of BRAC University for partial fulfillment of the requirement for the degree of
Bachelors of Science in Electrical and Electronic Engineering by

Tawhidur Rahman Abir-13221017
Touseef Saleh Bin Ahmed-14221004
Md. Tausif Rahman- 14121031
Sumaiya Jafreen- 14121041

Supervised by
Dr. Mohammed Belal Hossain Bhuiyan
Associate Professor
Department of Electrical and Electronics Engineering
BRAC University, Dhaka

Co-supervised by
Dr. Md. Ashraful Alam
Assistant Professor
Department of Computer Science and Engineering
BRAC University, Dhaka

Fall 2018, BRAC University

Declaration

We hereby declare that this paper titled "Handwritten Bangla Character Recognition to Braille Pattern Conversion using Image Processing and Machine Learning" is our original work and has not been presented elsewhere for assessment or award of any other degree or any other publication. Any additional resources have been properly acknowledged and referred.

Tawhidur Rahman Abir

Touseef Saleh Bin Ahmed

Md. Tausif Rahman

Sumaiya Jafreen

Dr. Mohammed Belal Hossain Bhuiyan

Associate Professor
Department of Electrical and Electronics Engineering,
BRAC University

Dr. Md. Ashraful Alam

Assistant Professor
Department of Computer Science and Engineering,
BRAC University

Acknowledgment

We are extremely grateful to our supervisor, Associate Prof. Dr. Md. Belal Hossain Bhuiyan and Co-supervisor Md. Ashraful Alam for their guidance and discussions. Without their continuous support this thesis would not have been possible. We also have to thank B.M Abir Hossain for helping and supporting us in learning and gathering information on Convolution Neural Network, Deep Learning which helped us program the software portion of the project. We are also grateful for the contributions of all the author in this field which helped us develop the idea and in writing our thesis. We also thank Rahul Kar for developing a well designed library for displaying bangla characters on an LCD screen.

Abstract

There has been significant development in the recent Education System with the rapid development of technology however there are very few facilities that can help those people with disabilities such people without sight. Braille has been one such method however it has not yet been digitized or has not been incorporated in the education system. On the other hand, many papers and works have been conducted on English Character Recognition with satisfactory outcomes with considerably good accuracy however such cannot be said about Bengali Characters. Some works have been done with considerable accuracy however its frequency is still low. This paper introduces the development of a prototype system which takes an image of a handwritten Bangla Character and applies the concepts of Image Processing using ***OpenCV*** and ***Machine Learning*** to capture and process an image then recognize the character and finally with the help of a device present the recognized character in a Braille pattern. Furthermore, this paper will also look into the different machine learning modules and assess the reliability and accuracy. For the machine learning part ***Deep Neural Network*** was applied on the image and then ***VGG-16***, ***Resnet-50*** and ***DenseNet-121*** which are modules of Convoluted Neural Networks where used. Then the outcome is passed into a device which will engrave the corresponding character into its respective Braille Pattern. The device will consist of ***Atmega2560 chipset (Arduino Mega)***, Servo Motor and LCD which will process the data and finally present it in Braille pattern using servo motor and will show the corresponding character on the LCD monitor.

Table of Content

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

Bangla is the fifth most spoken language in the world and the second most popular language in the Indian subcontinent with almost 220 million speaking the language in this region and 300 million all across the world [1]. [2,3] It is a language of great historical significance. UNESCO has recognized the 21st of February as International Mother Language Day to commemorate the sacrifice made by the martyrs who have given their lives for the language during 1952 language movement. Bangla is the mother tongue of the people of Bangladesh, a part of the greater Indo-European Language Family. Hence Bangla language is crucial and plays a huge role academically in these regions. Education is important and is one of basic Human Rights that all people are entitled to have. However, it is very difficult for the visually impaired population of the country to read and write which makes the perception of education and knowledge difficult. In this case listening becomes the only efficient means of obtaining knowledge however it is not as satisfying as it would be when you can actually read the words from book. This problem was faced by many who were visually impaired and finally in 1824, Louis Braille came up with the Braille, named after himself which was system for reading and writing for the visually disadvantaged.

The Braille first discovered[2,10] by Louis Braille in 1824 was the first system of reading and writing for the visually impaired people.[10] It was said to be the first binary form of writing. The visually impaired people use their fingers to feel the pattern or character.[2] The braille system was a six dot pattern with two columns and 3 rows that are arranged in a rectangle. A dot can be raised at one position resulting in sixty four permutations. Left side of the braille is numbered 1-3 and the right hand side is labeled 4-6 universally.

Recently the concept of automatic handwritten character recognition has increasingly become popular both academically and commercially and is being widely researched on. Research into fields such as Deep Learning and Machine Learning have led to the development of many algorithms which currently excel at recognizing handwritten character. However, the major challenge in recognizing characters as pointed out by Zahangir et al [3] is that handwriting tends to greatly vary in shape and size from individual to individual for each language. Furthermore, depending on the language the characters can either be

isolated or cursive in nature increasing the complexity of recognizing the character. In terms of Bangla character. It consists of 50 basic characters with 24 compound character and 10 distinct digits. Moreover, certain characters contain signs/marks on the top or bottom. Some characters are similar to each other and differ by something as small as a dot or a line. As a result, it makes it difficult to a achieve a reliable performance and thereby hindering development of Bangla character recognition.

1.2 Motivation

After much research a lot of work has been done on character recognition and there have been quite a few contributions on the development of the braille since the time of its emergence. However, seeing how Bangla language is spoken both regionally and abroad by many people so it is necessary to develop systems that accurately recognizes Bangla characters. According to our research there have been a contribution but it is still in small numbers compared to amount of contributions made in the recognition of English characters or other language. Despite that, the education system is still unattainable by many handicap individuals mainly the visually impaired population who are unable to read and hence cannot undergo a proper education. Bangla being the mother tongue of Bangladesh it is an absolutely crucial part of education. As we move more towards the future technology is playing even a bigger role in all sectors of the country. In addition, the growth has also resulted in the development of much more powerful computers with stronger processing capabilities. So, making use of present technology, we have come up with a prototype that combines the concepts of character recognition and braille that will bring about change to the education system giving even the blind an opportunity at education.

1.3 Overview

This thesis report is organized as follows with chapter 3 describing the basic theories behind Image processing and how they were implemented in this research. Chapter 4 looks into the dataset used and preprocessing that was needed to be done before using the dataset. Chapter 5 explains the fundamentals of Deep Learning covering basics of neural networks, deep neural networks and finally convolutional neural networks. Chapter 6 explores the existing architectures of Convolutional Neural Network. Chapter 7 describes hardwares and softwares used in the implementation of the system followed by results and outcomes in Chapter 8. Lastly Chapter 9 conclusively concludes our thesis.

Chapter 2

Literature Review

While completing our thesis, we have sought help of many different papers written by many authors that were related to the topic we have been focusing on that have greatly helped us. There have been many exceptional works in the field of character recognition based on Bangla handwriting. Out of all these some exceptional work was that done by the pioneers of Bangla Handwritten Character and Digit Classification U.Pal, A. Roy Bhattacharya. They provided the backbone that lead to several other contributions in this field of research. Their research and outcome have been a huge aid in our thesis.

In the work by Pal et. al [4], they basically used extracted features from the concept of water overflow from reservoir as well topological and statistical features of numerals to design a scheme. Pal then applied the scheme on individuals of different background and obtained recognition accuracy of 91.98%. However, in his work some important factors such as recognition reliability and response time have not been highlighted which are essential for practical applications. The reliability in any research related to character recognition is very crucial as it explains the relationship between the error rate and the recognition rate.

Another research in this same topic has been conducted by the author Bhattacharya. In his paper, [5] he proposed a model which was two stage character recognition scheme. The first stage consisted of 50 classes of basic Bangla Characters in which a rectangular grid consisting of regular spaced horizontal and vertical lines is overlaid on the character bounding box and feature vector for the first classifier is computed, where the response of this first classifier is analyzed to identify its confusion between a pair of similar shaped characters. In the second stage of the scheme involved another rectangular grid that is overlaid over the character bounding box to compute the feature vector, but this time the rectangular grid consists of irregularly spaced with horizontal and vertical lines over the character bounding box. In both the stages, they used Modified Quadratic Discriminant Function (MQDF) classifier and MLP as classifiers respectively.

Similarly, Das et al., [6] also worked on Bangla Character recognition. In this paper, taking into consideration all the complexity involved in the Optical Character Recognition for Bangla Character Recognition they presented an approach that makes an attempt to identify compound character classes from most frequently to less frequently occurred ones. In doing this, they developed a framework for incrementally increasing the number of learned classes of compound characters from more frequently

occurred ones to less frequently occurred ones along with Basic characters. The experimentation showed an average recognition rate of 79.25% after three-fold cross validation of data with future scope of improvement and extension.

Alongside machine learning, Image processing has been a popular research topic for a long time and has scope for many innovations in technological applications. Alongside that OpenCV has made a huge impact on the development of image processing and the its scope. Research on this field has been done by many but the one paper which helped in developing our thesis was that by Guobo Xie and Wen Lu. [7] In this paper, the number of copper core in small wire was found using OpenCV and image processing algorithms. It involved first using a camera to capture an image then they applied OpenCV functions for image processing and lastly, they used morphological opening and closing operations.

Another contribution which helped us greatly was the work done by Palekar et al, [8] in which they presented an implementation of a image to text converter. The paper also mentions the outlining steps needed to convert obtain text from a image file and then create a separate file containing the extracted text. They also considered the shortcomings of various applications and used various image processing techniques and filtrations to overcome the shortcomings.

During our research we have also read many paper and research related to the braille that have supported us in our thesis. One such notable work was done by Hossain et al. [2] In this paper, an experiment was conducted which used structured and state elimination method to validate and generate an expression from DFA (Determination Finite Automaton) design for Bangla to braille translator machine. Another contribution which helped us understand braille was made by Anupam. [9] In his paper, he stated that the 6-dots were slow and tedious and made a braille for English Character where he introduced a comprehensive unified braille Class value that incorporated a 8-dot Class value braille pattern which could represent maximum 256 distinct symbols. According him it was capable of solving problems while writing texts, mathematical and technical writings. Sutariya et al [10] in their paper introduced an economic system which took in a digital file as an input that helped the visually impaired people to learn the Braille Alphabets, numbers and various alphanumeric characters.

Chapter 3

Theory Behind Image Processing

Digital Image Processing is the use of computer algorithms to perform image processing on digital images.[8] Furthermore a much wider range of algorithms can be used on the input data and preventing problems of accumulation of noise, distortion of image or input signal. Recently the development of real time image processing systems that have various applications in the field of automation, embedded systems and robotics.

3.1 Color to Grey

The images used as the source image are colored however the problem with colored images are that they are complicated and harder to process. In order to make things easier it is converted to grayscale for ease of processing [11]. Grayscale images, otherwise known as black and white images are single samples that contains only the intensity information. Grayscale images constitutes of shades of gray ranging from black being the lowest intensity to white being the highest intensity. In addition these intensities are stored as an 8-bit integer thus allowing 256 possible combinations os shades of gray. Figure 1 below shows the input image after it has been converted to grayscale

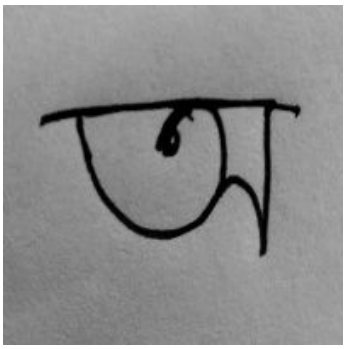


Figure 1 Grayscale Image

3.2 Thresholding

Another important concept that is frequently used in image processing is thresholding.[8] Thresholding or otherwise known as binarization is basically the process of converting a color image to bi-level one.[8,12] This works by setting a predefined threshold value above which the gray level is 0 and below it the level

of gray is 1 where 0 refers to black and 1 refers to white. As stated by G.Xie and W.Lu [7] the selection of a proper threshold value determines the whether the pixel is a background or an object. Moreover the thresholding helps in retaining the structural characteristics of the image. In our paper a correct value was needed in order to separate the the characters in white from the black background we also had to invert the image while thresholding to match our dataset. Furthermore, setting an optimum threshold value will depend upon factors such as light intensity, the different shades and intensities of gray, the method by which the image is captured(ie webcam or cell phone or raspberry pi camera). Therefore considering all these factors we have chosen a threshold value of 50. Figure 2 shows the image after thresholding has been applied on the grayscale image.

Figure 2 Image after Thresholding

3.3 Morphological Processing

Morphological processing refers to the [7] examination of the geographical structure of a given image by probing it with structural elements or kernels of varying in size and shape. Ultimately, resulting in nonlinear image operations which are capable of exploring geometrical and topological structures. Therefore by applying sequence of such operations on an image makes certain feature visible and helps distinguish between meaningful information from irrelevant distortions by reducing it to a sort of skeletonization.

In general, there are many operation but [7,13] we used the basic four morphological operations operations which are Erosion, Dilation, Opening and Closing operations. We will now explore the individual operations in brief. Erosion causes increase in goal pixel and erodes away the foreground. The way it works is it slides a kernel through the entire image and then a pixel from the original image is considered a 1 only all pixels under the kernel is a 1.

Dilation basically [14] increases the white regions in the image foreground in which each pixel is considered to be 1 if atleast one pixel in 1 under the kernel. So, what happens is object area increases in our case increases the size of the object which decreases due to padding so increases the accuracy during recognition of the character.

Figure 3 Image after application of Opening and Closing Operations

Opening and closing operations are used to reduce the noise. [14] Opening is basically erosion followed by dilation while closing is the reverse. Opening reduces the noise by first erosion which reduces the image size and dilation which increases the area whereas Closing removes small holes inside the foreground objects, or small black points on the object. This allowed us to make the character more visible and clear so it is understandable.

Chapter 4

Dataset

4.1 Acquisition

One of the hardest problems to solve in deep learning has nothing to do with neural nets: it's the problem of getting the right data in the right format.

And as Bangla handwritten recognition is a relatively new area of research, a proper dataset was hard to come by . Some examples of Datasets are

Dataset	Basic letter	Numeric	Compound letter
CMATERdb 3.1.3.3[1]	15,103	6000	42248
ISI [3]	30966	23299	None
Bangla Lekha	98950	19748	47407

Table 1. shows some example of existing Dataset

For out recognition task ,we are using BanglaLekha-isolated[] dataset. It has total 1, 66,105 handwritten character images. It consists sample of 84 different Bangla handwritten numerals, basic characters and compound characters which was collected from different geographical location of Bangladesh and different age group. Where first 11 characters are vowel followed by 39 consonant characters then 10 characters are Bangla numerals and rest 24 are Bangla compound character [15].

The dataset was preprocessed in following ways

- Foreground and background were inverted so that images have a black background with the letter drawn in white.
- Noise removal was attempted by using the median filter.

- An edge thickening filter was applied.
- Images were resized to be square in shape with appropriate padding applied to preserve the aspect ratio of the drawn character.

Figure 4. A section of the Dataset

4.2 Preprocessing

As these data from dataset has different image size so it was a bit challenging to train them as we used fixed image size which is 50x50 pixel for each image. Therefore image pre-processing is needed for handling this big size of data. Then as the data was preprocessed a bit before , not much preprocessing was needed. We padded the image with zeros and used Bicubic interpolation to smoothen out the Characters(we augmented the data during the training in real time).The Augmentation Was Domein dataspace using elastic distortions [16] by width and height shifting. The range was kept 0.4 for this shifting we used Keras preprocessing API to preprocess the data .

Chapter 5

Fundamentals of Deep Learning

5.1 Neural Networks:

Neural network or artificial neural network is one of the most talked about topic nowadays . Neural network is a machine learning technique which enables a computer to learn from the provided data. The idea of Neural network in computing came from the way biological nervous system process information. [17]

In the computing world, neural networks are organized on layers made up of interconnected nodes which contain an activation function. These patterns are presented to the network through the input layer which then communicates it to one or more hidden layers. The hidden layers perform all the processing and pass the outcome to the output layer[17]. The figure 5 shows a simplest form of a Neural Network Where *x* is Input , w is weights and y is output .

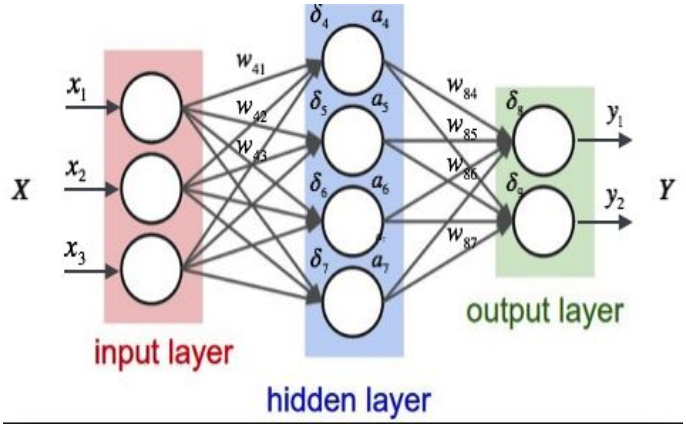


Figure 5 Basic Neural Network Structure

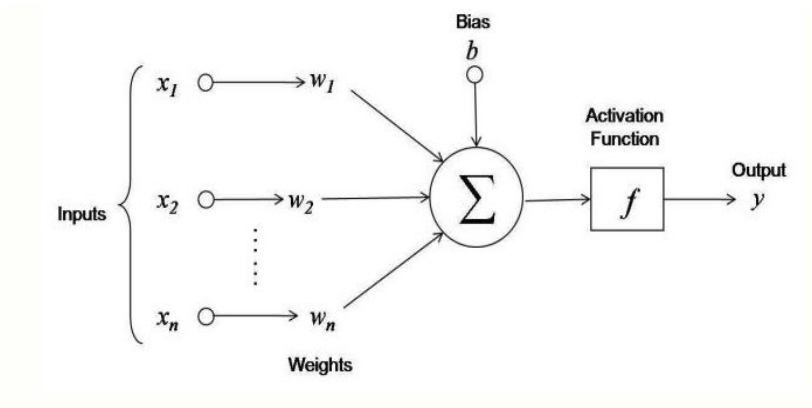


Figure 6. Skeleton structure of Basic Neural Network Structure

5.2 Deep Neural Network:

Deep Neural Network belongs to a type of machine learning algorithms that uses a connection of several layers of nonlinear processing units for changing and extracting features. Each successive layer make use of the output from the previous layer as input .It is used in both supervised (e.g., classification) and unsupervised (e.g., pattern analysis) Learning . It has the capability of learning several levels of representations that correspond to varying levels of abstraction .[18,19,20]

Deep learning architectures such as DNNs , RNNs and CNNs have been applied to fields including computer vision, natural language processing, recognition of speech , audio recognition, social network filtering, machine translation, drug design, medical image analysis ,bioinformatics, where the results produced are very much comparable and sometimes even better than human experts

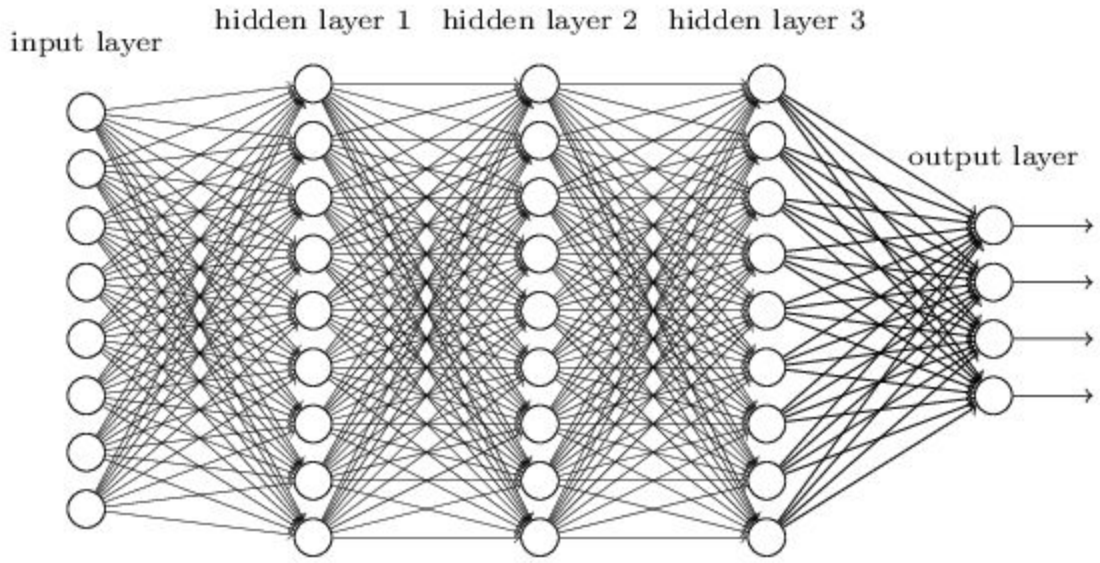


Figure 7 shows a deep neural network with 3 hidden layers

5.3 Convolutional Neural Networks(CNN)

In recent years image classification has turned out to be one of the most important part of computer vision , it represent a very significant part of our study work and life. Image classification is usually a mixture of several different components such as image preprocessing, image segmentation, key matching identification, and feature extraction.[21].Due to advancement in various image classification techniques , we can extract the information quicker and also apply it in scientific experiments,medical equipment, security systems ,traffic identification, face recognition , Character recognition and numerous other fields.

The Concept of Convolutional Neural Network was inspired my the mammalian visual System Arehture [21].in 1962 Hubel Wiesel proposed a Visual structure model inspired from the cat's visual cortex and for the first time The idea of receptive field has been proposed. In 1980 the Fukushima proposed The first hierarchical structure recognition used to process images. The first CNN was introduced by LeCun in [22] which was later improved in [23] . The paper owed a way of developing multilayer artificial neural network called LeNet-5 , that was used to classify Handwritten number . Just like other neural networks ,LeNet-5 has Several layers that can be trained with the backpropagation algorithm[24]. But due to the unavailability of large training data and computing power during that time , LeNet-5 cannot perform well on more complex problems, such as large-scale image and video classification. There have been

numerous methods since 2006 to overcome the difficulties encountered during the training of deep neural networks. But in 2012 Alexnet [25] , made a breakthrough on the Image classification task . It won the Imagenet challenge that year . after that there have been numerous proposed models to improve its performance such as ZFNet [26],VGGNet [27] and GoogleNet [28] .

Currently , the main aspect where optimization of a CNN is focused on afe the design of Convolutional and pooling layers , loss function, the activation function,regularization and Convolutional neural network can be applied to problems that are practical .

CNN's are multilayer artificial neural networks , which are specifically designed to handle two-dimensional data input data .Every layer in the networks is made up of multiple two dimensional planes, and every single plane consists of several independent neurons, two adjacent layers of neurons connected to each other, and doesn't have a connection between the neurons of the same layer . the inspiration of CNN came from early time delay neural networks (TDNN) [29],By sharing the weights in the time dimension,TDNN reduces the computational complexity in the network training process and it is can efficiently preprocess speech signals and Time sequence signal ,CNNs are more similar to a biological neural network as they use a weight sharing network structure and just by changing the width and depth of the network , the models capacity can be adjusted . This also has a strong assumption for natural images(statistical smoothness and local Correlation) . So to summarize , CNN is can reduce the computational complexity of a network model , has much less number of connection and weight parameters , and are mode easy to train then a fully connected neural network of the same size

CNNs consists of mainly 3 types of layer namely Convolutional layer, pooling layer and fully-connected layer. Fig. 8 shows the architecture of LeNet 5-[22] which is introduced by Yann LeCun.

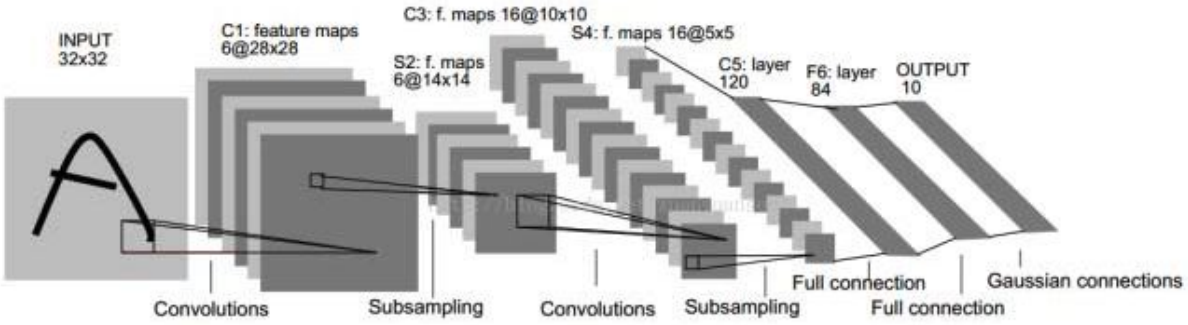


Figure 8 LeNet 5 Architecture [22]

Convolution Feature Extraction

Natural images have its inherent characteristics, that is, for a part of the image, its measurable attributes and different parts of the equivalent. This implies that the same learning feature can be used for all positions on the image using the features learned in a specific section of the image. For large-size image recognition problems, firstly, a small piece of local data is arbitrarily chosen from the image as a training sample. After that, the small samples are used to learn some features. Then, these features are used as filters while the original images are used for convolution operations. As a result, original image at any position is obtained on the different characteristics of the activation value. Xlarge is a large image with a resolution of r × c. Firstly, a small sample of xx is taken from large. The k features and activation values f (W) are obtained by training sparsely from the encoder (1) x small + b (1)),where W (1) and b (1) are the trained parameters. The corresponding activation value fs (W (1) xsmall +b (1) for each x × the size of xs in xlarge), is then calculated. Upon further use of the activation value of x small and convolution of these activation values fs, the feature map of k × (r - a + 1) × (c - b +1) convolution is attained. The Two-dimensional convolution calculation diagram is illustrated in Fig. 9. A raw input image with a resolution of 128x128, can contain 200 8x8 size feature fragments of the image which have been obtained by pre training. These 200 feature fragments are then used to convolve each 8 ×8 small block region in the original image. Each feature fragment can get a convolutional feature map of 121× 121. Finally, the whole image can be obtained 200 × 121× 121 convolutional feature map. Fig. 9 illustrates the schematic diagram of two dimensional convolution operation.

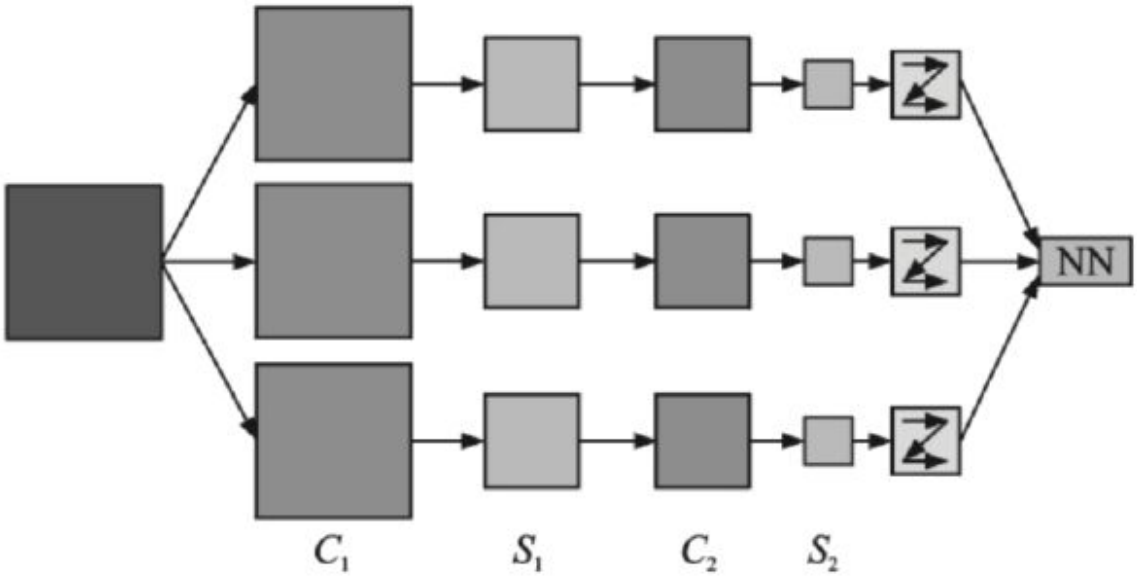


Figure 9 Simplified convolution neural network structure.[22]

Pooling Operations

The final classification result is obtained by extracting the features extracted from the convolution layer into the classifier for training. Ideally, all the features extracted from the convolution layer can be input directly into the classifier. However, a very large computational overhead in involved in this case, particularly for large-size high-resolution images. For instance, if an image sample with an input of 96× 96 size considered, then the convolution operations are assumed to have been performed using 200 8 × 8 size convolution cores in the convolution layer. Each Convolution kernel gives output of one (96 - 8 + 1) × (96 - 8 + 1) = 7921 dimension, while the final convolution layer provides output of a feature vector of 7 921 × 200 = 1 584 200 dimensions. A very large computational resource is required and a serious over-fitting problem is encountered while attempting to input such high dimensional features into the classifier. However, it is highly likely that the feature obtained in a local region of the image, is applied equally in another local area. This happens due to the image containing a “static” attribute. Therefore, aggregate statistical operations can be executed on the characteristics of the different locations in a local area of the image. This is referred to as "pooling". For example, calculate the maximum (or average) of a convolution feature in the local area, called the maximum pool (or average pool). Specifically, after the convolution feature is obtained, the convolution feature is distributed into a plurality of m × n size disjoint areas. It is assumed here that the pooled area size is m × n. After that, the pooling operation is performed

on these areas. The characteristic map after pooling is shown in. Fig. . To obtain a pooled feature map, the maximum pooling is implemented on a 4-block non coincident sub-region using a 3 × 3 size window. The selection of continuous range in the image as pooled area, and the use of only the convolution features generated by same mplicit neurons, causes these pooled feature units to have translation invariance. This mean the same pooling feature can still be obtained and the classifier can still provide the same classification result, even if the object in the original image produces a small translation. The problem of overfitting can be avoided due to the fact that these statistical features can greatly reduce the dimension of the eigenvector. Moreover, the computational effort required by the training classifier is reduced, hence expanding the training data effectively.

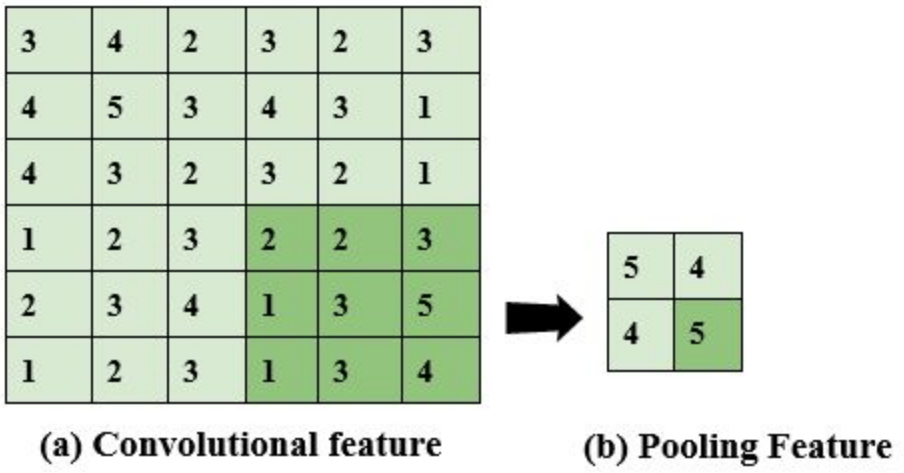


Figure 10. Illustration of two-dimensional convolution operation and The maximum pool operation diagram where (a) Convolutional feature (b) Pooling features

Fully-connected Layer

Usually the layer that classifies in a CNN is one or more fully-connected layers .They take all neurons in the previous layer and connect them to every single neuron of current layer. The Fully-connected layers flattens the 2D matrix removing the perceived spatial information . the end fully-connected layer is followed by an output layer . Softmax regression is one of the mostly used for classification tasks , because of its generating a well-performed probability distribution of the outputs. Sometimes SVM is also used in combination with CNNs to solve different classification tasks[30].

Optimization

Optimization is a crucial part of a CNN, Optimization algorithms helps us to minimize(or in rare cases maximize) the Loss function and the way they do this is by using Gradient values. Gradient Descent is the most important technique and the foundation of how CNNs are optimized. Gradient descent is majorly used to update weights in a Neural Networks towards a direction that an minimize the loss Function . the Figure 11 shows how the value of weights are updated to reduce the error function E(x) .

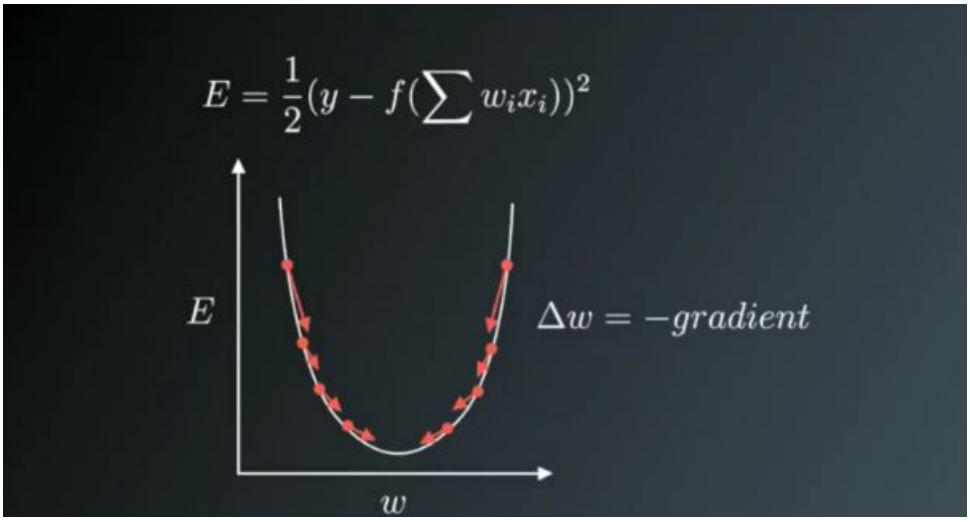


Figure 11 Optimization Curve[31]

The traditional Batch Gradient Descent will calculate the gradient of the whole Dataset but performing only one update , which makes is very slow and hard to control for datasets which are very large to fit in memory . the size of the weight update depend on the learning rate $-\eta$

SGD (Stochastic Gradient Descent) is similar to Gradient Descent but performs weight updates for each training example . It is usually much faster techniques as it performs one update at a time .

$\theta = \theta - \eta \cdot \nabla J(\theta; x(i); y(i))$, where {x(i) ,y(i)} are the training examples.

But the problem with SGD is that because of frequent updates and fluctuation it the end it complicates the convergence to the actual minim and will keep ove shooting due to the frequent updates.

Another common optimizer that is used is **RMSprop (Root Mean Square Propagation)**,[31] The RMSprop optimizer is similar to the gradient descent algorithm with momentum.In the standard gradient descent algorithm, you would be taking larger steps in y- direction and smaller steps in the x-direction.e RMSprop optimizer restricts the oscillations in the vertical direction. Therefore, increasing learning rate and the algorithm could take larger steps in the horizontal direction converging faster. The value of momentum is denoted by γ and is usually set to 0.9.

$$\mathbf{V(t)=\gamma V(t-1)+\eta \nabla J(\theta)}.$$

Recently **Adam** Optimizers[32] r frequently being used in CNN . Adam is an adaptive learning rate optimization algorithm that's been designed specifically for training deep neural networks. Adam can be looked at as a combination of RMSprop and Stochastic Gradient Descent with momentum. It uses the squared gradients to scale the learning rate like RMSprop and it takes advantage of momentum by using moving average of the gradient instead of gradient itself like SGD with momentum. Adam is an adaptive learning rate method, which means, it computes individual learning rates for different parameters.

Chapter 6

Architectures of Convoluted Neural Network

Image classification has been one of the most researched topics in the recent years and there have been several different architectures for Convolutional Neural network , for receiving higher accuracy on the Image classification task.

Image net is a large visual database designed for use in visual object recognition software research with over 14 million URLs of images have been hand-annotated by ImageNet to indicate what objects are pictured. Every year they arrange a competition (ILSVRC-ImageNet Large Scale Visual Recognition Challenge" from 2010 onwards) where participants are provided with 150,000 photographs, collected from flickr and other search engines, hand labeled with the presence or absence of 1000 object categories[33] .

Below in Figure 12 are some CNN architectures that have won the competition with their respective percentage error.

- AlexNet
- VGGNet
- GoogLeNet
- ResNet

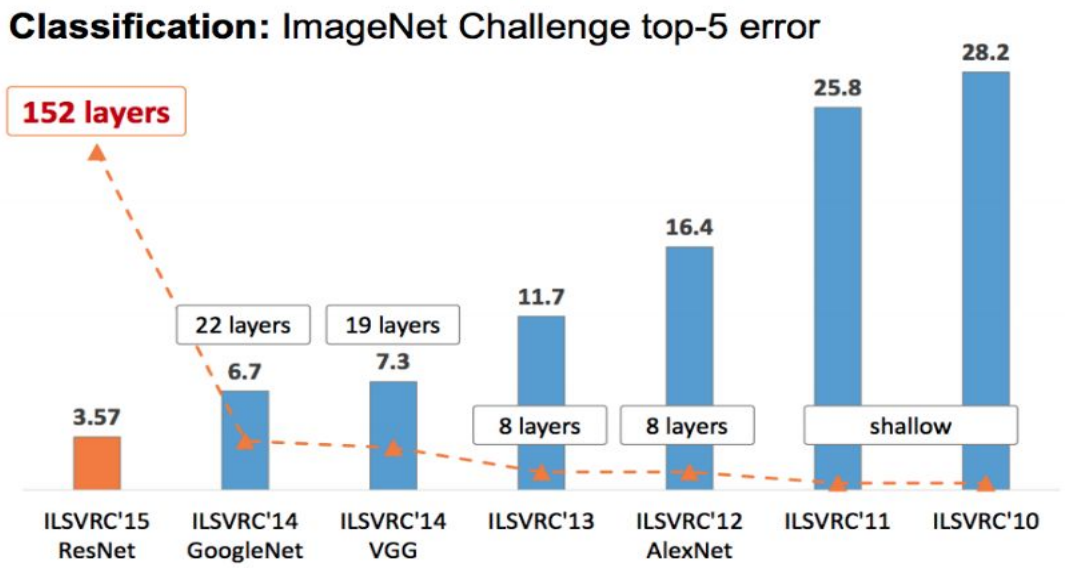


Figure 12 ImageNet Challenge

taken from: (https://medium.com/@RaghavPrabhu/cnn-architectures-lenet-alexnet-vgg-googlenet-and-resnet-7c81c017b848)

6.1 AlexNet

It was thefirsth CNN architecture to recieve major improvements on the ImageNet dataset

And it won the competition on 2012 [34] . Figure 13 shows the architecture of AlexNet.

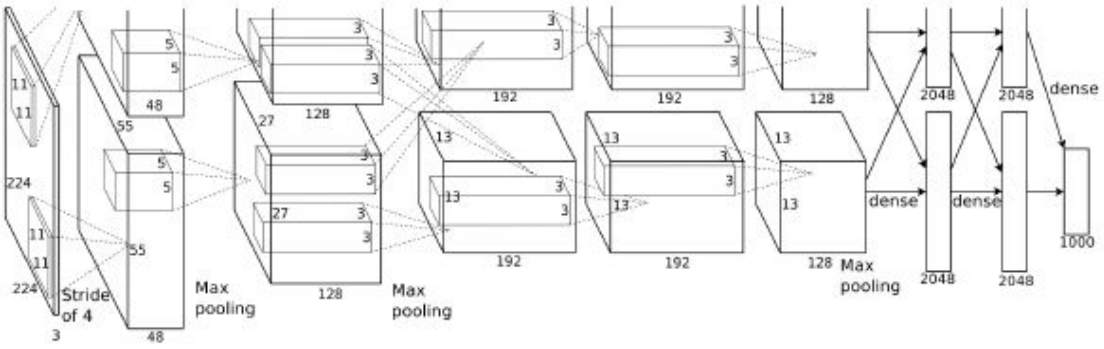


Figure 13 shows the architecture of AlexNet.[34]

Conceptually, The Architecture is is identical to the architecture of LeNet-5 .Convolutional Layers are followed by pooling layers several times finally a fully connected network is applied .

It is described in the table below .

#	Type	Filters @ Patch size /stride	Parameters	FLOPs	Output size
	Input				3@ 224x224
1	Convolution LCN	96@ 11x11x3 /4	34944	211 M 12 M	96 @ 55x55 96 @ 55x55
2	Max Pooling	3x3 /2	0	301 K	96 @ 27x27
3	Convolution LCN	256@ 5x5x48 /1	3056	448 M 3 M	96 @ 27x27 256 @ 13x13
4	Max Pooling	3x3 /2	0	50K	256 @13x13
5	Convolution	384@ 3x3x256 /1	885120	299 M	384 @ 13x13
7	Convolution	384@ 3x3x192 /1	663936	224 M	384 @ 13x13
9	Convolution	256@ 3x3x256 /1	442624	150 M	256 @ 13x13
10	Max Pooling	3x3 /2	0	50 K	256 @ 6x6
11	FC	4096 neurons	37752832	75 M	4096
12	FC	4096 neurons	16781312	34 M	4096
13	FC	1000 neurons	4097000	8 M	1000
Σ			60 965 224	3300M	11 22 568

Table 2. Architecture of AlexNet

6.2 VGG-16

VGG 16 is another very well known network [35] short for **V**isual **G**eometry **G**roup it was developed in Oxford . it has won the (ILSVRC) in 2014 for localization. The network has 16 layers which can learn parameters . the the architecture falls in similar category as AlexNet but it uses only 3x3 filters and is much deeper. Figure 14 below shows a visualization of the Architecture (note that all convolutional layers use SAME padding) . and a detailed description of the network is given in table 3

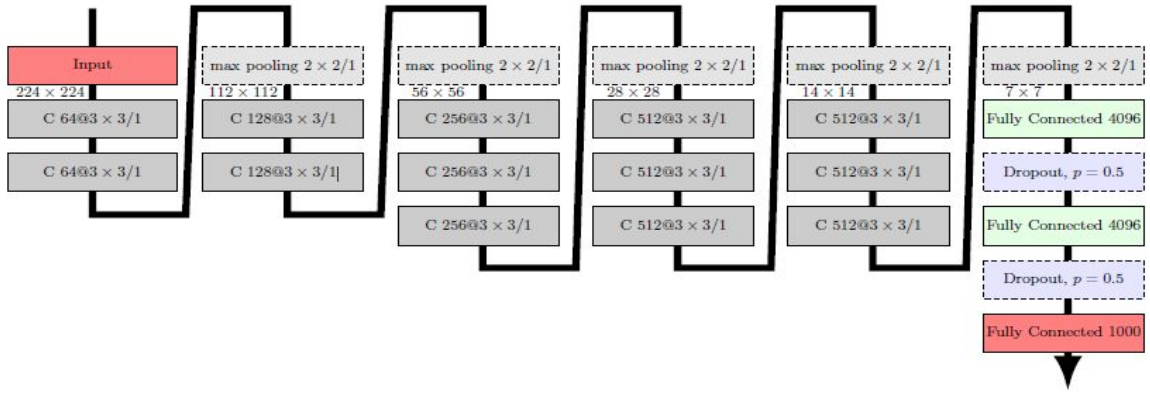


Fig 14 Visual description of VGG16[35]

#	Type	Filters @ Patch size /stride	Parameters	FLOPs	Output Size
	Input				3 @ 224x224
1	Convolution	64 @ 3x3x 3/1	1 792	186M	64 @ 224x224
2	Convolution Max pooling	64 @ 3x3x 64/1 2x2 /2	36 982 0	3712M 2M	64 @ 224x224 64 @112x112
3	Convolution	128 @ 3x3x 64 /1	73 856	1856M	128@ 112x112
4	Convolution Max pooling	128 @ 3x3x128/1 2x2 /2	1 47 584 0	3705M 1M	128@ 112x112 128@ 56x56
5	Convolution	256 @ 3x3x128 /1	2 95 168	1853M	256@ 56x56
6	Convolution	256 @ 3x3x 256/1	5 90 080	3703M	256@ 56x56
7	Convolution Max pooling	25 6@ 3x3x 256/1 2x2 /2	5 90 080 0	3703M <1M	256@ 56x56 256@ 28x28
8	Convolution	512 @ 3x3x 256/1	11 80 160	1851M	512@ 28x28
9	Convolution	512 @ 3x3x 512/1	23 59 808	3701M	512@ 28x28
10	Convolution Max pooling	512 @ 3x3x 512/1 2x2 /2	23 59 808 0	3701M <1M	512@ 28x28 512@ 14x14
11	Convolution	512 @ 3x3x512 /1	23 59 808	925M	512@ 14x14
12	Convolution	512 @ 3x3x512 /1	23 59 808	925M	512@ 14x14
13	Convolution Max pooling	512 @ 3x3x 512 /1 2x2 /2	23 59 808 0	925M <1M	512@ 14x14 512@ 7x7
14	FC Dropout	4096 neurons	102 764 544 0	206M 0	4096 4096
15	FC Dropout	4096 neurons	16 781 312 0	35M 0	4096 4096
16	FC	1000 neurons	4 097 000	8M	1000
Σ			1 38 357 544	31000M	15245800

Table 3 shows the description of the architecture

The Table contains only the layers that has trainable parameters .All convolutions are zero padded to prevent size changes and use ReLU activation functions. The channels mean is subtracted from each pixel as a preprocessing step. Dropout is only calculated during training time. The dropout probability is 0.5.

6.3 GoogleNet, Inception V2 and V3

As the number of trainable parameters an operations increases t become problematic to apply to thousands of images, so In order to solve this problem maintaining the classification quality, The researchers from Google developed **GoogleNet** which contains Inception modules[36]. It won the (ILSVRC) in 2014 for classification.The Inception module essentially only computes 1 x1 filters, 3x3 filters and 5 x5 filters in parallel, but applied bottleneck 1 x1 filters before to reduce the number of parameters. The Figure 15 and 16 shows the Inception module and the Architecture of GoogleNet.

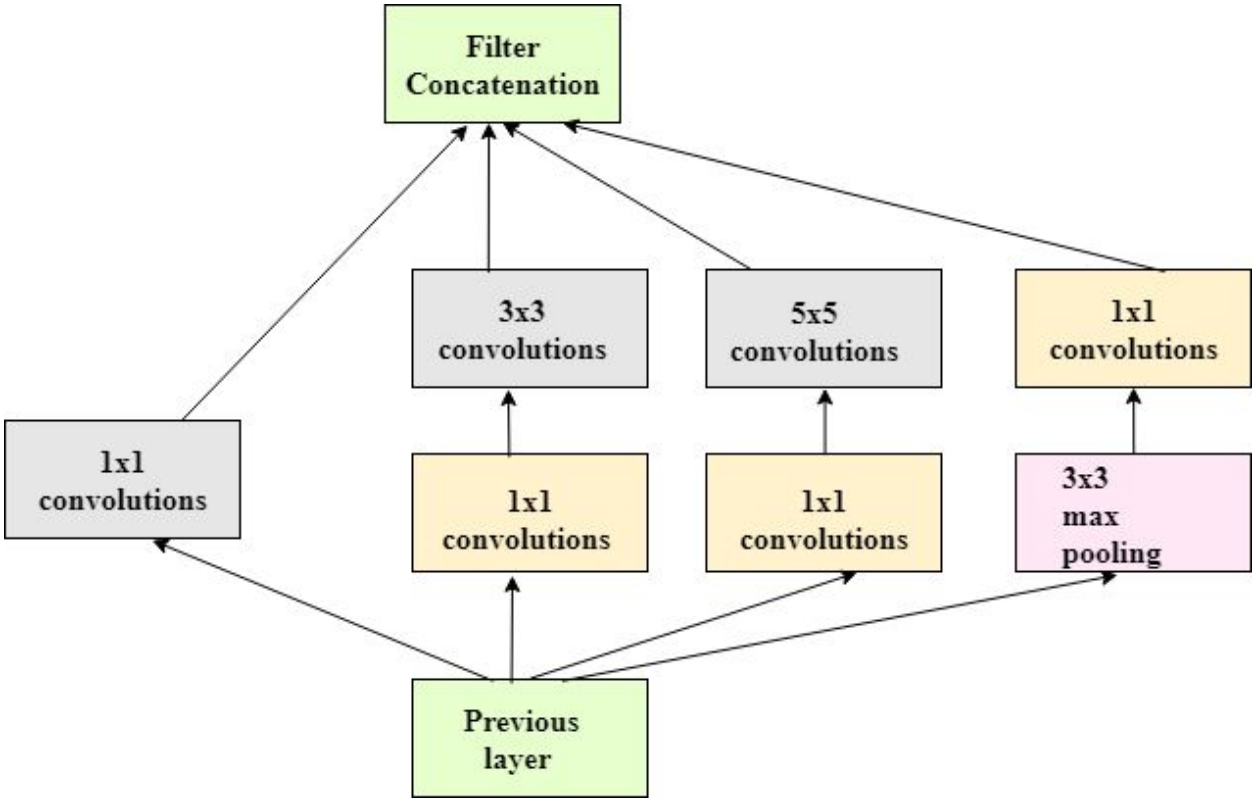


Fig 15 Inception Module

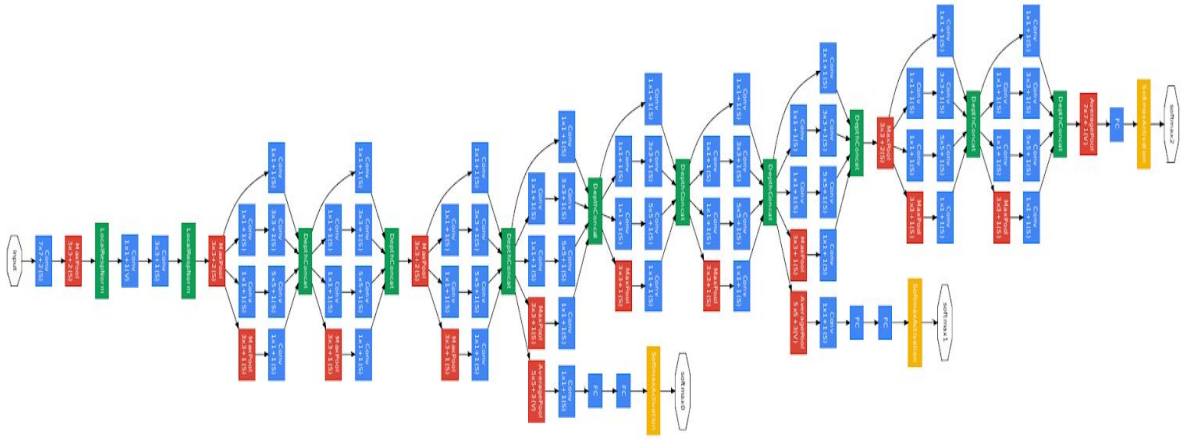


Figure 16 GoogleNet [https://joelouismarino.github.io/blog_posts/blog_googlenet_keras.html]

In **Inception V2** , 5x 5 filters and replaced them by two successive layers of 3x3 filters. Figure 17 shows the inception V2 module .

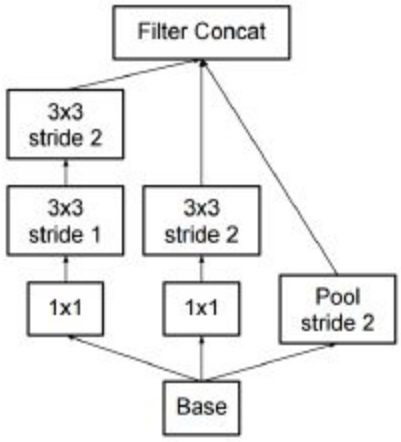


Figure 17 Inception V2 module

Inception v3 introduced Batch Normalization to the network .

6.4 Resnet

Even though previously , increasing the layers would increase the accuracy of the networks , as seen in LeNet-5, AlexNet, and VGG16 , it is not always true . Even though we can build a deep neural network with more than 100 layers theoretically but in reality, they are hard to train due to the problem of vanishing gradient . During each iteration of training a neural network , the weights update in proportion to the partial derivative of the error function with respect to the current weight so if the gradient becomes to small then the updates to weights will be very insignificant and may completely stop the neural network from further training at a point .

To solve this problem Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun introduced the concept of Residual Networks (ResNets) in their paper [37] . The RestNEtss allows deeper Neural networks to overcome the vanishing gradient problem . In 2015 a 152 layer ResNet won the (ILSVRC) by a considerable margin with a error rate of 3.57% which is about alf the loss of GoogleNet that wont the year before .The research found that diving a Neural Networks into 3 layer chunks and passing the input into each chunk straight through to the next chunk, along with the residual output of the chunk minus the input

to the chunk that is reintroduced, helped eliminate much of this disappearing signal problem . It didn't require and additional parameters or changes to the learning algorithm were needed . So ResNet breaks down a a deep Neural Networks in small chunks of network that are connected through skip connections to form a bigger network . Figure 18 shows the Residual block in ResNet and Figure 19 shows the architecture of a 34 Layer Resnet .

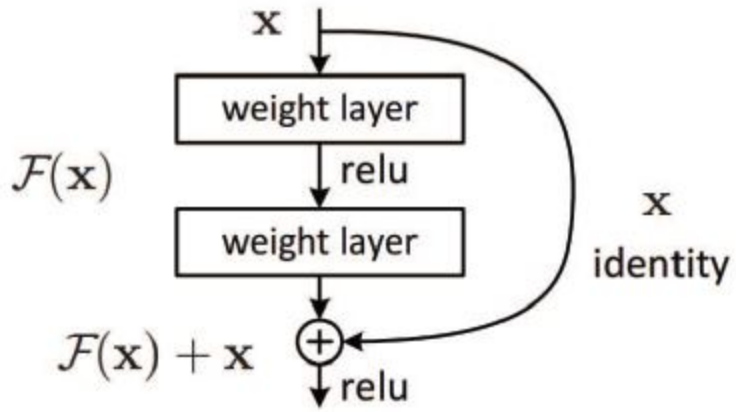


Figure 18 A Residual block of ResNet[37]

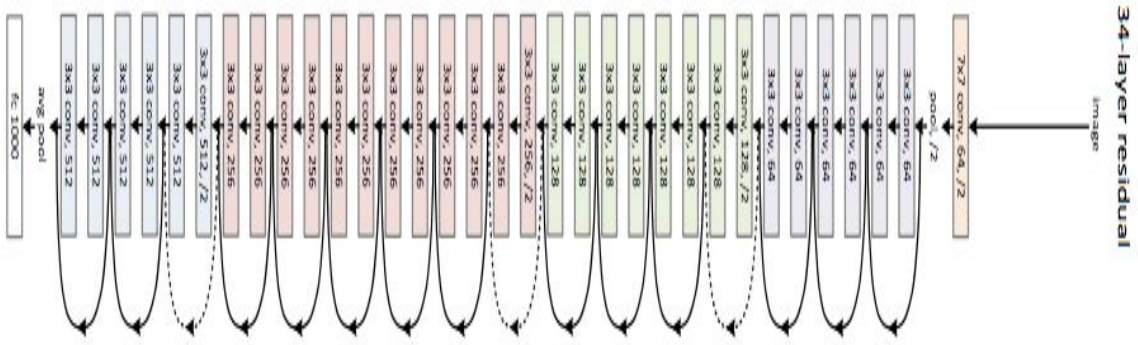


Figure 19 ResNet 34[37]

6.5 Inception V4

In 2016 Inception v4 and Inception-ResNet were introduced in the same paper [38]. It consists of 4 main building blocks :The stem,Inception A, Inception B and Inception C. The authors mentioned Inception-v4 is a deeper, wider and more uniform simplified architecture than Inception-v3.The stem, Reduction A and

Reduction B use max-pooling, whereas Inception A, Inception B and Inception C use average pooling. The stem, module B and module C use separable convolutions. Figure 20 shows the architecture of the following :Inception A, Inception B and Inception C blocks , Figure 21 shows Reduction A and Reduction B blocks . Figure 22 shows the Schema for Inception V4 .

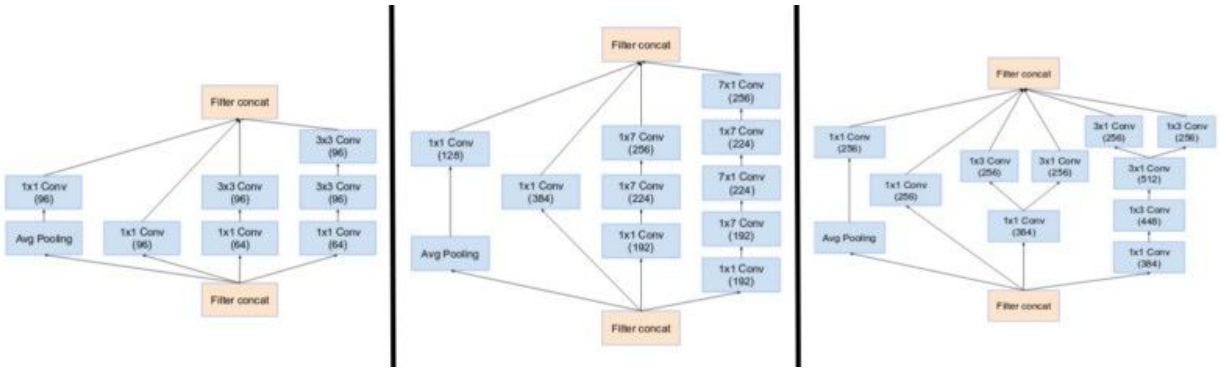


Figure 20 Inception A, Inception B and Inception C blocks respectively[24]

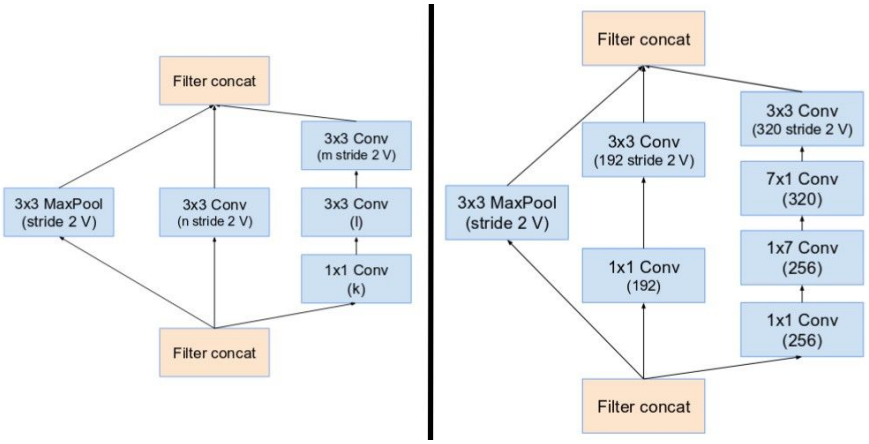


Figure21 Reduction A and Reduction B blocks[24]

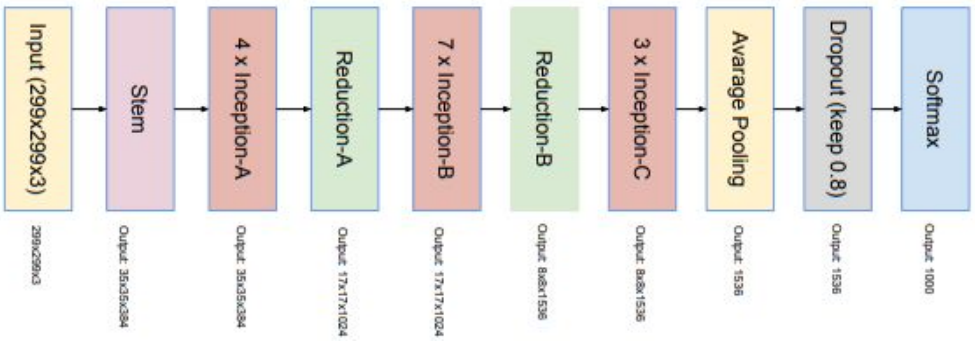


Figure 22 Schema for Inception V4[24]

The Table below shows the text form of the network .

#	x	Type	Parameters	Output size
		Input		3 @ 299 x299
1		Stem	605728	384 @ 35x 35
2	4x	Inception A	317632	384 @ 35x 35
3		Reduction A	2306112	1024 @ 17 x 17
4	7x	Inception B	2 936 256	1024 @ 17 x 17
5		Reduction B	2 747 392	1536 @ 8 x 8
6	3x	Inception C Global Average Pooling Dropout(p=0.8)	4 553 088 0 0	1536 @ 8 x 8 1536 @ 1 x 1 1536 @ 1 x 1
7		Softmax	1 537 000	1000
Σ			42 679 816	

Table 4 : Inception V4 Network

6.6 DenseNet:

Densely Connected Convolutional Networks or DenseNet[39] is an extension to ResNet Architecture .In Resnet the vanishing gradient in deeper Neural Networks was solved by adding identity blocks that merges previous layer into a future layer so by adding additive merges we are forcing the network to learn residuals (errors i.e. differ between some previous layer and current one). In contrast, DenseNet paper proposes concatenating outputs from the previous layers instead of using the summation.

In recent works we have seen that we can make dper Convolutional networks more accurate and efficient to train if they have sorter connection between layers close to the input and those close to the output . DenseNet is based on this ,it connects each layer to every other layer in a feed-forward fashion . feature maps of all previous layers are used as inputs for each layer .

Not only it solves the vanishing gradient problem ,bt also strengthen feature propagation, encourage feature reuse, and substantially reduce the number of parameters.

DenseNetrequres less number of parameters compared to traditional CNN's as there is no need to relearn redundant feature maps . it also has improved flow of information and gradients throughout the network, which makes them easy to train . Furthermore, it is also observed that dense connections have a regularizing effect, which reduces overfitting on tasks with smaller training set sizes. Both DensseNEt and ResNets concatenate features from different layers, but Dnsenets are simpler and more efficient.

Figure 23 shows a visualization of DenseNet and table2 shows the architecture in text form .

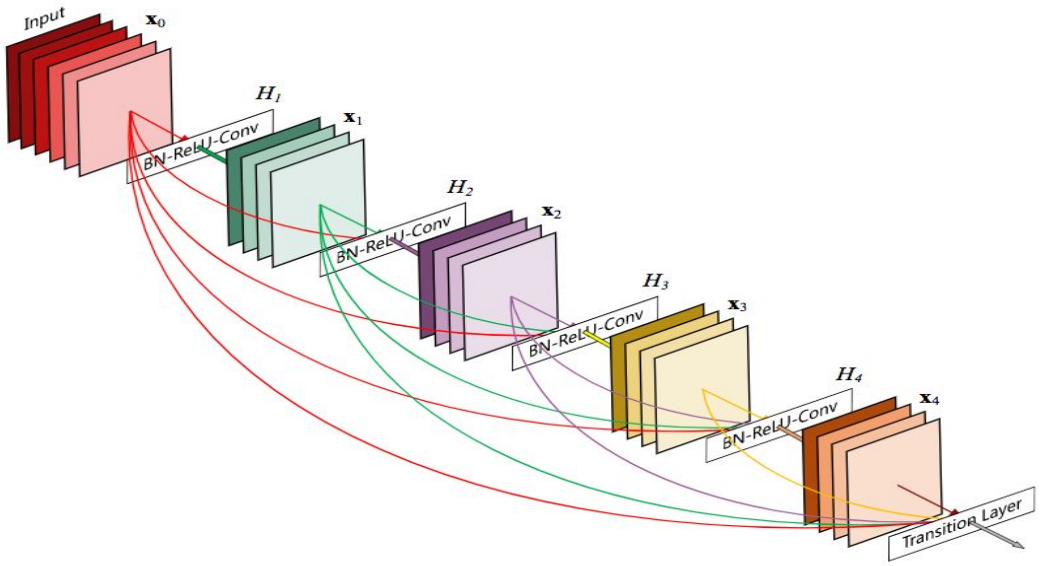


Figure 23 shows a visualization DenseNet Architecture (source : https://arthurdouillard.com/post/densenet/)

Layers	Output Size	DenseNet-121($k = 32$)	DenseNet-169($k = 32$)	DenseNet-201($k = 32$)	DenseNet-161($k = 48$)
Convolution	112 × 112	7 × 7 conv, stride 2			
Pooling	56 × 56	3 × 3 max pool, stride 2			
Dense Block (1)	56 × 56	[1 × 1 conv, 3 × 3 conv] × 6	[1 × 1 conv, 3 × 3 conv] × 6	[1 × 1 conv, 3 × 3 conv] × 6	[1 × 1 conv, 3 × 3 conv] × 6
Transition Layer (1)	56 × 56	1 × 1 conv			
	28 × 28	2 × 2 average pool, stride 2			
Dense Block (2)	28 × 28	[1 × 1 conv, 3 × 3 conv] × 12	[1 × 1 conv, 3 × 3 conv] × 12	[1 × 1 conv, 3 × 3 conv] × 12	[1 × 1 conv, 3 × 3 conv] × 12
Transition Layer (2)	28 × 28	1 × 1 conv			
	14 × 14	2 × 2 average pool, stride 2			
Dense Block (3)	14 × 14	[1 × 1 conv, 3 × 3 conv] × 24	[1 × 1 conv, 3 × 3 conv] × 32	[1 × 1 conv, 3 × 3 conv] × 48	[1 × 1 conv, 3 × 3 conv] × 36
Transition Layer (3)	14 × 14	1 × 1 conv			
	7 × 7	2 × 2 average pool, stride 2			
Dense Block (4)	7 × 7	[1 × 1 conv, 3 × 3 conv] × 16	[1 × 1 conv, 3 × 3 conv] × 32	[1 × 1 conv, 3 × 3 conv] × 32	[1 × 1 conv, 3 × 3 conv] × 24
Classification Layer	1 × 1	7 × 7 global average pool			
		1000D fully-connected, softmax			

Table 1. DenseNet architectures for ImageNet. The growth rate for the first 3 networks is $k = 32$, and $k = 48$ for DenseNet-161. Note that each "conv" layer shown in the table corresponds the sequence BN-ReLU-Conv.

Table 5 DenseNet Architecture for ImageNet.

(Source : https://towardsdatascience.com/densenet-2810936aeebb)

There are two main components in DenseNet

Dense Block : these block contains the group of layers that connects to all the previous layer , it has

- Batch Normalization
- ReLU activation
- 3x3 COnvolution

Transition Layer: as Dense concatenates all the feature maps instead of residuals like esNet, it would be problematic to concatenate feature maps of different sizes , so the feature maps of each layer has to be the same size . However down-sampling is essential to CNN. Transition layers between two dense blocks assure this role. It has

- Batch Normalization
- 1x1 convolutions
- Average pooling

6.7 Choosing a model

In our thesis, we have tested three models of different style of architecture namely VGG 16 , Resnet-50 and DenseNet-121 and comparing their accuracy for our dataset we have decided to chose DenseNet121 as this gives us higher accuracy. Moreover, for a fair test the data was pre processed the same way for all four architectures and each of them had same learning rate with same number of epochs .

Chapter 7

System Implementation

7.1 System Architecture

In the proposed system, we first capture a picture of the handwritten character using a camera. The handwritten character is written in a white background with a black marker. The image is then processed using ***OpenCV*** and ***Python***. The image is first converted to grayscale/binary. The image we obtain from the camera will be a white background with black writing and we need to invert the image as the images in the dataset being used are black background with white character after this thresholding is applied on the image. The image size needs to be increased so this done by padding the images however in doing this the character becomes small thus the accuracy will be low so it is then increased by applying bicubic interpolation which increases the size of the character and smoothes out any irregularity. The image is then ready for character recognition. For Character recognition , ***Deep Neural Network*** was applied on the image followed by, ***VGG 16***, ***Resnet 50*** and ***DenseNet 121*** which are modules of ***Convolutional Neural Networks*** where used for the determination and recognition of the character. Each of the method above is tested and the accuracy is recorded for each method and finally the method with the most accuracy is implemented. Each character is assigned with its own unique class value. This class value is then passed into the ***Arduino (Atmega2560 chipset)*** microcontroller which controls six ***servo motors***. The braille patterns are of 2x3 sizes so we have used six servos. These servos are in turn are connected to six pins which can be made to move up and down. When a hand is placed on the top the user will be able to feel a sensation allowing them to be able to tell the character that the combination resembles. Here, the microcontroller has been programmed with multiple switch and case method. The Arduino receives the Class value as input which is the condition for the switch case method. Furthermore, each case contains the braille pattern for the corresponding character Class value. As a result the pins are either made to rise by the servo as dictated by the code. The system also consists of a LED which will show the corresponding character. The entire architecture is summarized in the flow diagram in figure 24.

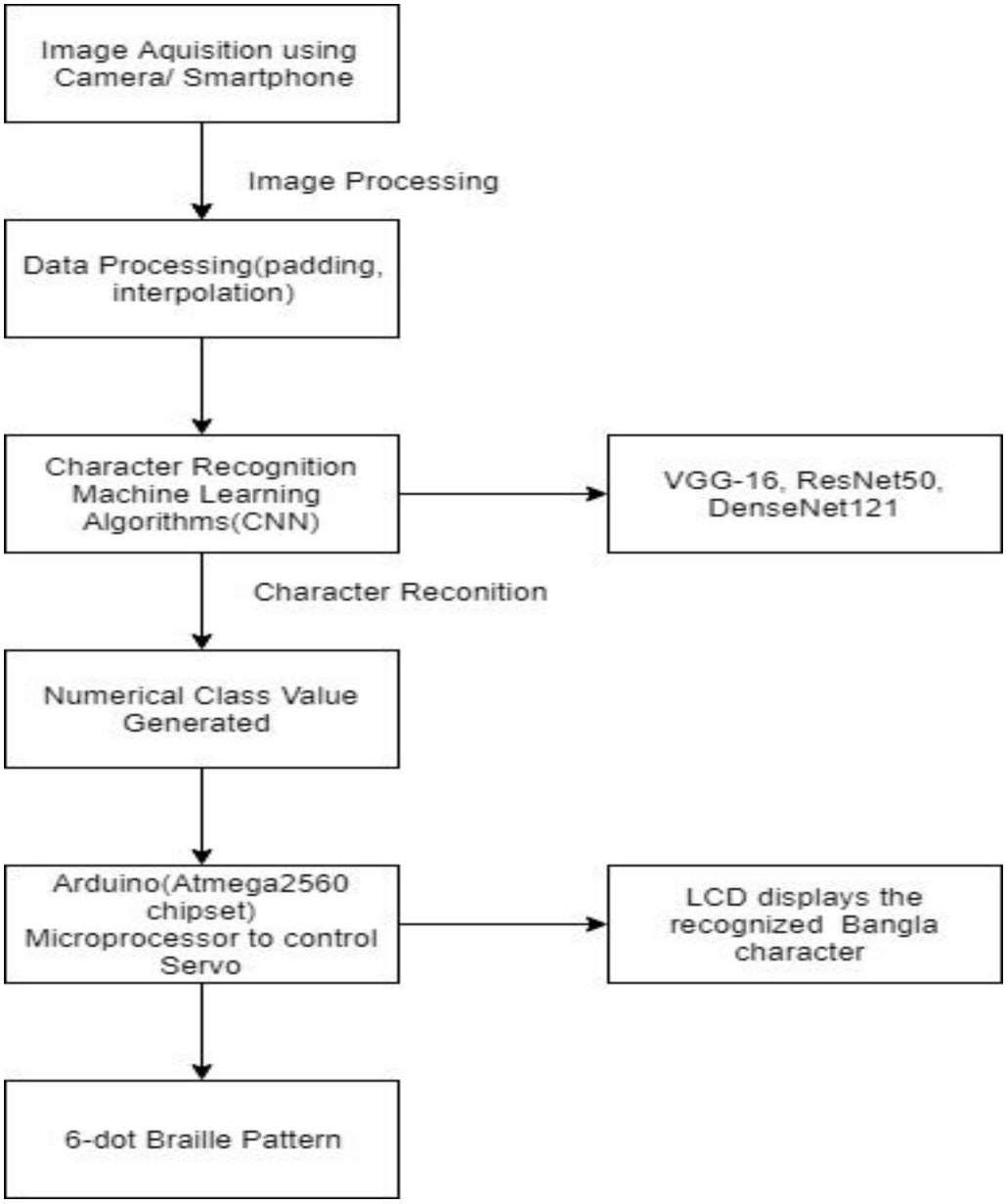


Figure 24 System Architecture of the Implemented System

7.2 Software Implementation

OpenCV. OpenCV (Open Source Computer Vision Library) free for both academic and commercial use. It has C++ and is released under a license of BSD. It is contained with the interfaces of python and java. Moreover, it supports Windows, Linux, Mac OS, iOS and Android. OpenCV was intended for

computational efficiency and with a solid spotlight on real-time applications. Since, it is Written in advanced C/C++, the library can exploit multi core processing. Then again it is facilitated with OpenCL, so that it can take advantage of the hardware acceleration of the underlying heterogeneous compute platform. In our thesis we used openCV to process the raw data so it is more understandable and easy to use. Techniques such as Grayscale, Thresholding(cv.threshold) were applied on the image to prepare it as the input for deep learning algorithm.

Python. Python is a broadly adopted high level, universally useful, translated, dynamic programming language. The design ideology of python highlights code readability, and the syntax of it enables developers to express codes in less lines which would be difficult in languages such as C++ or Java. The language grants the constructs which are intended to enable clear programs on both large and small scales. Python supports different types of programming archetypes which incorporates object-oriented, imperative and functional programming or procedural styles. It is contained with a dynamic type system and automatic memory management included with a large and comprehensive standard library. For every processing techniques python programming language has made us able to write short code fragments. With all these, it facilitated us to develop a multi-level processing mechanism. Thus this programming language was very advantageous in the digital processing of the stock images through writing very simple and easily understandable python codes.

Tensorflow. It is an open source software library which is used for high performing numerical computation. The flexible architecture of TensorFlow enables simple distribution of computation across a variety of platforms such as CPUs, GPUs, TPUs and also from desktops to the clusters of servers to mobile and edge devices. TensorFlow is developed by the researchers and engineers of the Google Brain team within Google's AI organization. It features with powerful support for machine learning and deep learning and within many other scientific domains the flexible computation core is used. TensorFlow provides a variety of different tool kits that allow you to write code at your preferred level of abstraction.using the tf.layers method abstract you can play with the layers of a neural net. You can build a model and evaluate the model performance using the tf.metrics method. The most widely used level is the tf.estimator API, which allows you to build (train and predict) production ready models with easy. The estimator API is insanely easy to use and well optimised. Although it offers less flexibility, it has all that is needed to train and test your model. The basic data type in this framework is a Tensor**.** A Tensor is an N-dimensional array of data.

Keras. It is a high leveled neural networks API which is written in python. Keras has the capability to run on top of TensorFlow, CNTK, or Theano. With a focus to enable fast experimentation it was developed.

- Having a user friendly environment, modularity and extensibility it allows easy and fast prototyping.
- Keras is supported with both convolutional networks and recurrent networks as well as the combinations of the two.
- It runs smoothly on CPU and GPU.

7.3 Braille Implementation

The Braille design is based of the 6-dotted braille pattern system. For the construction of the braille we have used six servo motors. The servos motors are placed in 3x2 pattern in a box where each individual servo is controlled by the Arduino. The component used in designing the braille system are

1. Arduino Mega(ATmega2560 chipset)
2. Servo Motor
3. 8V double cell Lithium Polymer Battery
4. 7.5V 1100mAh Lipo Battery(Two Cell)
5. DC-DC Buck Converter(LM2596)
6. Liquid Crystal Display(LCD)
7. Wires

Arduino Mega(ATmega2560 chipset). Arduino Mega 2560 is a microcontroller board based on ATmega2560 chipset [40]. It contains a total of 54 digital input/output pins [40]. Out of which 14 of those are digital input/output pins can be used as PWM(Pulse Width Modulation) outputs [40]. In addition. it also has 16 analog inputs, 4UART's which are hardware serial ports, a 16Mhz crystal oscillator and a USB port for uploading codes into the microcontroller. Additionally, it contains a power jack, ICSP header and reset button [40] to refresh the given progress into this microcontroller. In the implemented system the Arduino Mega is used to used to control the servo motors The choice behind using the Arduino is because Programming language used for this chip is Arduino Software (IDE), based on C++. Therefore, being relatively inexpensive and it also simplifies the process of working with microcontrollers.

Servo Motor. Servo Motor are linear actuators or rotary systems [41] which allows precise controls over angular position, velocity and acceleration. For position feedback it has suitable motor coupled to a sensor[41]. The position feedback allows the servo to control motion and final position of the rotor. A servo motor typically has three wire connections a power, ground and signal.

DC-DC Buck Converter. This is a step down power module which consists of the LM2596 regulator. The LM2596 is a monolithic integrated circuit that is and ideal and easy design for a buck converter capable of driving loads of 3A with outstanding line and load regulation. It has a operation frequency of 150Hz allowing smaller size filter component. In our design the LM2569 is perfect in regulating the flow of current to each of the six servo so that the servo operate properly.

System Overview. The core constitution of the braille are six servo motors that are used to form the formation of 3x2 braille pattern of Bangla Letters. The servo motors were controlled by rotating the rotors up to 90 degrees. After the character has been successfully recognized by the desired and suitable CNN architecture a numerical value is generated that corresponds to the generated character. This numerical value is now the input of the system. Its is sent to the ATmega via serial communication. This is done using a pySerial, which is a Python API module that allows access for serial port. Furthermore it also has "read" and "write" capabilities. The Arduino receives the numerical values as a String via serial communication. This value is then compared against a set of predefined class values. Whenever any pattern is recognized and matched with its corresponding predefined class values, the rotors of the servo moves 90 degrees which are dots [44] for the pattern of the braille and the other servos stay in 0 degrees position for the cross. Hence producing a braille pattern representing the corresponding character. Figure 25 shows the braille patterns for a few bangla characteristics while figure shows the schematic circuit diagram of the Braille and figure 26 shows the schematic diagram of the braille system implemented

Print	ক	খ	গ	ঘ	ঙ	চ	ছ	জ	ঝ	ঞ
Bangladesh	● ○ ○ ○ ● ○	● ● ○ ○ ● ●	● ● ● ● ○ ○	● ○ ● ○ ○ ●	○ ● ○ ○ ● ●	● ● ○ ○ ○ ○	● ○ ○ ○ ○ ●	○ ● ● ● ○ ○	● ○ ○ ● ● ●	○ ○ ● ● ○ ○
India		○ ● ○ ○ ○ ●							○ ○ ○ ● ● ●	

Print	ঁ	ং	ঃ	ঁ	ঽ	।
Bangladesh & India	○ ● ○ ○ ○ ○	○ ○ ○ ● ○ ●	○ ○ ○ ○ ○ ●	○ ○ ○ ○ ● ○	○ ○ ● ○ ○ ○	○ ○ ● ● ○ ●

Figure 25 Sample of a few general Bangla Character Braille patterns

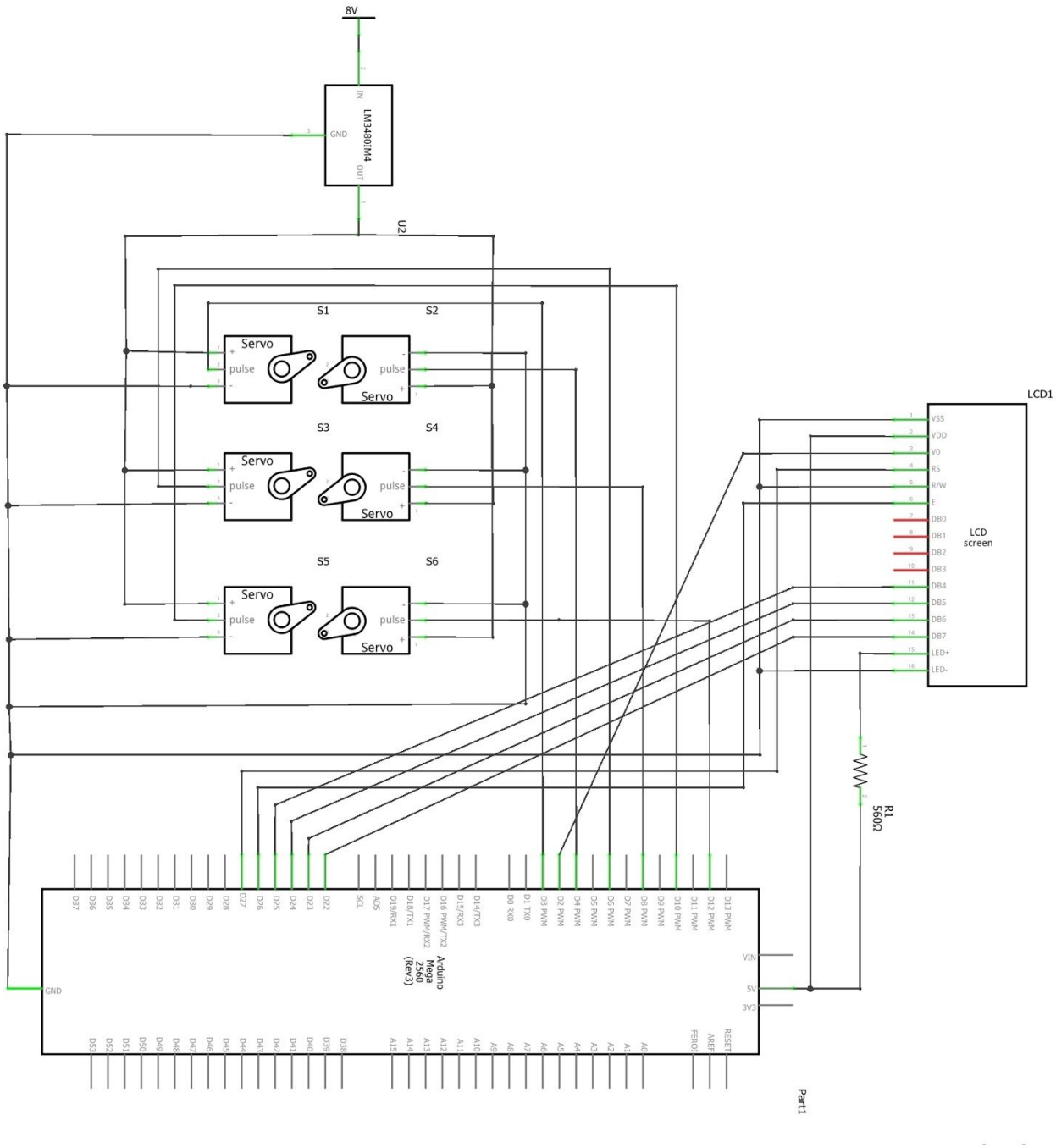
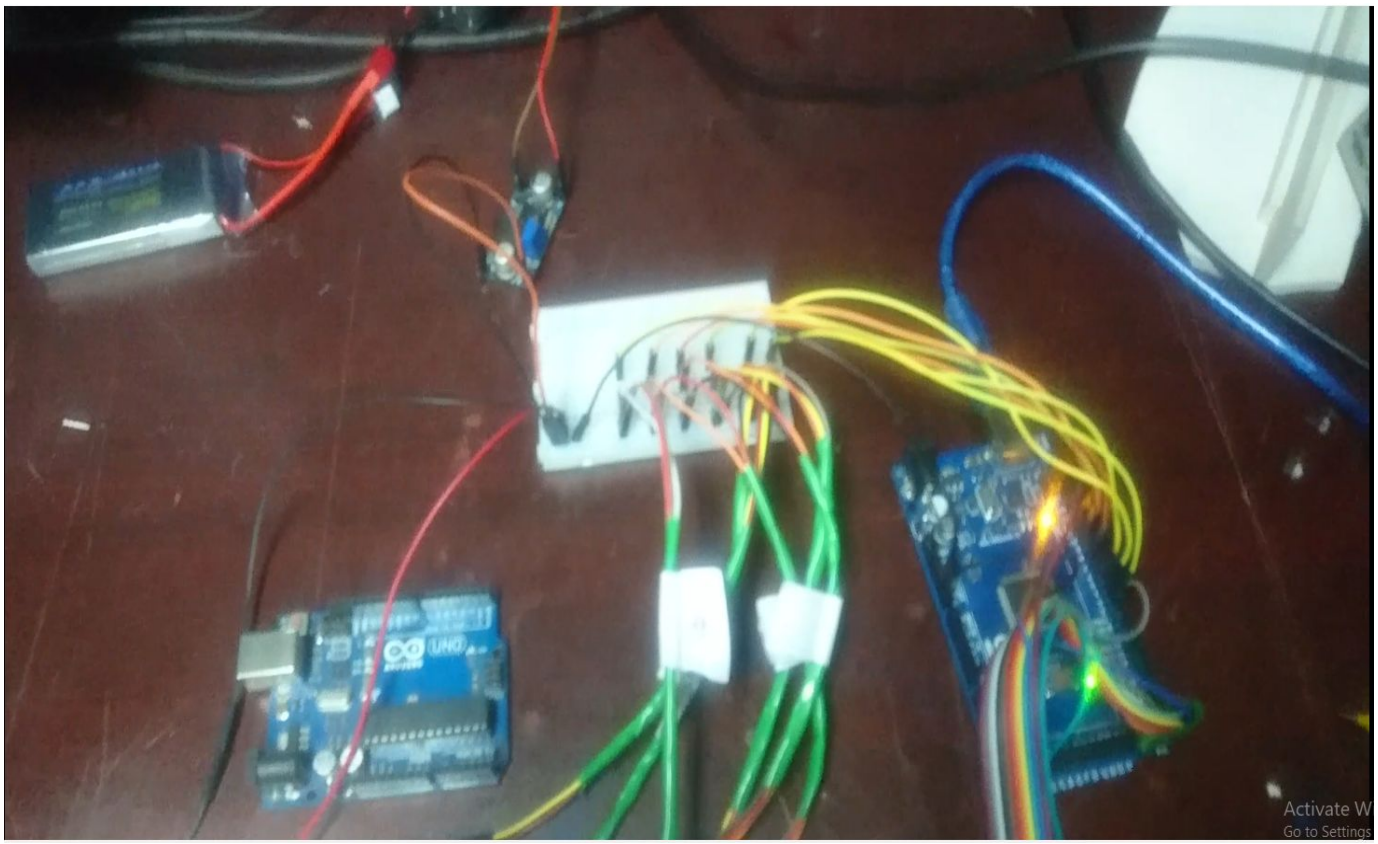


Figure 26 Schematic Diagram of Braille

Front View

Back View



Chapter 8

Result and Analysis

For our thesis research we have chosen and decided to test three different architecture models and have implemented the architecture that has the higher accuracy or the lowest loss function. The three models we used was VGG 16, ResNet50 and Densenet 121. We have trained all three models for 40 epochs and the input image to the model have undergone same preprocessing. As a result we have observed that DenseNet gives the best accuracy among the three models

For the training we have initial used 32x32 images for the dataset in ResNet and got a Validation accuracy of 88.13% .

Then we increased the image size to 40x40 which resulted in an increased in the overall Validation accuracy of 90.62% However this caused the computation time per epoch to increase. Thus becoming slightly time consuming.

Afterwards we tried images of size 75x75 bit it also took a lot of time to complete and the overall increase in accuracy which was negligible .

In the end we chose 50x50 image size that gave optimal desired increase in accuracy with an acceptable computational time of 1078 seconds per epoch.

The table below shows compares the results obtained from the testing of the three models after 40 epochs training with an image size of 50x50.

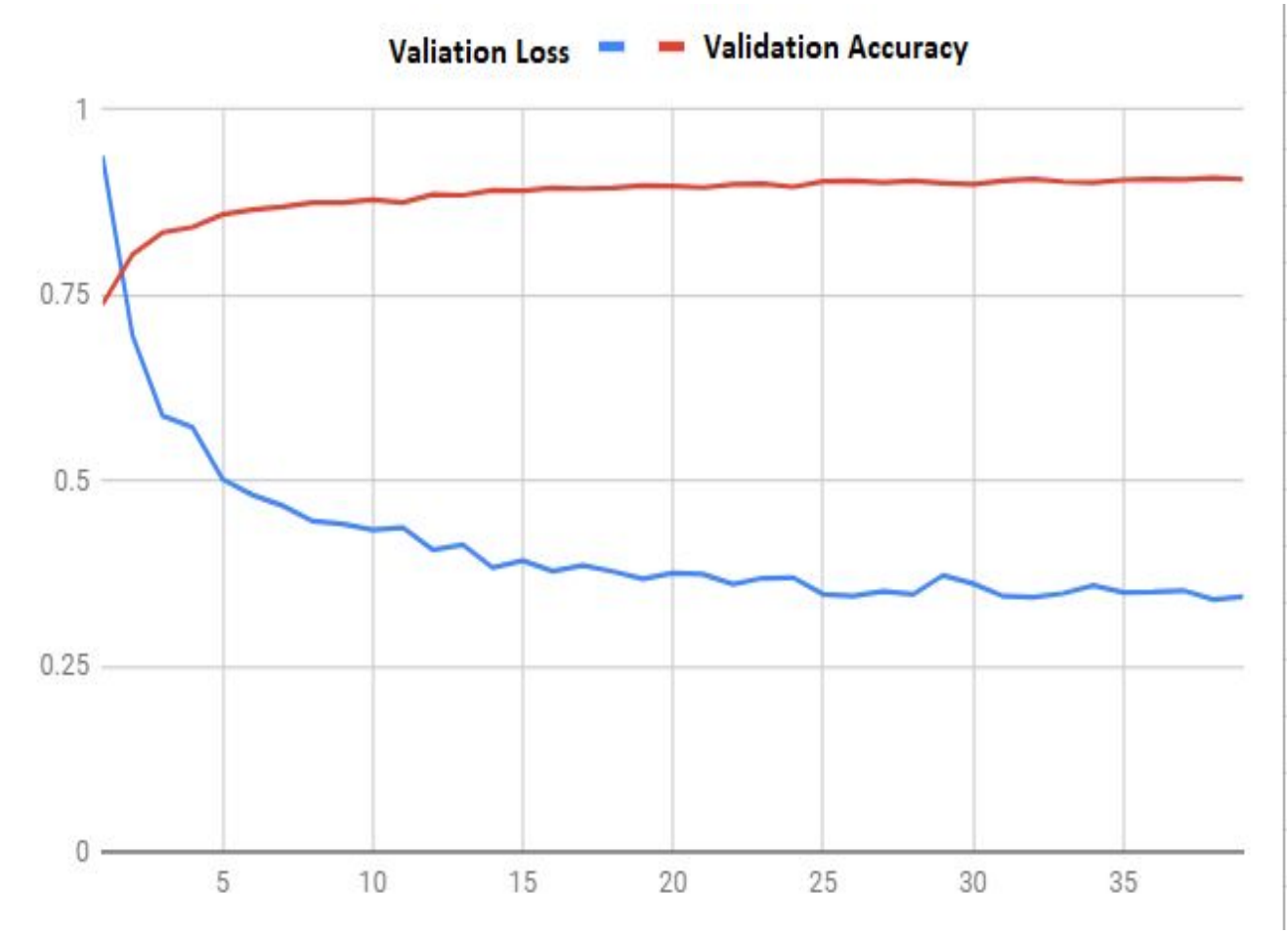
Parameters	VGG-16	ResNet 50	DenseNet 121
Precision	0.91	0.92	0.93
Recall	0.90	0.92	0.93
f1-score	0.91	0.92	0.93
Support	14950	14950	14950
Accuracy	0.9462	0.9358	0.9408

Validation Accuracy	0.9070	0.9212	0.9284
Test Accuracy	0.9154	0.9189	0.9364
Loss	0.2842	0.2298	0.1924
Validation Loss	0.3266	0.3156	0.3038
Test Loss	0.3137	0.3078	0.3024

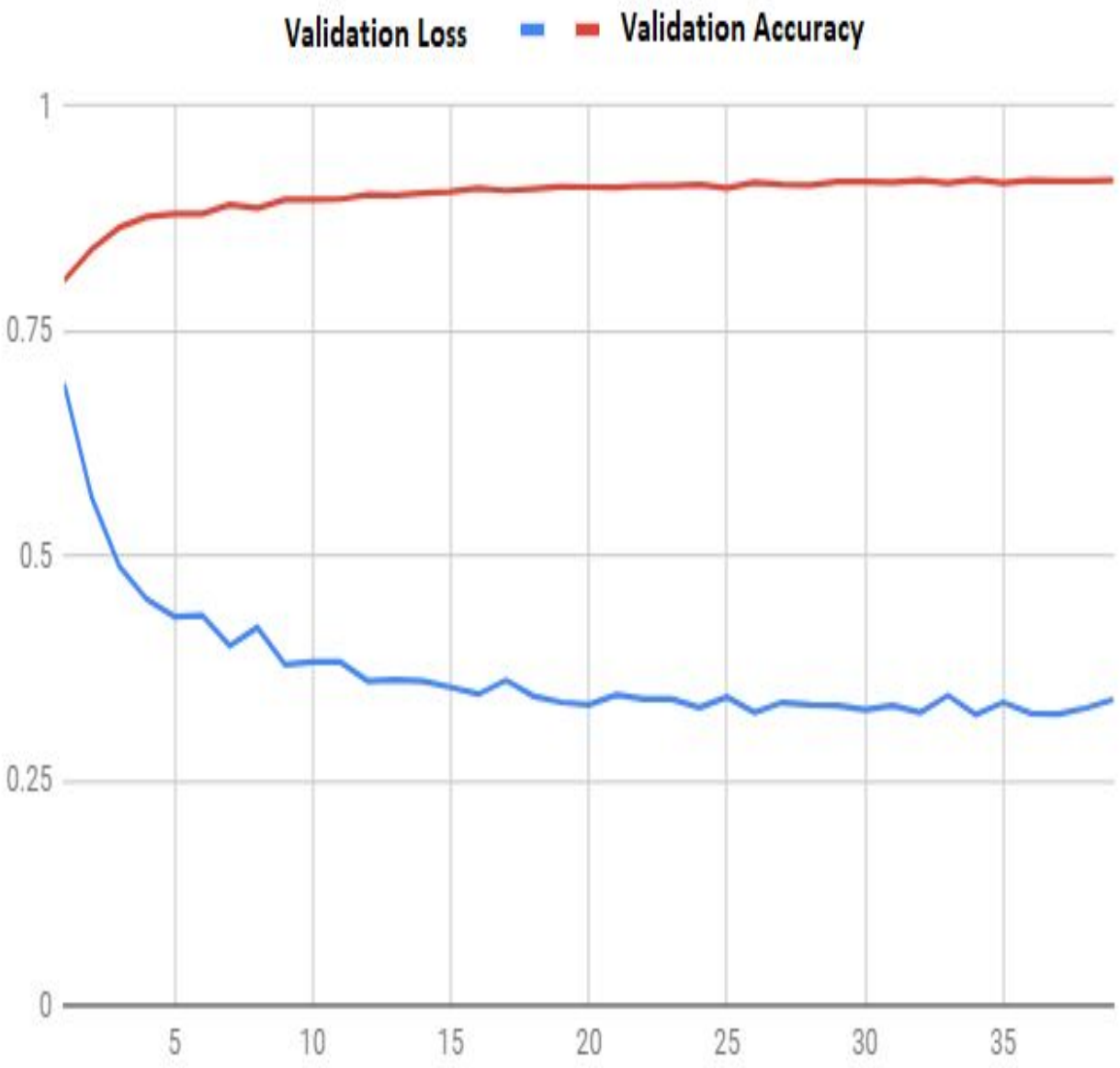
Table. Test Results of the three models

The factors brought into consider where the accuracy, Validation and Test Accuracy, the Loss, Validation and Test Loss.

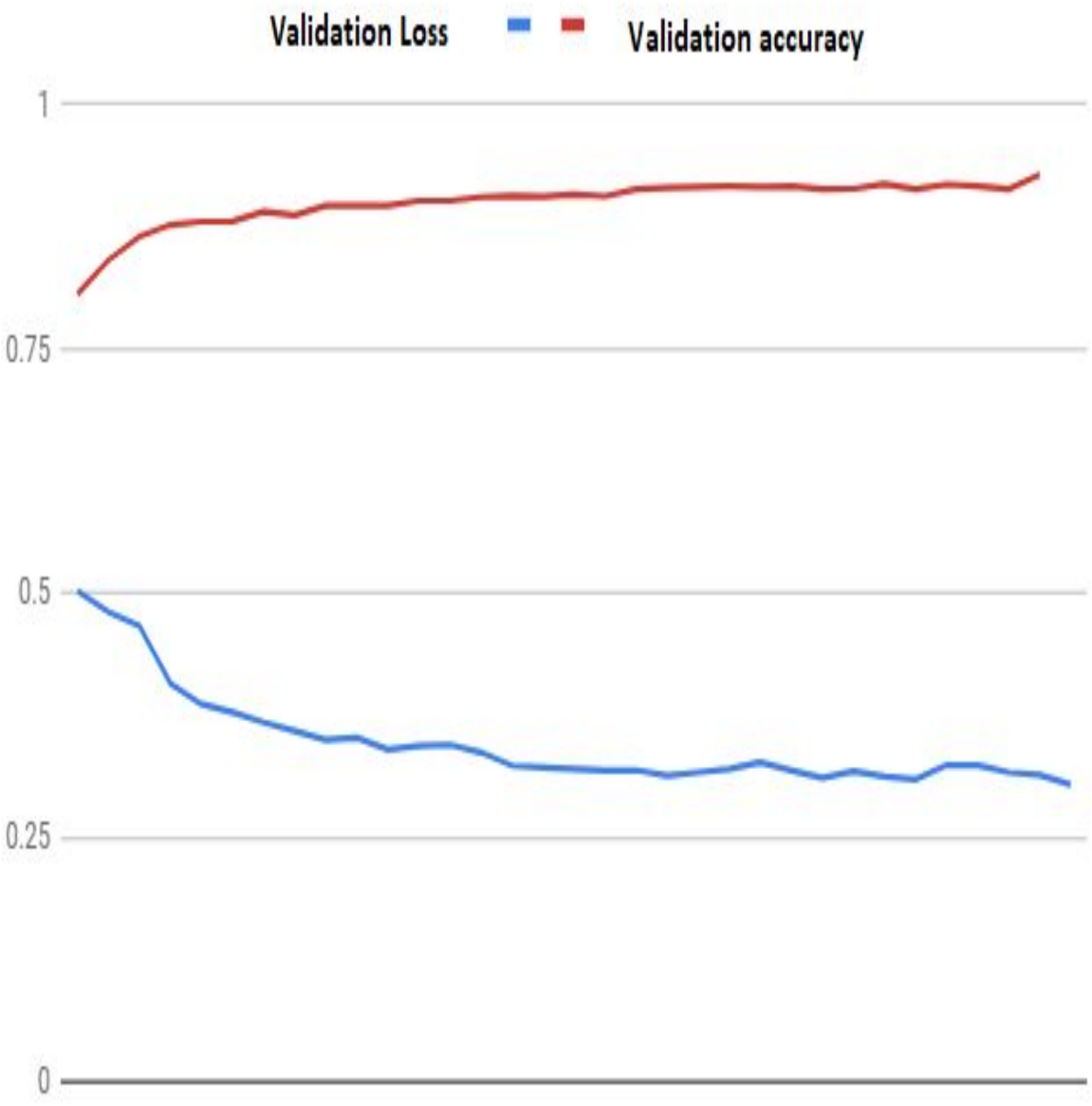
Results for VGG_16



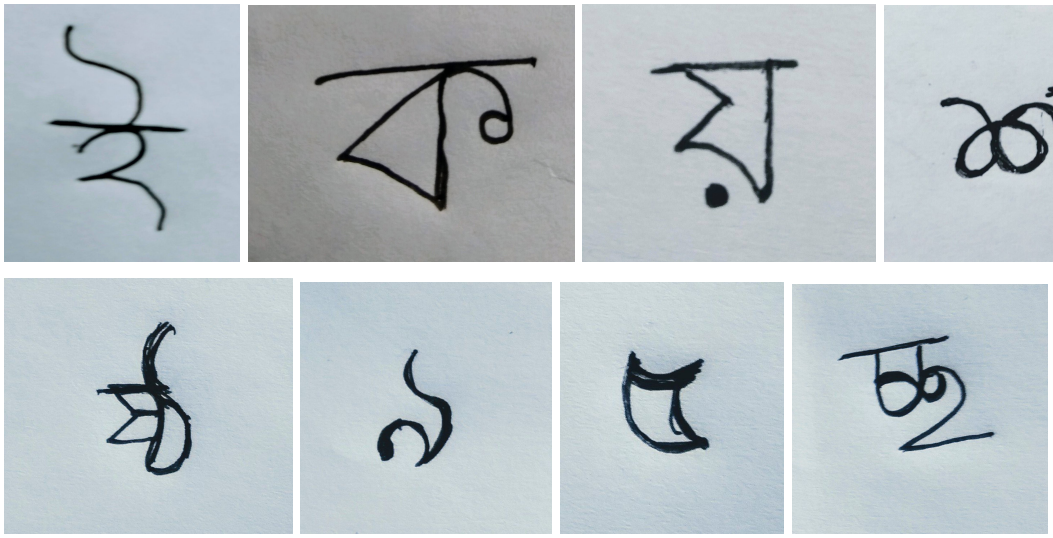
Results for ResNet



Results for DenseNet



Captured Images

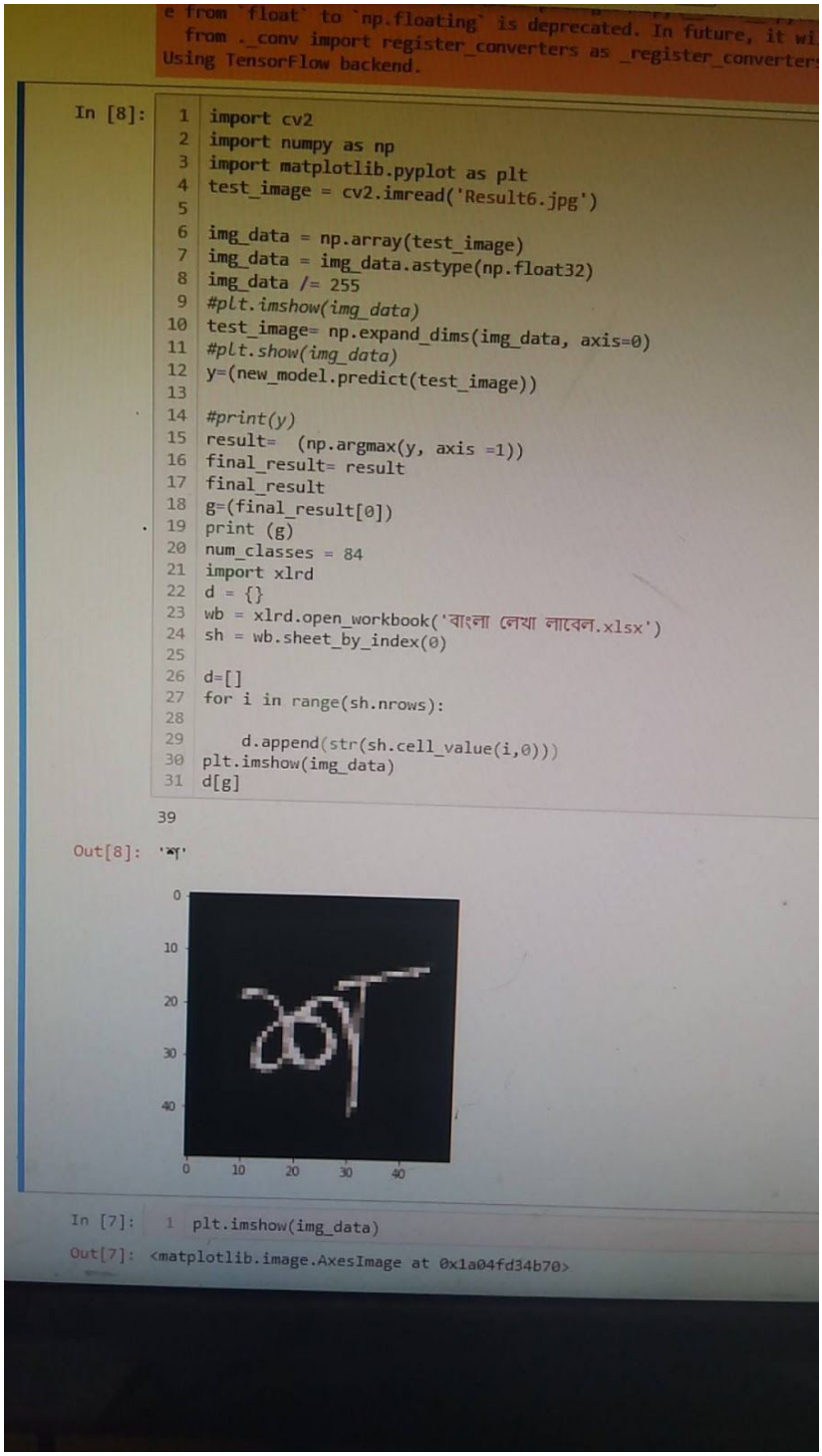


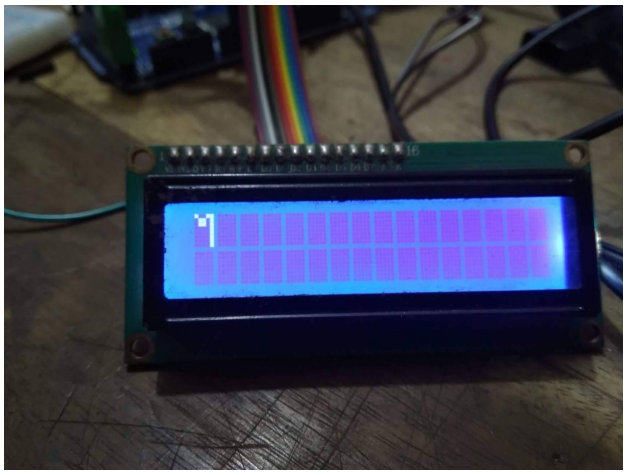
Preprocessed Image

- Resized to 50x50 and padded with black pixels where needed
- Used BiCubic Interpolation
- Turned the image from Color to grayscale
- Binarized and inverted the colors

- Turned the image to numpy array
- normalized by dividing by 255
- And expanded dimension as the mode require a 4D array as input.

Testing

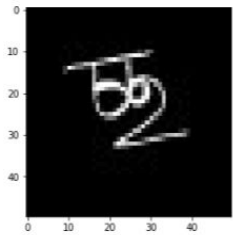

e from `float` to `np.floating` is deprecated. In future, it wi
from ._conv import register_converters as _register_converter
Using TensorFlow backend.
In [8]:
import cv2
import numpy as np
import matplotlib.pyplot as plt
test_image = cv2.imread('Result6.jpg')
img_data = np.array(test_image)
img_data = img_data.astype(np.float32)
img_data /= 255
#plt.imshow(img_data)
test_image= np.expand_dims(img_data, axis=0)
#plt.show(img_data)
y=(new_model.predict(test_image))
#print(y)
result= (np.argmax(y, axis =1))
final_result= result
final_result
g=(final_result[0])
print (g)
num_classes = 84
import xlrd
d = {}
wb = xlrd.open_workbook('বাংলা লেখা লাবেল.xlsx')
sh = wb.sheet_by_index(0)
d=[]
for i in range(sh.nrows):
d.append(str(sh.cell_value(i,0)))
plt.imshow(img_data)
d[g]
39
Out[8]: 'অ'
0
10
20
30
40
0
10
20
30
40
In [7]:
plt.imshow(img_data)
Out[7]: <matplotlib.image.AxesImage at 0x1a04fd34b70>



Other Predicted Outputs

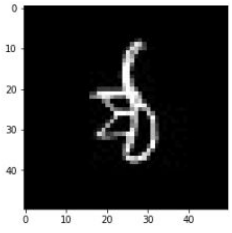
66

Out[15]: '৬৯'



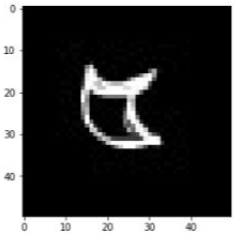
77

Out[19]: '৺'



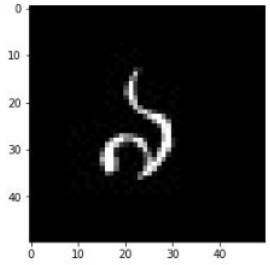
55

Out[17]: '5.0'



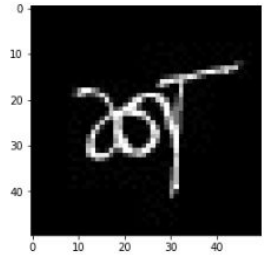
59

Out[9]: '9.0'



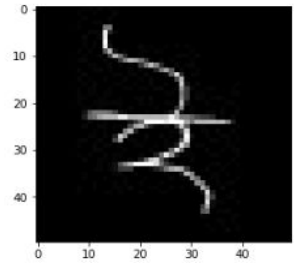
39

Out[20]: '৯৭'



2

Out[22]: '২'



Chapter 9

Future Scope and Conclusion

9.1 Future Scope

This prototype system successfully identifies and presents a braille pattern of any single simple or compound Bangla Handwritten Character. As a result, the system has a significant application in the primary and secondary level education which still heavily depends on Handwritten resources,lecture materials etc. Furthermore not only academic it also has a wide application in any field which will allow any visually impaired person to read any type of documents. In addition, it can be improved to detect a full word or even a sentence that way a entire handwritten document can be printed and represented in a braille pattern. As machine learning is a popular field hence newer models are being researched on everyday trying to improve the accuracy of recognition even further thus the system can also be used as means of testing different Convolutional Neural Network Models.

9.2 Conclusion

Many individual research has has been done on braille and similarly many have researched on machine learning and image processing which are both currently very popular topic but there have been very few in context of Bangla Handwritten Character Recognition. Hence this prototype system makes use of both present technology and the individual concepts and incorporates them into one.

References

[1] M A Karim, M Kaykobad, M Murshed, “Technical Challenges and Design Issues in Bangla Language Processing”, IGI Global, July, USA, 2013

[2] S A Hossain, L A Biswas, M I Hossain, “Analysis of Bangla-2-Braille Machine Translator”, International Conference on Computer and Information Technology, December, Bangladesh, 2014

[3] M Z Alom, P Sidike, T M. Taha, V K. Asari, “Handwritten Bangla Digit Recognition Using Deep Learning”,

[4] U.Pal, B.B Chauduri, “Automatic Recognition of Unconstrained Off-Line Bangla Handwritten Numerals” Third International Conference on Advances in Multimodal Interfaces, Bejing, China, 2000

[5] U. Bhattacharya, M. Shridhar, S. K. Parui, P. K. Sen and B. B. Chaudhuri, “Offline recognition of handwritten Bangla characters: an efficient two-stage approach, Pattern Analysis and Applications”, vol. 15, no. 4 , pp. 445-458, 2012

[6] N. Das, B. Das, R. Sarkar, S.Basu, M. Kundu, M. Nasipuri, “Handwritten Bangla Basic and Compound character recognition using MLP and SVM ”, Journal of Computing, vol 2, Feb, 2010

[7] G. Xie, W. Lu, “Image Edge Detection Based on OpenCV”, International Journal of Electronics and Electrical Engineering, vlo 1, No 2, June, 2013

[8] R. R. Palekar, S. U. Parab, D. P. Parikh, “Real Time License Plate Detection Using OpenCV and Tesseract”, International Conference on Communication and Signal Processing, April, 2017

[9] A. K Garg, “Braille-8 -The unified braille Unicode system: Presenting an ideal unified system around 8-dot Braille Unicode for the braille users world-over, IEEE International Conference on Advanced Networks and Telecommunications Systems, November, 2015

[10] R. D. Sutariya, H. S. Singh, S. R. Babariya, S. A. Kadiyar, D. H. Modi, “Refreshable Braille Display for the Visually Impaired” 14th IEEE India Council International Conference(INDIC), Roorkee, India December, 2017.

[11] A.Koschan, “A Comparative Study on Color Edge Detection”, IEEE Proc. ACCV’95, Singapore,1995, pp 478-545.

[12] V. K. Mishra, S. Kumar, N. Shukla, Image Acquisition and Techniques to Perform Image Acquisition, SAMRIDDHI-A Journal of Physical Sciences, Engineering and Technology, vol 9, issue 9, July, 2017

[13] F. Wang, G. T. Jiao, and Y. Du, “Method of fabric defectdetectin based on mathematical morphology”, Journal of test and measurement technology, Vol. 21, pp. 515-518, 2007.

[14] https://docs.opencv.org/3.4/d9/d61/tutorial_py_morphological_ops.html

[15] M. Biswas, R. Islam, G. K. Shom, M. Shopon, N. Mohammed, S. Momen, M. A. Abedin. BanglaLekha-Isolated: A Comprehensive Bangla Handwritten Character Dataset

[16] Sebastien C. Wong ; Adam Gatt ; Victor Stamatescu ; Mark D. McDonnell.
Understanding Data Augmentation for Classification: When to Warp?

[17]Understanding Neural Network: A beginner’s guide Posted by Ashish Sukhadeve on August 6

[18] Ciresan, Dan; Meier, U.; Schmidhuber, J. (June 2012). "Multi-column deep neural networks for image classification". *2012 IEEE Conference on Computer Vision and Pattern Recognition*: 3642–3649. arXiv:1202.2745. doi:10.1109/cvpr.2012.6248110. ISBN 978-1-4673-1228-8

[19] Krizhevsky, Alex; Sutskever, Ilya; Hinton, Geoffry (2012). "ImageNet Classification with Deep Convolutional Neural Networks" (PDF). *NIPS 2012: Neural Information Processing Systems, Lake Tahoe, Nevada*

*[20]*Google's AlphaGo AI wins three-match series against the world's best Go player". *TechCrunch*. 25 May 2017.

[21] T. Guo, J. Dong, H. Li and Y. Gao, "Simple convolutional neural network on image classification,"

[22] LecunY, Bottou L, Bengio Y, et al. Gradient-based learning applied to document recognition[J]. Proceedings of the IEEE, 1998, 86(11):2278-2324.
[23] Cun Y L, Boser B, Denker J S, et al. Handwritten digit recognition with a back-propagation network[C]//
0

[24] Hecht-Nielsen R. Theory of the backpropagation neural network[M]// Neural networks for perception (Vol. 2). Harcourt Brace & Co. 1992:593-605 vol.1.

[25] Krizhevsky A, Sutskever I, Hinton G E. ImageNet Classification with Deep Convolutional Neural Networks[J]. Advances in Neural Information Processing Systems, 2012, 25(2):2012.

[26] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in ECCV, 2014.

[27] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in ICLR, 2015.
[28] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with Convolutionals," Co RR, vol. abs/1409.4842, 2014.

[29] A. Waibel, et al. Phoneme recognition using time-delay neural networks. Acoustics, Speech and Signal Processing, IEEE Transactions on, 1989, 37 (3): 328-339.

[30] Y. Tang, "Deep learning using linear support vector machines," arXiv prprint arXiv:1306.0239, 2013.

[31]T. Tieleman and G. Hinton, "Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude," COURSERA: Neural Networks for Machine Learning, vol. 4, no. 2, 2012. [Online].

[32]D. Kingma and J. Ba, "Adam: A method for stochastic optimization," arXiv preprint arXiv:1412.6980, Dec. 2014. [Online]. Available: https: //arxiv.org/abs/1412.6980

[33]Olga Russakovsky*, Jia Deng*, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg and Li Fei-Fei. (* = equal contribution) ImageNet Large Scale Visual Recognition Challenge. IJCV, 2015.

[34] Thoma, Martin. "Analysis and Optimization of Convolutional Neural Network Architectures." *CoRR* abs/1707.09725 (2017): n. Pag.

[35] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," arXiv preprint arXiv:1409.1556, Sep. 2014.

[36] C. Szegedy, W. Liu et al., "Going deeper with convolutions," in Conference on Computer Vision and Pattern Recognition (CVPR).

[37] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun; Deep Residual Learning for Image Recognition. The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770-778

[38] C. Szegedy, S. Ioffe, and V. Vanhoucke, “Inception-v4, inception-resnet and the impact of residual connections on learning,”

[39] Huang, Gao, et al. "Densely connected convolutional networks." arXiv preprint arXiv:1608.06993 (2016).

[40] https://www.robotshop.com/media/files/pdf/arduinomega2560datasheet.pdf

[41] https://en.wikipedia.org/wiki/Servomotor#cite_note-1

[42] https://www.arduino.cc/en/Reference/Servo

[43] http://wiki.seeedstudio.com/Raspberry_Pi_3_Model_B/

[44] https://thepihut.com/products/raspberry-pi-3-model-b

[45] https://en.wikipedia.org/wiki/Bengali_Braille