Birzeit University
Department of Computer Science

Arabic Handwritten Digit Classification Based On Convolutional Neural Network Architecture

Ayah Sa'Fin
Saja Saleem
Omar Barghouthi

Supervised By

Dr Radi Jarrar

As a Partial Fulfillment Of The Requirements
For The Degree Of Bachelor Of Computer Science In
Birzeit University
8th June 2020

Abstract

With the improvement of technologies on day to day basis, our need for handwritten documents is starting to decrease significantly, but the problem is that we have too much data to be converted manually that is why we started to use OCR to convert these data. Throughout the last decade, researches came out trying to find the best algorithm that will give us the highest accuracy which will allow the user to use it without any concerns. In this research paper, we will be making a comparison between different Deep Learning algorithms by creating different models/applications for each one of them to find out which one will give us the best result in terms of both Arabic characters and Arabic digit by inputting an image to this model and allow it to be converted into a soft copies. The model we applied was implemented on dataset ADBase, MADBase, and combination of both.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Arabic language is one of the most used languages all over the world. It is the mother language of 310 Million people from different countries in the Arab world.[23] Arabic language has special characteristics that is different from any other language. For example, it comprises 28 characters that can be written in different ways depending on their position in words.This results in 106 different methods to write the character of the Arabic language. An important characteristic is that some characters have the same shape but with different dot location. In addition, Arabic characters are used in other languages like used in Urdu, Persian, old Turkish, and languages which makes it not restricted to the Arabic language only. Accordingly, the recognition of Arabic characters is really a challenging problem, especially the handwritten characters [2].

Arabic Handwritten Optical Character Recognition System, is the process of transforming handwritten characters into digital form. The purpose of this transformation is to increase human productivity[16].

The challenging part of recognising Arabic handwritten characters is mainly because of the different ways of writing, some people tends to combine dots and replace them with dash [56] as in the "Taa" alphabet example as shown in figure 1.3 below.

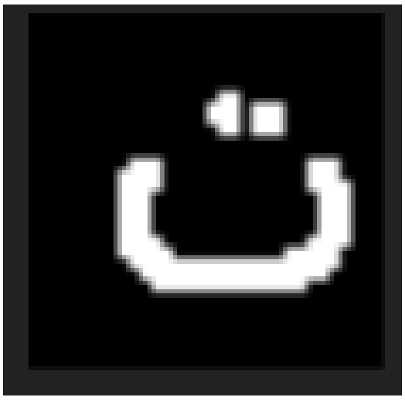


Figure 1.1: TAA' Alphabet testing example 1[1]

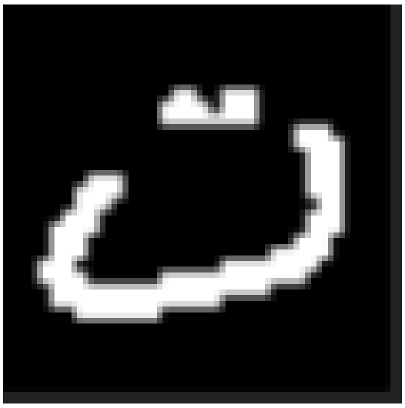


Figure 1.2: TAA' Alphabet testing example 2[1]

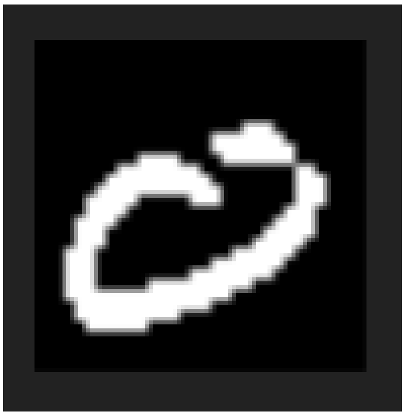


Figure 1.3: TAA' Alphabet testing example 3[1]

Creating an application that converts Arabic handwritten characters and digits is of a great help. It will make data digitization, archiving, and storing much easier and less expensive than storing images. It also has many applications such as automatically reading cheque numbers in banks,[28] or official card readers in companies. Additionally, it can be used in archiving system for old documents[55].

Images processing and object recognition projects vary a lot in means of the number of approaches, that could be followed in solving them.These approaches are mainly statistical approaches such as Support Vector Machines[25], neural networks[27], ... Recently, Deep Learning approaches, that are based on deep neural networks, have gained the researches attention since 2012 [27]. It became the standard choice to go for many object recognition tasks including handwritten recognition. Moreover, it has different algorithms such as Recurrent Neural Network (RNN) [20]that is mostly used in Natural Languages processing (NLP)[30] problems, and Convolutional Neural Networks(CNN)[43] which is related image processing and object recognition problems. Our proposed model is based on deep learning networks to recognise Arabic handwritten digits. In particular, we will focus on testing the difference in performance of different deep learning network architectures and they perform at the task of Arabic handwritten digit recognition.

1.1 Optical Characters Recognition(OCR)

Optical Character Recognition (OCR) [16]is an important and widely used technology in offline[1] handwritten character recognition. It works as a system that converts text into computer based documents. Character recognition is partitioned into two categories online and offline systems. Online is text recognition while run time through a specialized screen such as those in tabloids and digitizers. The online recognition process is easy relatively compared to offline texts, that takes the graphical form as images using scanner.

With the development of life, the demand for storing information increased electronically, and that includes all documents or images printed or handwritten from here comes the need for the OCR, it is the process of converting those printed or handwritten texts into a digital format that can be modified and saved.

There are many applications on the OCR that we will discuss some of them.

1.1.1 OCR applications

The OCR is used in automatic number plate recognition by recognition the numbers that exist with pictures of these plates and is considered as a method of monitoring used by police stations, because through this system the images are stored for the plates with the numbers that have been recognised by the program, which is required by legal issues

There is another aspect in using OCR, which is malicious use. It is used to solve Captcha system that creates access to personal information that benefits harmful programmers. The Captcha system used to distinguish human from software by creating a set of images that contain letters and numbers of different shapes and patterns by adding a distracting backgrounds, and here the OCR frameworks are used to remove this noise and identify the numbers, thus harmful programmers can communicate the user's personal information.

[1] Offline and online systems [11]

1.2 Research problems and goals

This section will talk about the problem of OCR. Many systems exists. Recently there is the Deep Learning. It has different architectures. Then at last that "We aim at conducting experimental comparison between different architectures and we aim at testing which one performs the best at the task of recognising Arabic digits.

Arabic Handwritten Optical Character Recognition is a really challenging problem, regarding to the reasons are previously mentioned, wherefore, the contributors of this research proposed to take a portion of this project, which is specified in recognising Arabic handwritten digits.

The main contribution of this research is:

1. review past researches are done on Arabic Handwritten Digits Recognition (AHDR) problem.

2. propose CNN Deep Learning model that solves AHDR problem.

3. study the effect of CNN architectures on the AHDR problem.

4. use offered utilities of TensorFlow in implementing the proposed model.

Rest of this project is organised as follows: Chapter 2 reviews related work, Chapters 3 and 4 discuss methodology and experiments, Chapter 5 details system analysis, and Chapter 6 discusses the results and conclusions of the experiments.

Chapter 2

Literature Review

2.1 Machine Learning

2.1.1 Machine Learning Definition

Machine learning aims at building programs that make computers able to make decisions that have the potential to improve with experience. The Authoritative Definitions Where Tom Mitchell mentioned it in his book Machine Learning Indeed, machine learning can be defined as "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E"[41].

2.1.2 Types of Machine Learning

Machine learning algorithms can be divided into three main categories: supervised, unsupervised, and reinforcement learning. The following sections describe each category[8].

Supervised Learning Supervised learning algorithms are often described as task oriented, which is fairly common in classification problems. Supervised learning algorithms contain a set of dependent variables to be predicted from the given set of independent variables. Those variables are presented in a dataset. This dataset is usually split into two subsets: training and test sets [8].

The notion of supervised learning comes from the fact that each instance in the dataset is annotated with a class label. This type of learning in used in Artificial Neural Networks (ANN), Decision Tree, K- Nearest Neighbor (KNN), Support Vector Machines (SVM), and other algorithms [41].

Unsupervised Learning Unsupervised learning can be treated as data driven algorithm that is mostly used in clustering population in different groups, where the target or outcome is not given, in other words. This Type of learning is used to find hidden pattern of the data by exploring data analysis, the most common clustering algorithms are: [9]

- "Hierarchical clustering: builds a multilevel hierarchy of clusters by creating a cluster tree.
- "k-Means clustering: partitions data into k distinct clusters based on distance to the centroid of a cluster".

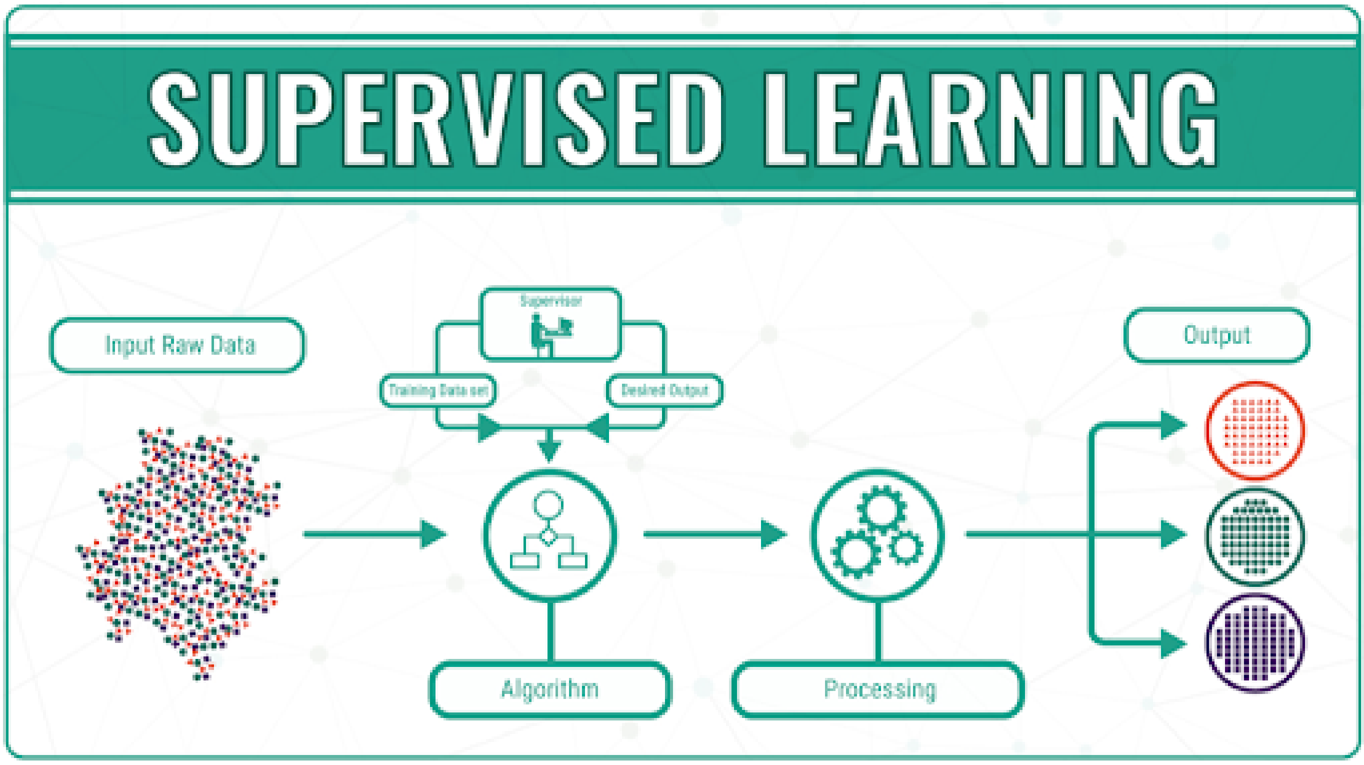


Figure 2.1: Supervised Learning[7]

- “Gaussian mixture models: models clusters as a mixture of multivariate normal density components”.
- “Self-organizing maps: uses neural networks that learn the topology and distribution of the data”.
- “Hidden Markov models: uses observed data to recover the sequence of states”.

“k-Means clustering partitions data into k distinct clusters based on distance to the Learning algorithm[42]

Reinforcement Learning

Reinforcement learning or semi-supervised learning is mainly based on trial and error algorithm where the machine shall train itself continuously from the past experience and capture the best possible knowledge to get the decision, by allowing the agent to take action after interacting with environment to maximize the rewards’ total.The usual model for RL is Markov Decision Process(MDP) [31].

As shown in figure 2.3 that Machine learning has many ways to solve the problems, but the most popular approaches in machine learning are Classification and Regression.

Regression problem Deal with real number (integer, floating point) as output, where its map an input X to output Y by function f(x) [4].

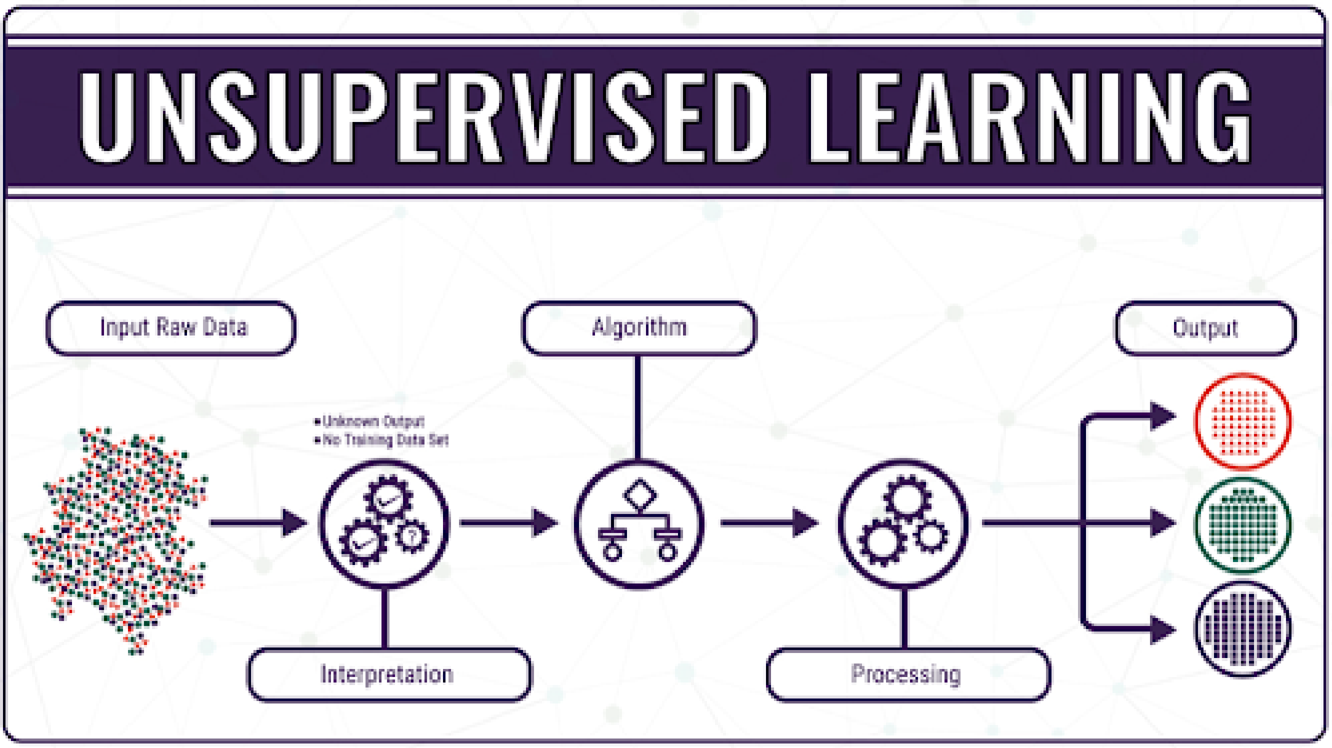
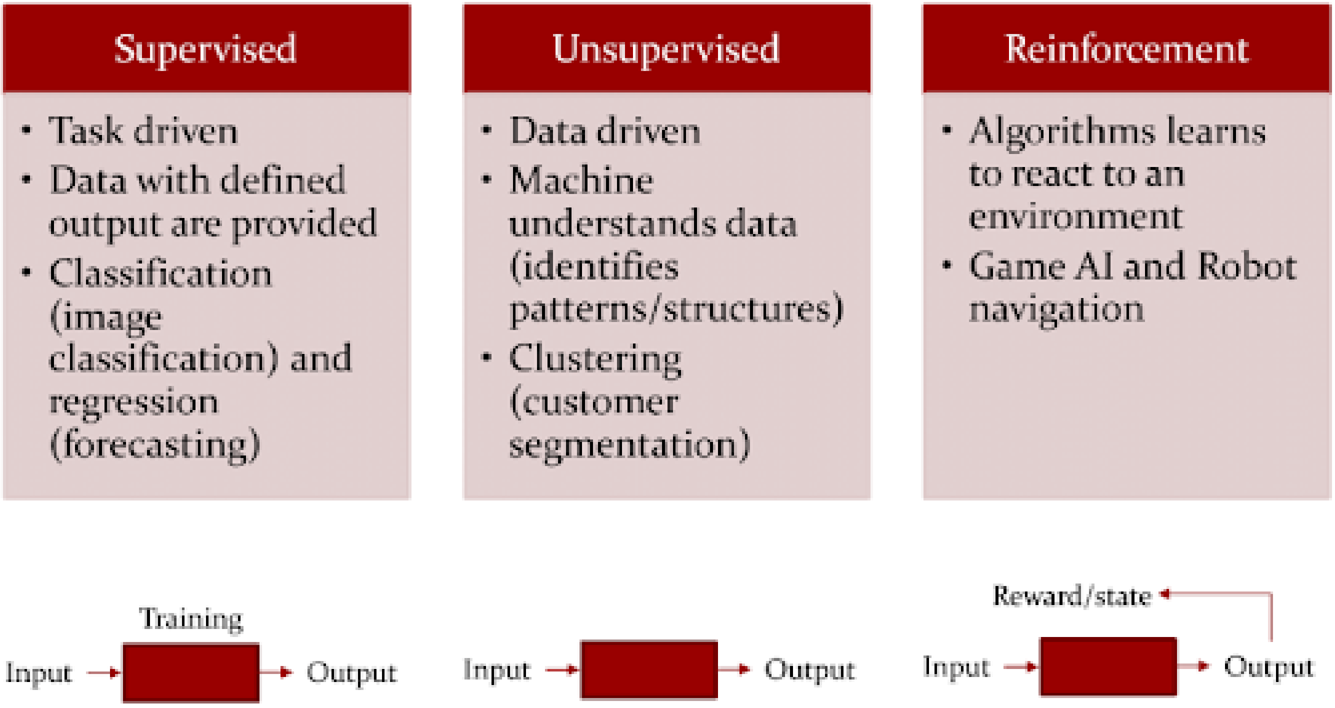


Figure 2.2: Unsupervised Learning[10]



Supervised

- Task driven
- Data with defined output are provided
- Classification (image classification) and regression (forecasting)

Unsupervised

- Data driven
- Machine understands data (identifies patterns/structures)
- Clustering (customer segmentation)

Reinforcement

- Algorithms learns to react to an environment
- Game AI and Robot navigation

Figure 2.3: Machine learning types[6]

classification problem In this research paper will focus on classification problems.

Some of classification problems are non-parametric classifiers, and it's measures are distance based one such as K-nearest neighbor[5]. [1]

classification problem deal with discrete value as output, where its map an input X to output Y by function f(x) [4].

[1] The K-nearest neighbor (K-NN) algorithm instance-based learning or lazy learning ,its supervised learning technique for classification or regression problems,such K: odd number

2.1.3 Pre-Processing

To fix the problems, one might face while working on OCR. Usually, most of these problems are dealt with using the following procedures: pre-processing, segmentation, feature extraction, recognition and post recognition. There are two different ways to approach OCR. The first one is what is called holistic based were to try to recognize into sub-words and then we try to recognize these words as they are. In the second approach, Segmentation, where we split the letter of the sentence by letter and then try to process each character individually [40] .

The main job of pre-processing is to improve the image before processing it and start with the segmentation process. Some of the problems that might face us will work on OCR is that the image is either blurry/low resolution or even of the image was taken with a bad angle and last but not least is lighting which can make the image unclear or not even visible[33].

The first step in pre-processing is noise reduction. Secondly, we need to binaries (to deal with the image as if it is a matrix of 0s and 1s) the image to reduce the intensity range of the image pixels (Useful if the image is converted into gray-scale).

Then, words and sub-words segmentation is a very important step because if we don't deal with overlapping characters between the lines which will cause a problem when the word/sub-word is later sent to the segmentation. Sub-words are usually detected by the vertical gaps between the characters (Since we are talking about the Arabic language we need to keep in mind that most letters are attached). To avoid the overlapping issue, an algorithm was invented but it was never named that compares the test using 8 pixels connectivity which means that the algorithm will scan the image with an 8-pixel radius if there are any characters in that radius then they will be counted as a part of the sub-word [40]. This algorithm was tested on 200 images from the IFN/ENIT dataset and it was reported that it got an accuracy of 85%.

The next stage of pre-processing is the Slope/Slant removal: this stage focuses on making the handwritten text image look like a printed image which means that the slope/slant must be removed. The slope of a sub-word/word is the angle that was made between it and the horizontal line. Many methods to solve such a problem each one has its complexity and process time (examples of those methods: horizontal projection) [40].

Two other algorithms can be used to remove the slant. The First one is removing it by using linear regression [40]. Figure 3.1 When we use this formula we can use it for a character of a sub-word at the place of using a whole sub-word.

Then use naive regression algorithm here we rotate the sub-word which is based on the regression line of this pixels. Which means that we rotate it each time until it fits as shown in the following formula.

$$y = slope * x + intercept$$

where slope is $\frac{n\sum xy - \sum x \sum y)}{(n\sum x^2 - \sum x)}$

Image A is how the output should be printed , B is the user input and Last one to the left (C) is after applying the naive regression to rotate the input image. The problem with this algorithm if we have outliers such as upper or lower tails.In order to deal with words that have any outliers we need to remove their ascenders and descender and after that we can calculate the linear regression of the sub-word. There have been derivations from the naive regression algorithm which are: Regression in a vertical box A, Regression in a horizontal box B, and Regression of the highest density region C [40].

The second method to remove the slant/slope is by using the incremental rotations which was applied by Al-Rahaideh “to find the maximal horizontal projection peak of words which defines the slope of the word (HPP) algorithm “. In order to have the ability to remove the

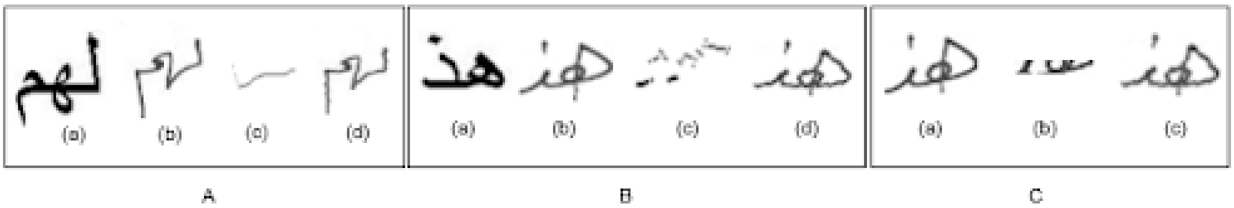


Figure 2.4: A:Vertical Box Regression (a) Printed rub-word, (b) Sloped sub-word, (c) Vertical boxes, (d) result after removing slope . B: Horizontal box Regression: (a) Printed sub-word, (b) Sloped sup-word, (c) Horizontal box. C: Regression of the highest d. [40]

.

slope/slant we can improve on his algorithm by using (HHP) Highest Horizontal Peak and (HVP) Highest Vertical Peak and then sue them with the already existing HPP in every rotation [40] .

One Need to calculate the improved HTP on each one:

1. The current sub-word with zero rotation[40].

2. The one degree rotated sub-word in a clockwise direction[40].

3. The one degree rotated sub-word in the anti-clockwise direction[40].

After calculating the HTP of each one of the three, the highest one should be calculated and known as well.. If (a) has the highest value then we don't need any rotation, and if (b) is the highest then we need to make a clockwise rotation four times each one of them 3 degrees and last if (c) is the highest then perform the same thing as in (b) but this time is going to be counter clockwise [40].

2.2 Similar Projects

There are many methods that were created to solve the Arabic OCR and Arabic digit handwritten recognition.starting from Harifi and Aghagolzadeh in (2007) they use 230 numbers written manually by 23 different people. Also, they used 500 numbers written manually by 50 different people, and the result for their algorithm based on MLP (Multi-Layer Perceptron) network 97.6.[29]. Mahmoud (2008) suggested a technique of recognizing handwritten digit using Support vector machines (SVMs) and Gabor-based features. He used a database with 21120 samples that were written by 44 people, It contains 70% of samples for training and 30% for testing. The result for their algorithm with 3 scales and 5 orientations is 99.85% and by increment the number of scales and orientation by 1 the result becomes 97.94% [39].

Alireza Alaei, Umapada Pal, and Nagabhushan (2009) studied a method to recognize the Persian digit using (SVMs) with a dataset of 80.000 samples written by hand,60.000 samples of training and 20.000 for testing. The result of their work was 98.71%.[13]. Isah A.Lawal, Radwan E.Abdel-Aal, and Sabri A.Mahmoud (2010) use the abductive network method, and they studied the performance of its structure on a dataset for Arabic handwritten digits with 21120 samples written by 44 different people. The result was 99.03%[36].

In 2011 Melhaoui et al suggested a method for Arabic Digits Recognition based on K-nearest neighbor and multi-layer perceptron technique by training the model on a dataset consisting of 200 images for testing and 400 images for training and This method achieved an average of %99 of the small database.[44].P. Pandi Selvi and T. Meyyappan in (2013) used the backpropagation neural network for Arabic digit handwritten recognition and they suggested pre-treatment of images by removing the noise and binarization then training of BPNN, finally the recognition phase, Training performance rate for a small dataset was 96%[49]

In 2014, Maen Takruri, Rami Al-Hmouz, and Ahmed Al-Hmouz expected that the single classification of Arabic digits will not give a reliable classification rate, so they implemented more levels of classification, the classifier relied on Fuzzy C-Means, which was the first level, followed by the second level Support Vector Machine SVM, when more details are needed, but at the third level, they have unique pixel-units that define areas The unique pixels for each of the digit were presented by applying them to a database containing 3510 pictures where 60% of them were for training and the other 40% were for testing. As a result of their work, they got an accuracy of 88%.[52].

In 2017, AKM Ashiquzzaman and Abdul Kawsar Tushar designed a model that identifies handwritten Arabic digit based on CNN, and the model was tried on a dataset called CMATERdb and their work achieved an accuracy rate 97.4%.[14].In the same year 2017, Ali A. Alani create a model that recognizes handwritten Arabic digit based on CNN and RBM. The model was applied to the same set of data that was used with [14] which is CMATERDB. Where RBM extracted the features from the images and then these features are inserted to CNN. This has been shown The model has an accuracy rate of 98.59%, which exceeds the mentioned percentage in[14]

In 2019, AKM Ashiquzzaman, Abdul Kawsar Tushar, Ashiqur Rahman, and Farzana Mohsin made a lot of modifications were made to the existing model with [**?**], so the activation function changed from the linear unit (ReLU) to the Exponential linear unit (ELU), The program was trained to the same set of data, the accuracy rate reached 99.4%[15]

Chapter 3

Proposed Architecture

This section will describe the approaches were followed in our system, some CNN architectures were chosen to be implemented for Arabic handwritten digit classifier

3.1 Convolutional Neural Network Architectures

3.1.1 ResNet

ResNet stands for Residual Network, is the winner of ImageNet challenge in 2015, this architecture is a classical neural network which is used as backbone in computer vision tasks. This algorithm was made to train a deep neural network with 150+ layers successfully.[21]

ResNet Weakness is vanishing gradient problem that occurs when certain activation function added to the neural network which makes loss function goes to zero, in other words, the algorithm can't learn more so the learning will be hard, since the gradient is back-propagated to the earlier layers, and repeating this step may cause the gradient too small, then performance will be saturated. [21]

In degradation problems, shallower layers leads to better performance, so here lies the importance of skip function, which skips extra layers while training. [47]

Kaiming He et. al proposed to use residual blocks by connecting the output layer, with input of an earlier layer instead of mapping the input x to an output Y following equation

$$M(x) = y$$

which called direct mapping. Residual function was built to use the difference between a mapping applied to x and the original input, as the following formula

$$F(x) = M(x) - x$$

, the output of residual block passed to the skip function

$$M(x) = F(x) - x$$

[24] 3.1 shows the skip function[48]

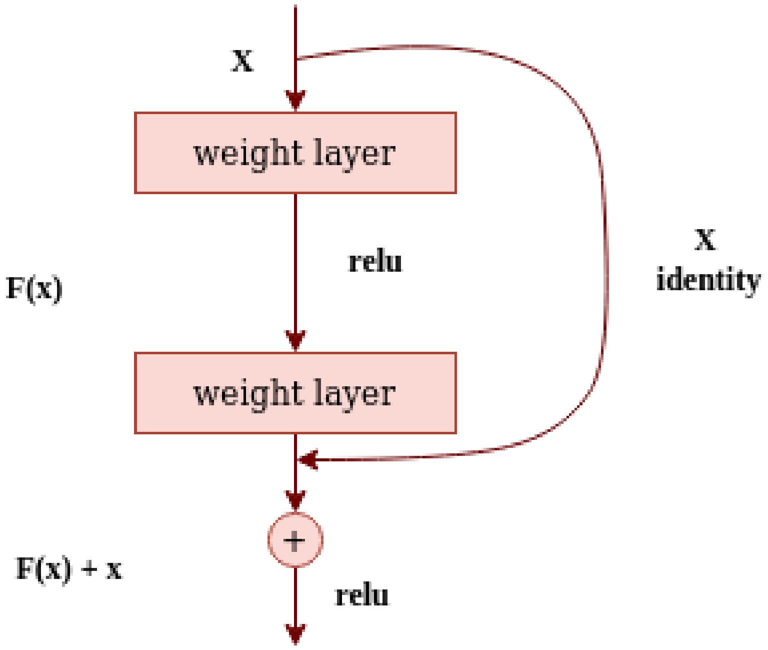


Figure 3.1: Skip Function

3.2 AlexNet

The pictures in natural life contain many details and also with a very large amount of data. So AlexNet was found to use it on high-resolution images like ImageNet scale while reducing training time and improving performance.AlexNet was designed by the Super Vision group consisting of Ilya Sutskever, Geoffrey Hinton, Alex Krizhevsky.[17] In 2012 Alex Net won the Image classification challenge (ILSVRC)[45], excelling by reducing the top 5 error from 26% to 15.3%[17].This algorithm was trained with 1.2 million images, these images have 1000 different categories. [45]

The architecture of AlexNet

1. 5-Convolutional layers
   (a) Conv-1 with 11x11 filter size
   (b) Conv-2 with 5x5 filter size
   (c) Conv-3 with 3x3 filter size
   (d) Conv-4 with 3x3 filter size
   (e) Conv-5 with 3x3 filter size
2. 3-Max pooling layers with 3x3 filter size
3. 3-Dense layers
4. 1-Output dense-softmax layers

AlexNet is made up of eight layers: The First layer, is a 227x227x3 RGB image which is the input for the first convolutional layer. Here the image passes through the first convolutional

Table 3.1: AlexNet Architecture

Layer No.	Layer	Feature Map	size	Kernel Size	Stride	Activation
Input	Image	1	227x227x3	-	-	-
1.1	Convolution	96	55x55x96	11x11	4	relu
1.2	Max Pooling	96	27x27x96	3x3	2	relu
2.1	Convolution	256	27x27x256	5x5	1	relu
2.2	Max Pooling	256	13x13x256	3.3	2	relu
3	Convolution	384	13x13x384	3x3	1	relu
4	Convolution	384	13x13x384	3x3	1	relu
5.1	Convolution	256	13x13x256	3x3	1	relu
5.2	Max Pooling	256	6x6x256	3x3	2	relu
6	FC	-	9216	-	-	relu
7	FC	-	4096	-	-	relu
8	FC	-	4096	-	-	relu
Output	FC	-	1000	-	-	Softmax

layer which is made up of 96 filters, a 11*11 kernel size and a 4*4 stride. This first layer will change the dimensions of the image to 55*555*96. After that the first max-pooling layer is applied which has a 3x3 pool size and a stride which will reshape the image to 27x27x96[53] .

Second layer we will complete with second convolutional layer with 256 filters, a 5*5 kernel size and a 1*1 stride Then a maximum pooling layer with filter size 3 x 3 and a stride of. The output of this layer will be a 13x13x256 image.

For the second layer there will be another convolutional layer followed by a max-pooling layer, the convolutional layer will have 5*5 sized filter[53] .

Third, fourth, and fifth layers are convolutional layers with filter size 3 x 3 and a stride of one. Third and the fourth layers use 384 feature maps where fifth layers use 256 filters. Each one of these layers is followed by a max-pooling layer[53] .

Sixth layer the output of the convolutional layers flattens through a fully connected layer with 9216 feature maps each of size 1x1. Seventh and eighth layers use 2 fully connected layers with 4096 units again. Finally, the softmax output layer ŷ[45][53] [34]

3.3 VGG

VGG-16 is most preferred to extract features from the image where it was ranked second in ILSVRC competition in 2014[54], it's trained on ImageNet dataset contains fourteen million images and a thousand categories[18].VGG-16 was developed by Zisserman and Simonyan[54].VGG-16 took this name because it contained sixteen layers of weight, a convolutional layer, max-pooling layer, activation layer, and fully connected layer[18].

The number of all layers that have weight and which do not have weight is twenty-one layer

1. 13 convolutional layer all with size 3x3 Conv 1-1 to Conv 1-2 has 64 #filters.
2. Conv 2-1 to Conv 2-2 has 128 #filters.
3. Conv 3-1 to Conv 3-3 has 256 #filters.
4. Conv 4-1 to Conv 4-3 has 512 #filters.
5. Conv 5-1 to Conv 5-3 has 512 #filters.
6. 5 max-pooling layer with 2x2 filter size
7. 3 fully connected layer

First layer is 224x224x3 then it's passed to the first convolutional layer (CONV1-1) with 64 filters with size 3x3 the output size of this layer is 224x224x64, and then it passes in the second layer (CONV1-2) with same number of filters and same size and, then it continues by entering or passing through all layers, each with its number of filters and sizes as shown in the figure 9.10 [26]

Table 3.2: VGG Architecture

Layer Name	Output size	#filters	filter size	stride	padding	# parameters
INPUT	224x224x3	-	-	-	-	-
CONV1-1	224x224x64	64	3x3	1	1	(3*3*3)*64=1728
CONV1-2	224x224x64	64	3x3	1	1	(3*3*64)*64=36,864
MAX POOL1	112x112x64	-	2x2	2	-	-
CONV2-1	112x112x128	128	3x3	1	1	(3*3*64)*128=73,728
CONV2-2	112x112x128	128	3x3	1	1	(3*3*128)*128=147,456
CONV2-2	112x112x128	128	3x3	1	1	(3*3*128)*128=147,456
MAX POOL2	56x56x128	-	2x2	2	-	-
CONV3-1	56x56x256	256	3x3	1	1	(3*3*128)*256=294,912
CONV3-2	56x56x256	256	3x3	1	1	(3*3*256)*256=589,824
CONV3-2	56x56x256	256	3x3	1	1	(3*3*256)*256=589,824
CONV3-3	56x56x256	256	3x3	1	1	(3*3*256)*256=589,824
MAX POOL3	28x28x256	-	2x2	2	-	-
CONV4-1	28x28x512	512	3x3	1	1	(3*3*256)*512=117,648
CONV4-2	28x28x512	512	3x3	1	1	(3*3*512)*512=2,359,296
CONV4-3	28x28x512	512	3x3	1	1	(3*3*512)*512=2,359,296
MAX POOL4	14x14x512	-	2x2	2	-	-
CONV5-1	14x14x512	512	3x3	1	1	(3*3*512)*512=2,359,296
CONV5-2	14x14x512	512	3x3	1	1	(3*3*512)*512=2,359,296
CONV5-3	14x14x512	512	3x3	1	1	(3*3*512)*512=2,359,296
MAX POOL5	7x7x512	-	2x2	2		-
FC6	4096	-	-	-	-	7*7*512*4096=102,760,448
FC7	4096	-	-	-	-	4096*4096=16,777,216
FC8	1000	-	-	-	-	4096*1000=4,096,000

3.4 GoogLeNet

This architecture won ImageNet Challenge, by achieving a top-5 error rate of 6.67% which is too close to human level performance. [19]

GoogLeNet is made of 22-layers, and almost 12x less parameters. The idea behind GoogLeNet was to make a model that could be used on limited resources devices such as smart phones[51]

GoogLeNet is characterized by it's inception layer and it's learn able filters[50].

1. The first layer will be the input layer.
2. The second layer is a convolutional layer.
3. Then we will have a maxpooling layer.
4. The fourth and fifth layer will be 2 convolutional layer.
5. The six layer will be another max pooling layer.
6. Followed by 2 inception blocks which will then be followed by a max pooling layer.
7. From the 9th layer until the 13 layer we will have 5 inception blocks which will then be followed by a max pooling layer.
8. Then there will be another 2 inception blocks followed by a global average pooling.

The algorithm works as follows first of all we have the input layer which will receive the image after it was preprocessed and then resized to the correct size, in this case 28*28 pixels. After that, the image will be sent to a convolutional layer, this layer will have 64 filters, and a 7*7 kernel size with a relu activation function, the output of this layer will be sent to max pooling layer which will have a 3*3 pool size with padding set to same, and an activation function of relu.

After the max-pooling there will be 2 convolutional layers following each other. The first one will be close to the first convolutional layer but with a kernel size of 1*1 while the second one will have 192 filters and a kernel size of 3. After these two layers there will be another max pooling layer with the same details as the last one without padding added to it.

Before continuing with the function we need to explain what is an inception block and how does it work. An inception block is group of layer that were made in order to have the ability to reuse it again. An example of the inception block is in 3.2

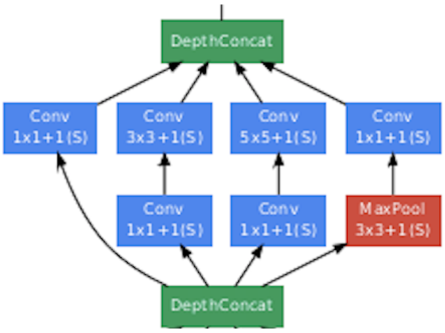


Figure 3.2: Inception Block[32]

After that there will be 2 inception blocks followed by a max-pooling layer similar to the first one. And then there will be another 5 inception blocks followed with a max-pooling layer.

And at last there will be another 2 inception blocks followed by a global average pooling layer and ended with a dropout of 0.4.

Table 3.3: GoogLeNet Architecture

Layer type	Output Shape	Parameter Number
Conv2D	None, 28, 28, 6	156
average pooling2d One	Average None, 14, 14, 6	0
Dropout	None, 14, 14, 6	0
Conv2D	None, 14, 14, 16	2416
average pooling2d	Average None, 7, 7, 16	0
Dropout	None, 7, 7, 16	0
Flatten	None, 784	0
Dense	None, 120	94200
Dense	None, 84	10164
Dense	None, 10	850

Full model table, and diagrams are shown in 9.8 9.5

3.5 LeNet-5

LeNet-5 is a classic Convolutional Neural Network architecture that was mainly proposed by Yann LeCun, Leon Bottou, Yosuha Bengio and Patrick Haffner to help in the printed and handwritten characters back in the 1990s. This algorithm contains 8 layer and they are as the following[22]

1. The first layer is the input layer.
2. The second layer is a Convolutional layer.
3. The third layer will be a subsampling layer.
4. The fourth and the fifth layer are just like the second and the third respectively .
5. The sixth layer will be our last Convolutional layer.
6. The seventh layer will be the fully connected layer
7. And at last we will have the output layer which will display the output.

So let's explain how each layer works. After that we received an input and it got sent to the first convolutional layer.[46]

The LeNet-5 architecture was as follows: For the first convolutional layer the input size was 28*28 with a relu activation function and after passing through the first C1-layer it will become 6 different maps each map will contain a 5*5 kernels. Each neuron of C1 plans has an input received by 5*5 receptive fields. This layer will contain 156 trainable parameters, 4704 number of neurons, and 122304 connections. After that, we will have the first sub-sampling layer S2 which will have 6 feature maps and a 2*2 kernel which will reduce the image dimensions to 14*14 and also reduce the connection. And then we will have a dropout layer which will prevent over-fitting

For the fourth and fifth layers we will have it just like the second and third layers but in the fourth layer and fifth will have the number of maps increased from 6 to 16 while the number of kernels will stay the same as they were before. The sixth layer which is our third and our last convolutional layer, here we will have 120 feature maps with a 6*6 kernel and then we will have our fully connected layer.[46]

Table 3.4: LeNet-5 Architecture

Layer type	Output Shape	Parameter Number
Conv2D	None, 28, 28, 6	156
average pooling2d	Average None, 14, 14, 6	0
dropout1 Dropout	None, 14, 14, 6	0
Conv2D	None, 14, 14, 16	2416
average pooling2d	Average None, 7, 7, 16	0
dropout2 Dropout	None, 7, 7, 16	0
Flatten	None, 784	0
Dense One	None, 120	94200
Dense Two	None, 84	10164
Dense Three	None, 10	850

o

Chapter 4

Ethical Consideration

Ethical consideration raised in each stage of this research from the initial stage to the final design, especially in data collection parts, in which all images will be used in later stage of this research will be for scientific research purposes only.And no tracking of users will be made.

Chapter 5

Experiments

5.1 Datasets

5.1.1 ADBase

The ADBase dataset contains 70000 images that were collected from 700 different students whether they were high school students,college students, or government institutions. Having the dataset collected from students from different age ranges so that the dataset can have different handwritten formats. The forms were scanned at 300dpi resolution. After these forms are scanned, binaries, an image was extracted and categorized.[12]

5.1.2 MADBASE

The MADBase dataset is a modified version of the ADBase dataset where both the size and the format of this dataset are the same as those in the MNIST dataset.

The MADBase dataset was created as follows. First, he images from the ADBase were imported and they recalculated the height and the width of this dataset to match the height and the width of the MNIST. Due to the normalization, the images have gray-levels.[12]

5.1.3 Arabic Handwritten Digit Dataset AHDD1

The AHDD1 [37] dataset is the inverted CSV version of the MADBase.

5.1.4 Input images

5.1.5 Preprocessing

Segmentation

For the segmentation of the inputted image which was imported by taking a photograph of a handwritten Arabic digit(The photo followed a certain format). This process extracted the Digits from the inputted images in order to have the ability to predict them using the machine learning models. After extracting the digits the images were sent to the preprocessing stage in order to improve the quality.

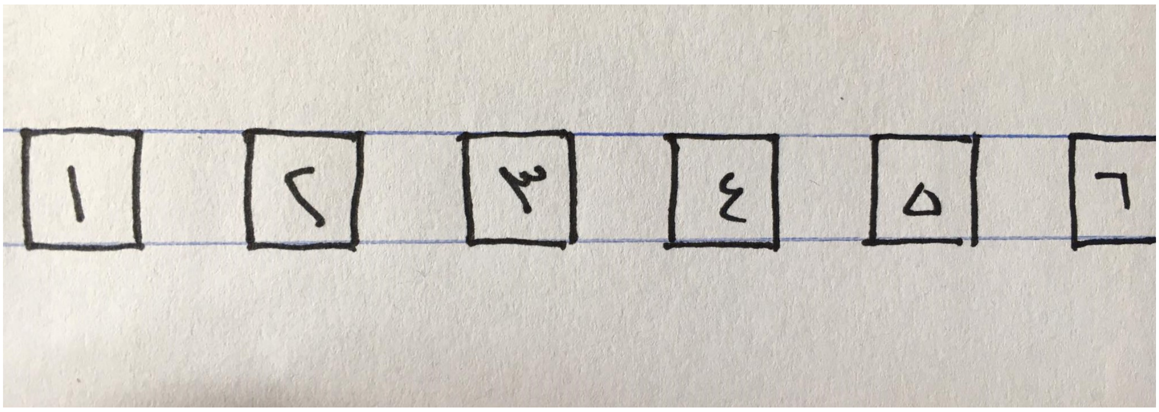


Figure 5.1: Example of the inputted image format

Image Cleaning

To improve the images quality, they were passed to various preprocessing steps. The first one of these steps was to clean the image, we did that by using both the enhanced method and sharpness method that were imported from the PIL library in python. After that, we had to resize the image. Also, we had o flip the color of the image (to make the number in white and the background is black) and then the dataset was transformed into a CSV file to train the model. These steps were also implemented on the inputted dataset which was created so that both of these datasets can be on the same form and can be compared to each other.

5.2 Over-fitting Avoidance

During the experiment phase, researchers mixed datasets by making MADBASE as training set, and ADBASE as testing set, then model's behaviour tended to overfit, as will be shown in results part, following methods are used to detect, and avoid overfitting

1. Cross validation using 9600 samples from 60K sample of the dataset
2. Early stopping with patience of 10
3. Data Shuffle
4. Swap parts from test set in ADBASE dataset to be in training set of the same dataset.

Chapter 6

Results and Discussion

6.1 Results

This section will show results were obtained by the experiments are done on CNN architectures -chosen from ImageNet competition[1] which trained and tested on AHDBASE, MADBase datasets

Results will be shown in parts, as mentioned in expirements part, that four model have been chosen to be implemented, which are: LeNet-5, GoogLenet, AlexNet, and VGG and results were as shown in tables 6.1, 6.2, 6.3 below.

Table 6.1: AlexNet And VGG Results

Model	Train Data	Test Data	Train Loss	Train Accuracy	Test Accuracy	Test Loss
VGG	MADBAS	MADBase	230.26%	9.74%	230.27%	10.00%
AlexNet	MADBAS	MADBase	1.37%	99.57%	4.96%	98.65%

As mentioned in experiment chapter, researchers decided to dive deeper in both models LeNet-5 and GoogLeNet, by implement them on multi datasets

1. GoogLeNet was fit using following settings
   (a) batch_size=100
   (b) epochs=3
   (c) verbose=1
   (d) LeNet-5
2. LeNet-5
   (a) validation_split=0.33
   (b) batch_size=20
   (c) epochs=5
   (d) verbose=1

As shown in table 5.2, some results seems that model suffers from overfitting, when test set changed to AHDBASE and the following results are produced after applying steps are mentioned in overfitting avoidance section

Results will be shown as follows:

[1] http://www.image-net.org/challenges/LSVRC/

Table 6.2: GoogleNet and LeNet-5 Results

Model	Train Data	Test Data	Train Accuracy	Test Accuracy
LeNet-5	MADBase	MADBase	98.9	96.6
LeNet-5	MADBase	ADBase	98.9	53.1
LeNet-5	ADBase	ADBase	99.1	40.9
LeNet-5	ADBase	MADBase	99.15	99.16
Googlenet	MADBase	MADBase	99.40	98.56
Googlenet	MADBase	ADBase	98.9	60.9
Googlenet	ADBase	ADBase	98.8	64.67
Googlenet	ADBase	MADBase	98.90	98.84

1. First table will show results after applying shuffling, cross validation and early stopping
2. Second table will show results after swapping parts from train dataset with others in train dataset

Table 6.3: GoogleNet and LeNet-5 Results on combinations of ADBase and MADBase datasets

Model	Train Accuracy	Validation	Test Accuracy
LeNet-5	98.6%	98.2%	98.0%
GoogLeNet	99.42%	99.12%	99.21%

6.2 Discussion

As shown in results section that overfitting was noticed when test set is not MADBASE even after implementing cross validation and early stopping methods, and this overfitting occurred due to the huge difference of image quality in both training and testing sets, since we take a random images datasets differences will be shown in terms of kurtosis, skewness and mean [35]

Mean

1. ADBASE
   (a) Test 0.447368
   (b) Train 0.789683
2. MADBASE
   (a) Test 239.403
   (b) Train 233.847

Kurtosis

1. ADBASE
   (a) Train -1.95518
   (b) Test 0.021048
2. MADBASE
   (a) Train 7.00891
   (b) Test 10.7089

Kurtosis is image noise indication, so high kurtosis means low noise and low resolution

Skewness

1. MADBASE
   (a) Train -2.9068
   (b) Test -3.4434
2. ADBASE
   (a) Test 0.211702
   (b) Train -1.42164

Image skewness is positive when image is more darker and has glossier surface, so postive skewness as in test image test image makes decision making about image surface harder than negative ones. In other words positive skewness indicates how unbalanced pixel distribution is

Mentioned image measurements proofed that train images in ADBASE dataset have higher resolution, and it's edge detection is better than others in test set, and this is the main reason behind the overfitting.

And about MADBASE, both train set and test set measurements values are close, so this is why the system gave high accuracies for both training and testing set when test set is MADBASE, or combination of both MADBASE and ADBASE

Finally, in all cases, we can claim that using the right CNN architecture with right dataset, could improve the process of Arabic handwritten digit classification

$$\begin{bmatrix}
1942 & 19 & 2 & 2 & 0 & 25 & 2 & 1 & 4 & 3 \\
3 & 1995 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\
2 & 2 & 1991 & 1 & 2 & 0 & 0 & 0 & 2 & 0 \\
0 & 1 & 8 & 1986 & 0 & 0 & 3 & 0 & 1 & 1 \\
2 & 10 & 6 & 1 & 1980 & 0 & 0 & 0 & 0 & 1 \\
9 & 0 & 11 & 0 & 1 & 1972 & 0 & 4 & 0 & 3 \\
0 & 8 & 0 & 0 & 1 & 0 & 1987 & 0 & 0 & 4 \\
0 & 1 & 0 & 0 & 0 & 1 & 0 & 1998 & 0 & 0 \\
0 & 2 & 2 & 0 & 0 & 1 & 1 & 0 & 1994 & 0 \\
1 & 1 & 1 & 0 & 2 & 1 & 5 & 0 & 2 & 1987
\end{bmatrix}$$

Figure 6.1: LeNet-5 Model Confusion Matrix Trained on ADBase and Tested on MADBase

6.2.1 Comparison to other systems

Our proposed system used aimed to improve previous models are made to classify Arabic Handwritten digits by using different CNN architectures and merging datasets.

The proposed architectures we followed achieved better accurices than ones are achieved in other systems with same or different datasets as shown in figure 6.5

LeNet-5 LeNet-5 highest accuracy was achieved by [22], they got 99.04% as shown in their implementation [38] using MADBase dataset, with loss of 1%, 12% for training and testing respectively.[22]

Our proposed LeNet-5 was trained on ADBase and tested on MADBase and this improved MADbase test accuracy of 11% since we got accuracy of 99.16%,98.97%, 99.14%, and loss of 2.53&, 4.02% 2.81% training, validation and testing respectively, table 6.1 shows model confusion matrix and 6.4 shows classification accuracy of each class.

Table 6.4: LeNet-5 Model Class Trained on ADBase and Tested on MADBase Classification Accuracy

Class	precision	recall	f1-score	support
0	0.99	0.97	0.98	2000
1	0.98	1.00	0.99	2000
2	0.99	1.00	0.99	2000
3	1.00	0.99	1.00	2000
4	1.00	0.99	0.99	2000
5	0.99	0.99	0.99	2000
6	0.99	0.99	0.99	2000
7	1.00	1.00	1.00	2000
8	1.00	1.00	1.00	2000
9	0.99	0.99	0.99	2000

Googlenet and AlexNet Both Googlenet and Alexnet weren't used before to classify Arabic handwritten digits, and the results we achieved were higher than LeNet-5 even with ADBase test set

VGG Underfitting was noticed in VGG model, and the reason behind this low accurices on bith training and testing is VGG is made for large scale and complex images, so the model is too complex for the problem[54]

Table 6.5: Other Systems Results

Result	Algorithm	Dataset size
97.6% [29]	MLP[2]	500
99.85% [39]	SVM[3]	21120
99.04% [22]	LeNet-5	60000[4]

Chapter 7

System Analysis

7.1 Product Description

Our application will be implemented to classify Arabic handwritten digits and convert it from captured image to text in PDF form

7.1.1 System objectives

1. Convert captured written text to soft copy.
2. Classifying Arabic handwritten digits could help organizations that have old documents to convert them to text instead of images which offers more storage.

7.1.2 Functional Requirements

1. The app shall have the ability to access the phone's camera/gallery
2. The system should send an HTTP request to the web service (Machine learning engine) containing the image and the file form.
3. Machine learning engine shall be able to process the image and export the text in an HTTP response
4. The system shall be able to send the generated text to pdf, docs, or any chosen format

7.1.3 Non-Functional Requirements

1. The system shall be easy to use and shouldn't create any confusion for the user. (Usability)
2. User shall be able to upload an image of a text document from camera or phone gallery
3. The system shall be responsive and be compatible with the limited resources of smartphones. (Performance)

7.1.4 System Requirements

1. The application shall display options to upload the photo
    (a) The application shall ask the user to allow access to both gallery and camera

(b) The system shall send the uploaded image and chosen file form to the machine learning engine.

(c) The system should take the image whether it is uploaded from the gallery or taken at that moment and create an HTTP request.

(d) The system should send an HTTP request that contains both the uploaded image and the chosen format to the Machine Learning Engine (MLE)

2. Machine learning engine shall process the image

   (a) The machine learning engine shall apply pre-processing steps on the image if required

   (b) The MLE shall pass the processed images to the machine learning algorithm

   (c) The MLE shall generate a file of the chosen form that contains the algorithm output text

   (d) The MLE shall send the generated file to the application in the requested form by an HTTP response

3. The system should create a soft copy version of the image.

   (a) The system shall receive the soft copy from the Machine Learning Engine.

   (b) The system shall display the generated file from the chosen data form with the received data from the MLE.

7.2 Use Case Diagram

Our application will only one actor which is the user, and other used systems such as machine learning library (TensorFlow, or the PDF reader will be as included system

7.2.1 actors

- User:user interaction with the system represented as uploading image, downloading,sharing or deleting the generated/ converted text

7.2.2 Use Case Specification

- User use case specification

1. Upload image: the application allows the user to upload image from gallery or capture it from the camera
2. Download text: the user shall be able to download text from the application and store it on it's device
3. View text: the system allows the user to view the generated text as PDF share Text: user shall be able to share the text was generated on different social media apps

Figure 7.1 shows the use case diagram

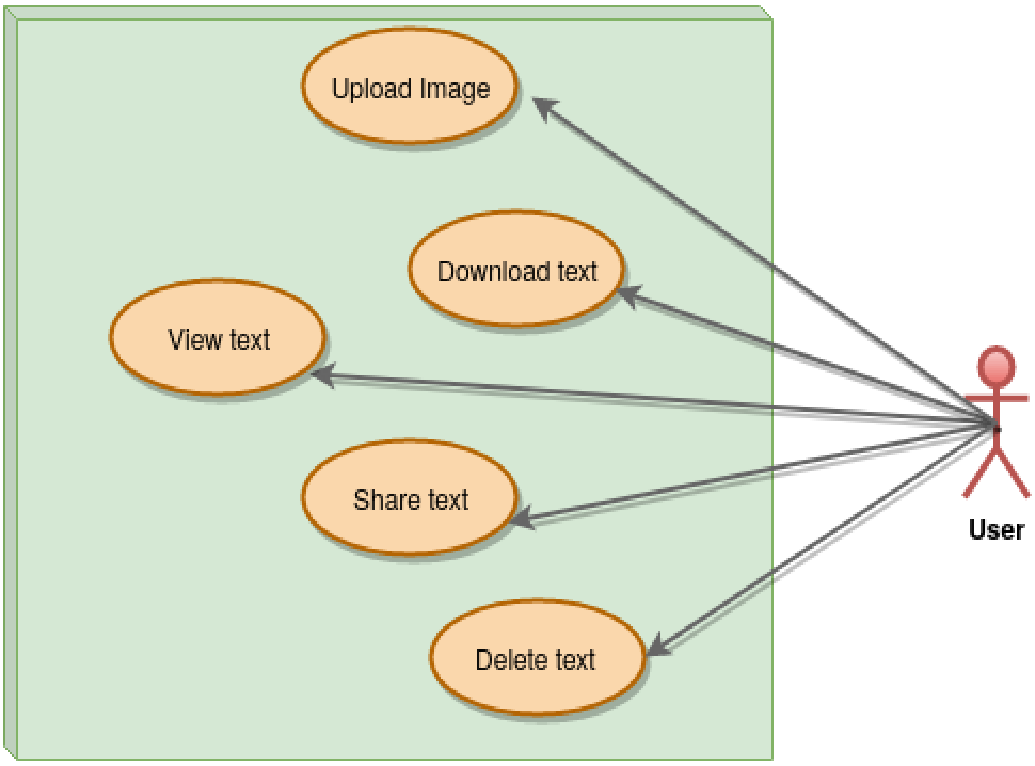


Figure 7.1: Use case Diagram

7.3 System Models

7.3.1 Class Diagram

Class diagram shows the structure of the system and figure 7.2 is the class diagram of our model and it shows classes, attributes, methods and the relationships between objects

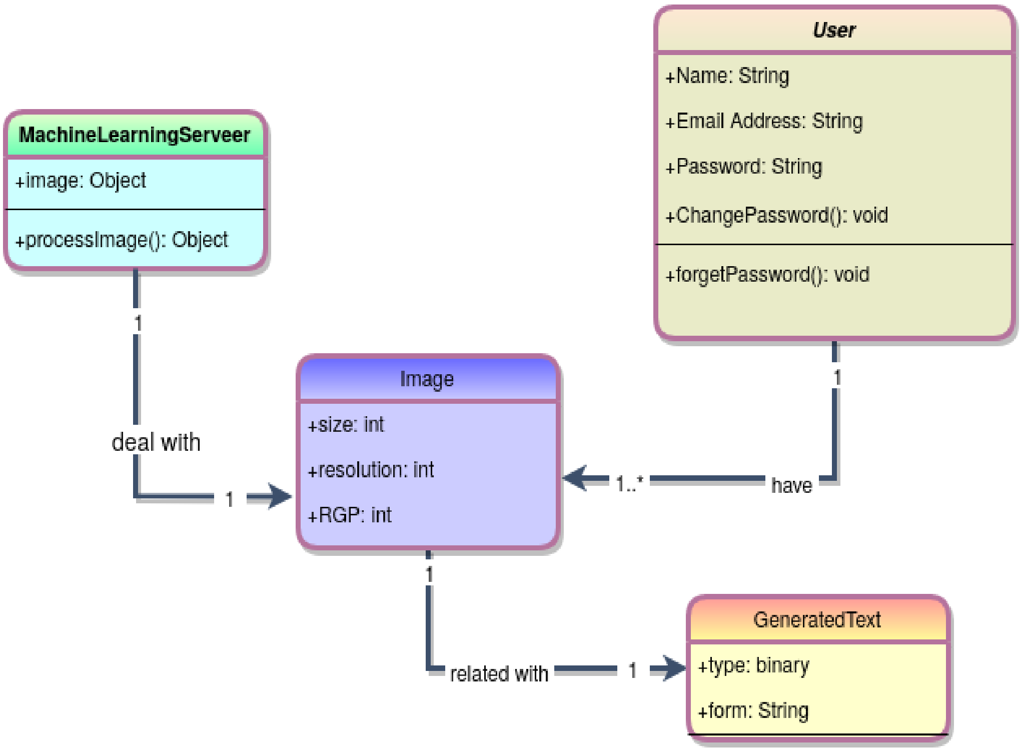


Figure 7.2: Class Diagram

7.3.2 Activity Diagram

An activity diagram shows the flow of dynamic aspects of the system from one activity to another, and figure 7.3 shows the activity diagram of our model.

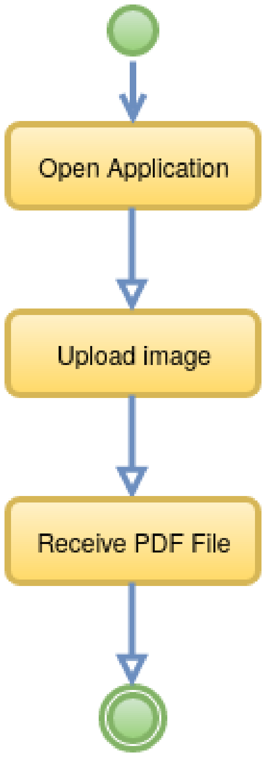


Figure 7.3: Activity Diagram

7.3.3 State Chart Diagram

state diagram show the behavior of class, figure 7.4 shows the state diagram of our model.

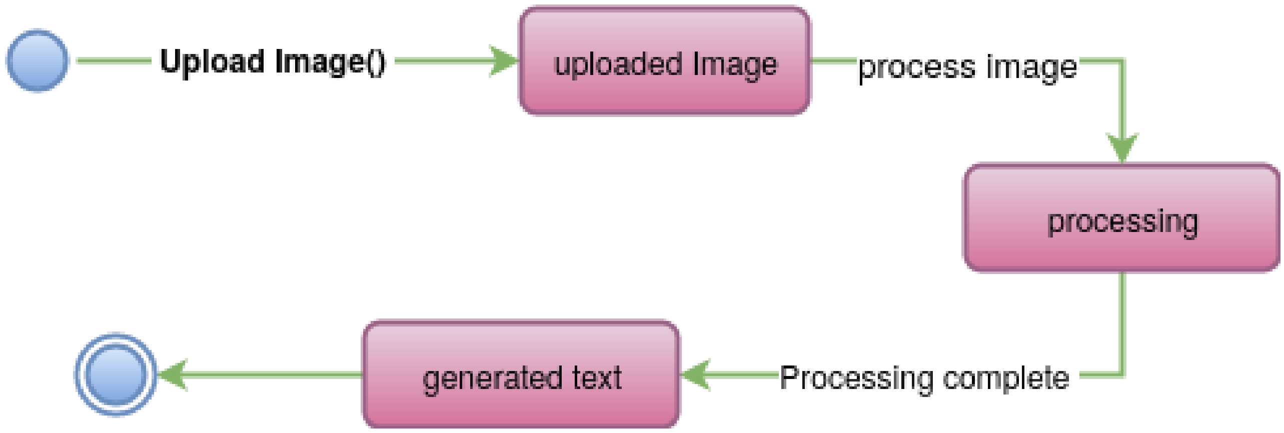


Figure 7.4: State Chart Diagram

7.3.4 Sequence Diagram

Sequence diagram show an interaction in a sequential order between objects, and the figure 7.5shows sequence diagram of upload image flow

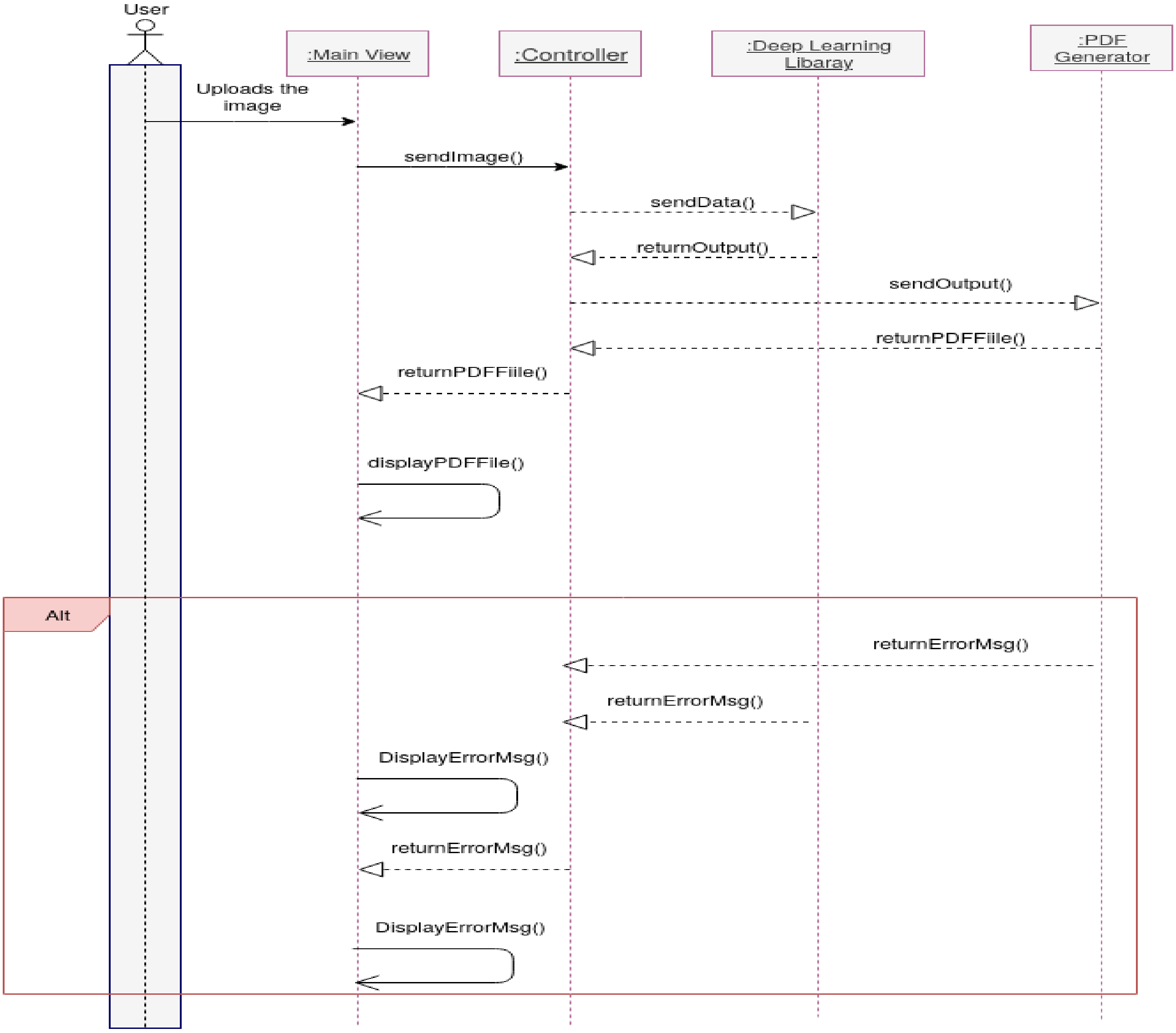


Figure 7.5: Sequence Diagram

7.4 System Architecture

7.4.1 Software Architecture

Implementation

This section will be separated into two parts:

1. Front-End
2. Back-End
3. APIs

Front-End: This part of application, were implemented as android application that receives image from user in two ways:

1. Import from Gallery
2. Take Image: by opening camera that contains a border to crop image directly once it's captured by user

Both ways requires user's permission to be accessed

Back-End: this part of implementation which contains all logic, and logic part is partitioned into two parts

1. Image processing, segmentation and conversion
2. Model Training, testing, and validation

Image Processing: This part was implemented using PIL[1][2], Open CV[3] -Python libraries-, to process image, resize, reshape it to have it converted in CSV form. same process was applied to dataset conversion and input images

Model Training, testing, validation: Keras libraries were used for this part, to train, fit, compile the model and produce it's metrics

APIs: Are the connection between Back and Front ends, Retrofit [4]has been used to implement APIs in user side "Android Application", and in the Back end, Falcon [5] has been used to implement APIs in server side

Back-End part has been implemented using python, and TensorFlow[6] libraries especially Keras[7] that was used to train, test, validate the model.[35]

[1] Image processing libraries in Python

[2] https://pillow.readthedocs.io/en/stable/

[3] https://docs.opencv.org/master/d6/d00/tutorial$_{py_r}oot.html$

[4] https://square.github.io/retrofit/

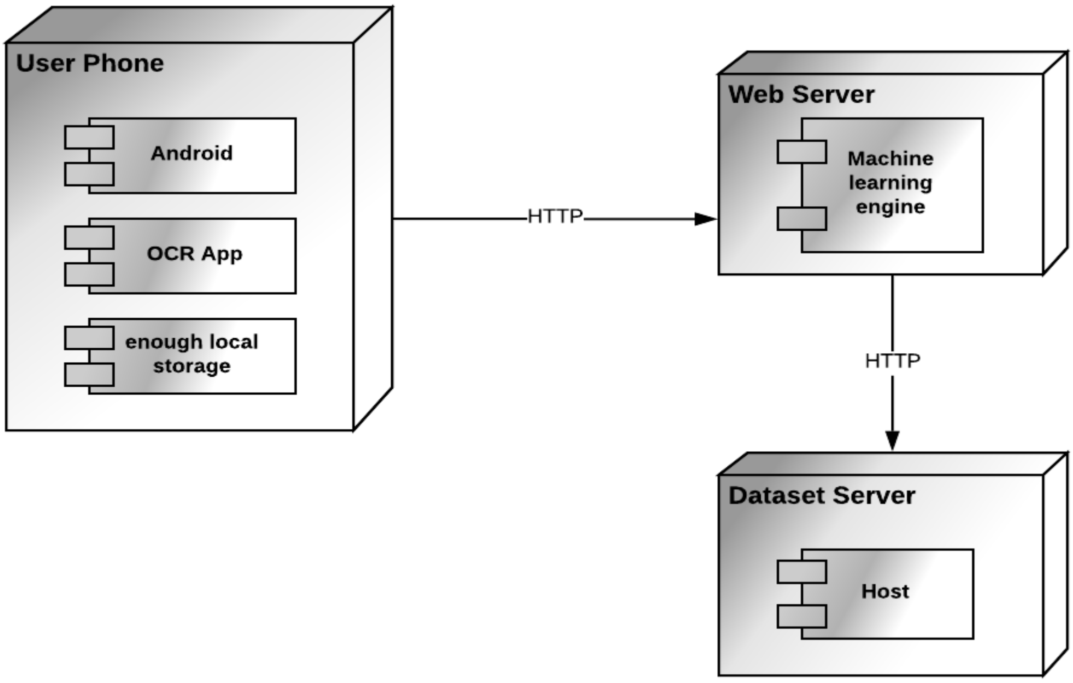
[5] Python web framework https://falconframework.org/

[6] https://www.tensorflow.org/

[7] https://keras.io/

7.4.2 Deployment Diagram

Deployment diagrams show the hardware processors for the system and the links of communication between the hardware component,and the following figure the deployment diagram of our model.



7.4.3 The N-Tiers diagram

The N tiers diagram will be partitioned into User and Server sections:

1. User
   - (a) UI Layer
   - (b) Device Service Layer
   - (c) Connector Layer
2. Server
   - (a) Interface Layer
   - (b) Business Layer
   - (c) Service Layer
   - (d) Data Access Object Layer(DAO)

UI Layer: Is the first interaction between user and system, where user uploads the image

Data service Layer: Represents the connection between system and device local storage, and how it's been accessed

Connector Layer and Interface Layer Are first and second lines communication -respectively-, between both user and server sides where the user inputs and server outputs are exchanged using APIs

Business Layer: The layer that orchestrates inputs to be sent to it's special component in service layer

Service Layer: Is the place that contains all logic, such:

1. PDF Generator
2. Dataset Services
3. Machine Learning Services

Data Access Object Service (DAO) :Is Data exchange layer that contains all datasete, it directly communicates with Dataset Services component in service layer. Figure 7.6 shows layers and how it connects with each other

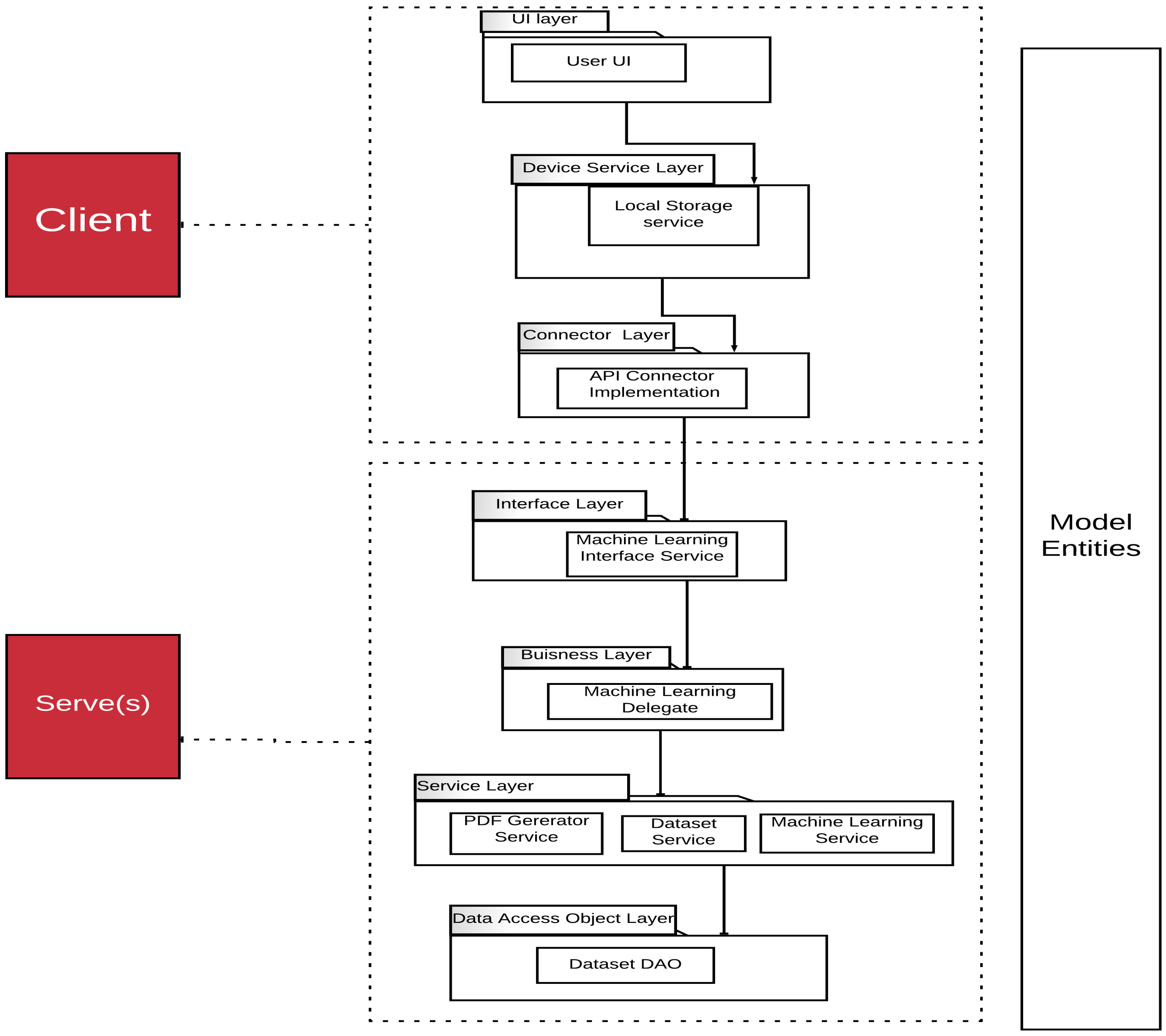


Figure 7.6: N-Tiers diagram

Chapter 8

Conclusion

Automated softwares to classify Arabic handwritten digits are very important, that could be used in card and cheque readers. This research paper proposed a system to classify Arabic handwritten digits using different CNN architectures, on ADBASE and MADBASE datasets and highest accuracy on both ADBASE and MADBASE test set was achieved by googlenet,both uderfitting and overfitting were noticed while experiment.

Overfitting occurred because of the low quality images in test set of ADBASE comparing to ones in MADBASE training set or ADBASE it self

Underfitting occurred when we used VGG model, since its designed for complex and large scale images,unlike the datasets we used.

Many regularization techniques were deployed on the system, such as dropout, and batch normalization, which improved system efficiency and performance. All system parts were implemented using Tenorflow Keras

As future work, we plan to implement our models on larger datasets that contains huge variation in it's images and dive deeper in other CNN architectures to examine which ones are suitable to be used on such datasets, and which of them could achieve higher accurices on training, validation and testing

Finally, in later stages of this research we aim to use CNN architectures to classify Arabic handwritten characters and words

Bibliography

[1] Arabic handwritten characters dataset. https://www.kaggle.com/mloey1/ahcd1.

[2] Arabic handwritten characters dataset. https://www.kaggle.com/raipiyush558/arabic.

[3] Cmultilayer perceptron.

[4] Difference between classification and regression in machine learning. https://machinelearningmastery.com/classification-versus-regression-in-machine-learning/.

[5] The distance function effect on k-nearest neighbor classification for medical datasets. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4978658/.

[6] Machine learning course. Published by Professor Radi Jarrar in Machine Learning course.

[7] Supervised learning methods using python. https://medium.com/@himanshuit3036/supervised-learning-methods-using-python-bb85b8c4e0b7.

[8] Supervised vs unsupervised learning: Key differences.

[9] Unsupervised learning. https://www.mathworks.com/discovery/unsupervised-learning.html.

[10] Unsupervised learning, howpublished = https://medium.com/@aditi22prerna/unsupervised-learning-a24caf362e79,.

[11] Review of online amp; offline character recognition. 4, May 2015.

[12] Sherif Abdleazeem and Ezzat El-Sherif. Arabic handwritten digit recognition. *IJDAR*, 11:127–141, 12 2008.

[13] A. Alaei, U. Pal, and P. Nagabhushan. Using modified contour features and svm based classifier for the recognition of persian/arabic handwritten numerals. In *2009 Seventh International Conference on Advances in Pattern Recognition*, pages 391–394, 2009.

[14] A. Ashiquzzaman and A. K. Tushar. Handwritten arabic numeral recognition using deep learning neural networks. In *2017 IEEE International Conference on Imaging, Vision Pattern Recognition (icIVPR)*, pages 1–4, 2017.

[15] Akm Ashiquzzaman, Abdul Kawsar Tushar, Ashiqur Rahman, and Farzana Mohsin. An efficient recognition method for handwritten arabic numerals using cnn with data augmentation and dropout. In Valentina Emilia Balas, Neha Sharma, and Amlan Chakrabarti, editors, *Data Management, Analytics and Innovation*, pages 299–309, Singapore, 2019. Springer Singapore.

[16] Arindam Chaudhuri, Krupa Mandaviya, Pratixa Badelia, and Soumya Ghosh. *Optical Character Recognition Systems*, pages 9–41. 12 2017.

[17] Siddharth Das. Cnn architectures: Lenet, alexnet, vgg, googlenet, resnet and more. . . .

[18] Siddharth Das. Cnn architectures: Lenet, alexnet, vgg, googlenet, resnet and more. . . .

[19] Siddharth Das. Cnn architectures: Lenet, alexnet, vgg, googlenet, resnet and more. . . .

[20] K.-L Du and M.N.s Swamy. *Recurrent Neural Networks*, pages 337–353. 12 2014.

[21] Priya Dwivedi. Understanding and coding a resnet in keras.

[22] Ahmed Elsawy, Hazem El-Bakry, and Mohamed Loey. Cnn for handwritten arabic digits recognition based on lenet-5, 10 2016.

[23] Carl Ernst. The global significance of arabic language and literature. *Religion Compass*, 7, 06 2013.

[24] Yasutaka Furusho and Kazushi Ikeda. *Effects of Skip-Connection in ResNet and Batch-Normalization on Fisher Information Matrix*, pages 341–348. 01 2020.

[25] Rohith Gandhi. Support vector machine — introduction to machine learning algorithms.

[26] Chih-Chun) Gary(Chang. Cnn architectures — vggnet.

[27] Enzo Grossi and Massimo Buscema. Introduction to artificial neural networks. *European journal of gastroenterology hepatology*, 19:1046–54, 01 2008.

[28] Karez Hamad and Mehmet Kaya. A detailed analysis of optical character recognition technology. *International Journal of Applied Mathematics, Electronics and Computers*, 4:244–244, 12 2016.

[29] A. Harifi and A. Aghagolzadeh. A new pattern for handwritten persian/arabic digit recognition.

[30] Aditya Jain, Gandhar Kulkarni, and Vraj Shah. Natural language processing. *International Journal of Computer Sciences and Engineering*, 6:161–167, 01 2018.

[31] Sergios Karagiannakos. The secrets behind reinforcement learning.

[32] Dimitris Katisios. Convnets.

[33] Amber Khan, Mariam Usmani, Nashrah Rahman, and Dinesh Prasad. Pre-processing images of public signage for ocr conversion. *Journal of Signal and Information Processing*, 10:1–11, 01 2019.

[34] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[35] Vijay Kumar and Priyanka Gupta. Importance of statistical measures in digital image processing. *International Journal of Emerging Technology and Advanced Engineering*, 2, 08 2012.

[36] I. A. Lawal, R. E. Abdel-Aal, and S. A. Mahmoud. Recognition of handwritten arabic (indian) numerals using freeman's chain codes and abductive network classifiers. In *2010 20th International Conference on Pattern Recognition*, pages 1884–1887, 2010.

[37] Mohamed Loey. Ahdd1.

[38] Mohamed Loey. Lenet-5 – a implemetation.

[39] S. A. Mahmoud. Arabic (indian) handwritten digits recognition using gabor-based features. *2008 International Conference on Innovations in Information Technology*, pages 683–687, 2008.

[40] Makki Maliki, Sabah Jassim, Naseer Al-Jawad, and Harin Sellahewa. Arabic handwritten: Pre-processing and segmentation. *Proc SPIE*, 8406:84060D, 04 2012.

[41] Thomas Michelle and Hill McGraw. *Machine Learning.* 1997.

[42] DITI MODI. Diving into k-means....

[43] Keiron O'Shea and Ryan Nash. An introduction to convolutional neural networks. *ArXiv e-prints*, 11 2015.

[44] El Melhaoui Ouafae, Mohamed Hitmy, and Fairouz Lekhal. Arabic numerals recognition based on an improved version of the loci characteristic. *International Journal of Computer Applications*, 24:36–41, 06 2011.

[45] Muhammad Rizwan. Alexnet: The architecture that challenged cnns.

[46] Muhammad Rizwan. Lenet-5 – a classic cnn architecture.

[47] Sabyasachi Sahoo. Residual blocks — building blocks of resnet.

[48] Sabyasachi Sahoo. Skip connections and residual blocks.

[49] P. P. Selvi and T. Meyyappan. Recognition of arabic numerals with grouping and ungrouping using back propagation neural network. In *2013 International Conference on Pattern Recognition, Informatics and Mobile Engineering*, pages 322–327, 2013.

[50] Christian Szegedy. Going deeper with convolutions.

[51] Christian Szegedy. Googlenet.

[52] M. Takruri, R. Al-Hmouz, and A. Al-Hmouz. A three-level classifier: Fuzzy c means, support vector machine and unique pixels for arabic handwritten digits. In *2014 World Symposium on Computer Applications Research (WSCAR)*, pages 1–5, 2014.

[53] Subham Tewari. Cnn architecture series — alexnet with implementation(part ii).

[54] Subham Tewari. Cnn architecture series — vgg-16 with implementation(part i).

[55] G. Vamvakas, B. Gatos, Nikolaos Stamatopoulos, and Stavros Perantonis. A complete optical character recognition methodology for historical documents. *Document Analysis Systems, IAPR International Workshop on*, 0:525–532, 09 2008.

[56] Khaled Younis. Arabic handwritten character recognition based on deep convolutional neural networks. *Jordanian Journal of Computers and Information Technology (JJCIT)*, 3, 03 2018.

Chapter 9

Appendices

9.1 Use Case Specifications

Scenario One: Upload Image

- Normal Scenario

  Younes wants to upload image from the gallery. so he press the upload image, and chose the image to upload from the gallery

- Alternative Scenario

  Younes doesn't have the image in the gallery and he wants to capture it, so he clicks the take image button then take the image and select done

- Error Scenario

  1. While uploading, Younes chooses file that it's not image, so system will display error message for him
  2. Phone is not connected to the Internet, he will receive error message

Table 9.1: Upload Image

Actor	User
Description	The user uploads the image (s)he wants using camera or gallery
Pre-requisite	The user gave the permission to access gallery or camera, based on the way he wants to upload the image in
Data	Image
Flow of event	(s)he has to chose to way to upload, then uploads the image
Trigger	User press the "Upload" button to send the image to the server
Post-Condition	Image uploaded to the server successfully

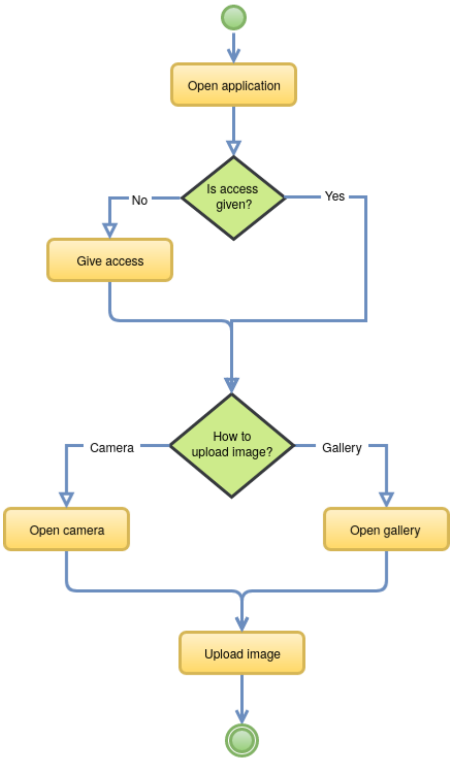


Figure 9.1: Upload image Activity Diagram

Scenario Two: Download generated text

- Normal Scenario

  After Younes uploaded the image he will receive a PDF file, then clicks download button

- Alternative Scenario

  Younes can view the text without download it.

- Error Scenario

Table 9.2: Download Text

Actor	User
Description	User downloads the PDF file by pressing the ”Download button”
Pre-requisite	Good Internet connection & enough space in phone storage
Data	PDF file
Flow of event	(s)he has press Download button
Trigger	User press the ”Download” button to download text
Post-Condition	PDF file saved in the phone storage

s The following figure shows the Download activity diagram

Scenario Three: Share Scenario

Younis wants to share text on, so he clicks share button then chooses the social media type he wants to share on

Table 9.3: Share File

Actor	User
Description	The user wants to share
Pre-requisite	1. Good Internet Connection 2. Social media app must be downloaded 3. Access to social media apps
Data	PDF File
Trigger	user press the ”share” button and chose the social media type (s)he wants to share on

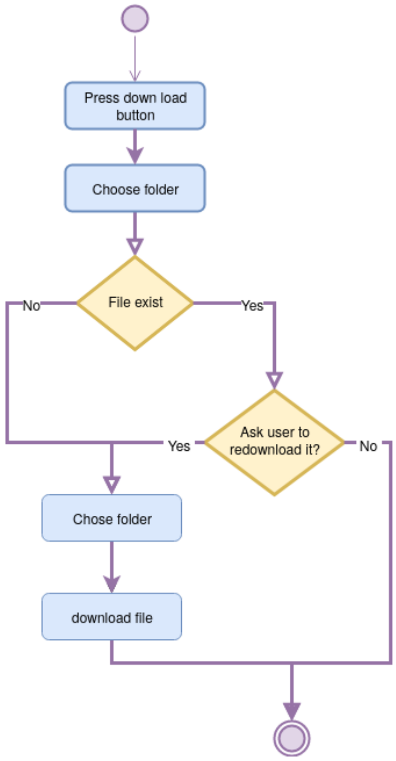


Figure 9.2: Download Activity Diagram

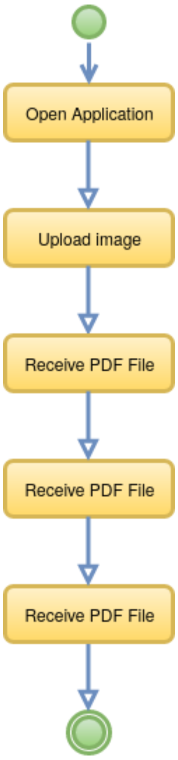


Figure 9.3: share Activity Diagram

Scenario Four: View Scenario

Younis wants to view the text without downloading item so he press ”Display” button

Table 9.4: View File

Actor	User
Description	The user wants to view the text without installing it
Pre-requisite	1. Good Internet Connection 2. Access to phone storage
Data	PDF File
Trigger	user press the ”Display Text” button to view the text
Post-Condition	Image uploaded to the server successfully

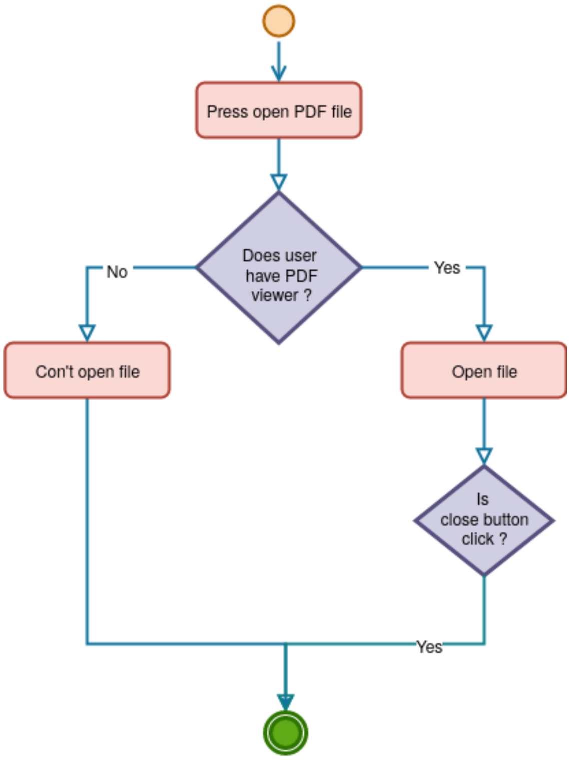
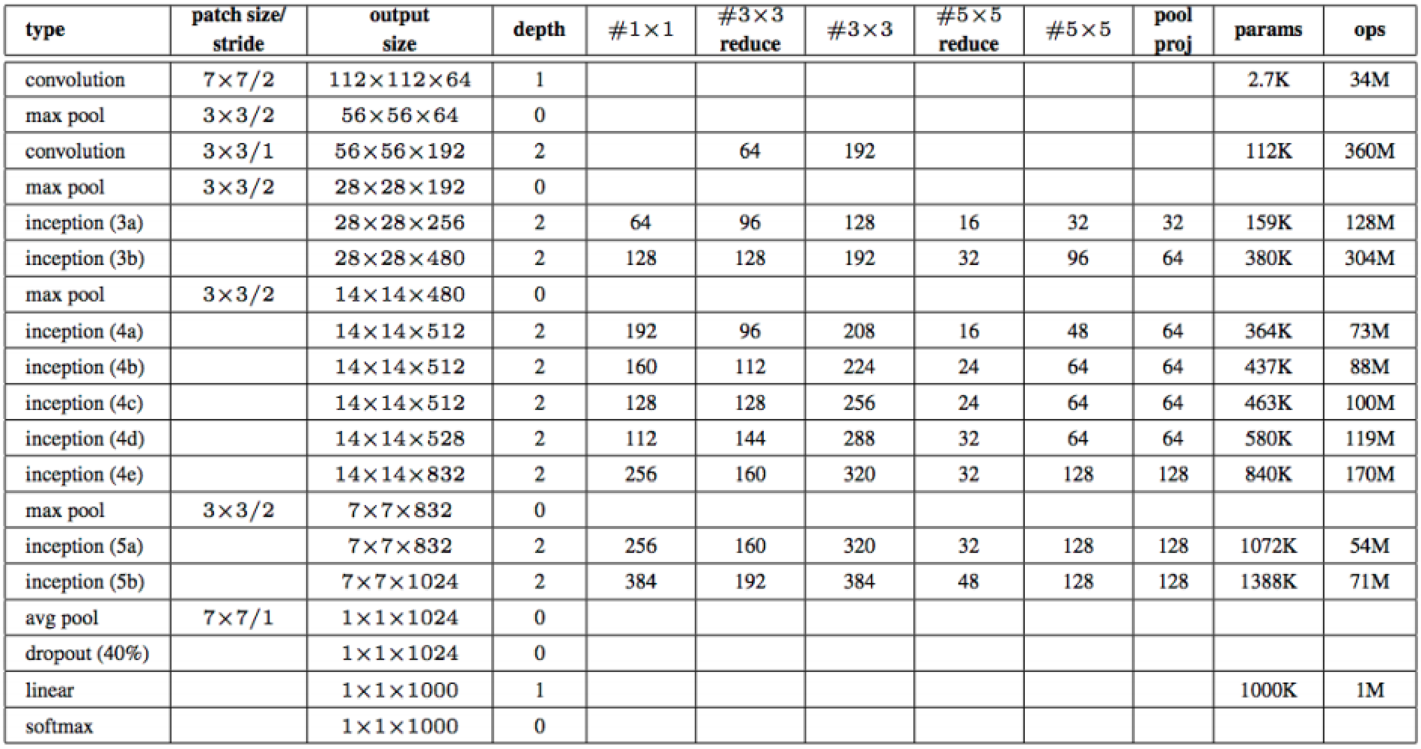


Figure 9.4: View Activity Diagram

9.2 CNN Architectures

Following section will CNN architectures Diagrams and tables

9.2.1 GoogLeNet



type	patch size/ stride	output size	depth	#1×1	#3×3 reduce	#3×3	#5×5 reduce	#5×5	pool proj	params	ops
convolution	7×7/2	112×112×64	1							2.7K	34M
max pool	3×3/2	56×56×64	0								
convolution	3×3/1	56×56×192	2		64	192				112K	360M
max pool	3×3/2	28×28×192	0								
inception (3a)		28×28×256	2	64	96	128	16	32	32	159K	128M
inception (3b)		28×28×480	2	128	128	192	32	96	64	380K	304M
max pool	3×3/2	14×14×480	0								
inception (4a)		14×14×512	2	192	96	208	16	48	64	364K	73M
inception (4b)		14×14×512	2	160	112	224	24	64	64	437K	88M
inception (4c)		14×14×512	2	128	128	256	24	64	64	463K	100M
inception (4d)		14×14×528	2	112	144	288	32	64	64	580K	119M
inception (4e)		14×14×832	2	256	160	320	32	128	128	840K	170M
max pool	3×3/2	7×7×832	0								
inception (5a)		7×7×832	2	256	160	320	32	128	128	1072K	54M
inception (5b)		7×7×1024	2	384	192	384	48	128	128	1388K	71M
avg pool	7×7/1	1×1×1024	0								
dropout (40%)		1×1×1024	0								
linear		1×1×1000	1							1000K	1M
softmax		1×1×1000	0								

Figure 9.5: GooLeNet Table[32]

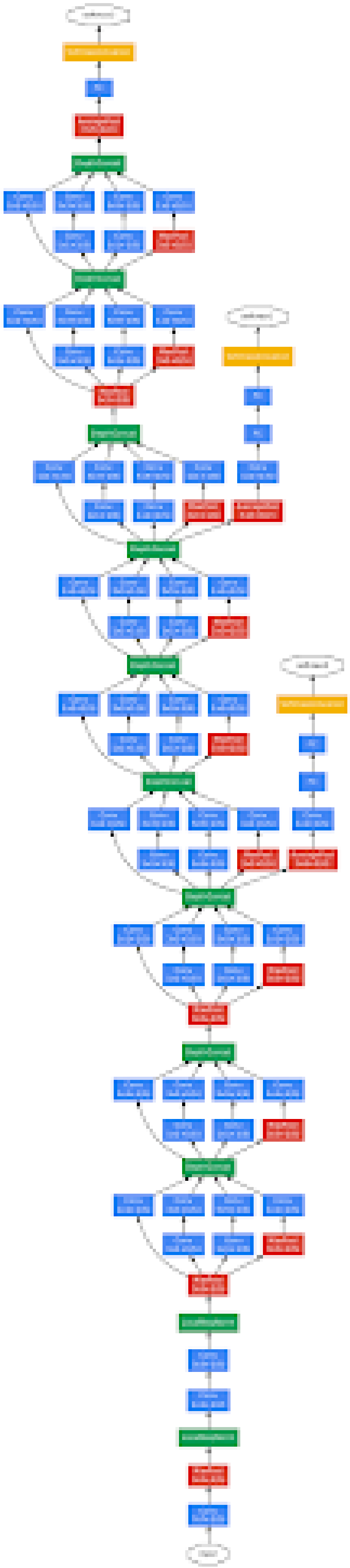


Figure 9.6: GooLeNet Full Architecture[32]

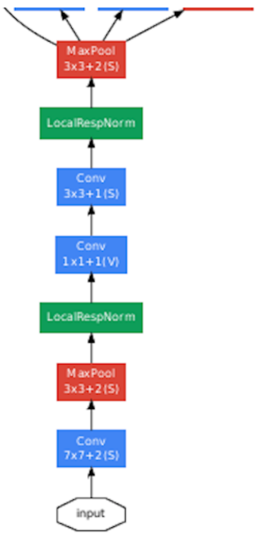


Figure 9.7: GooLeNet Model[32]

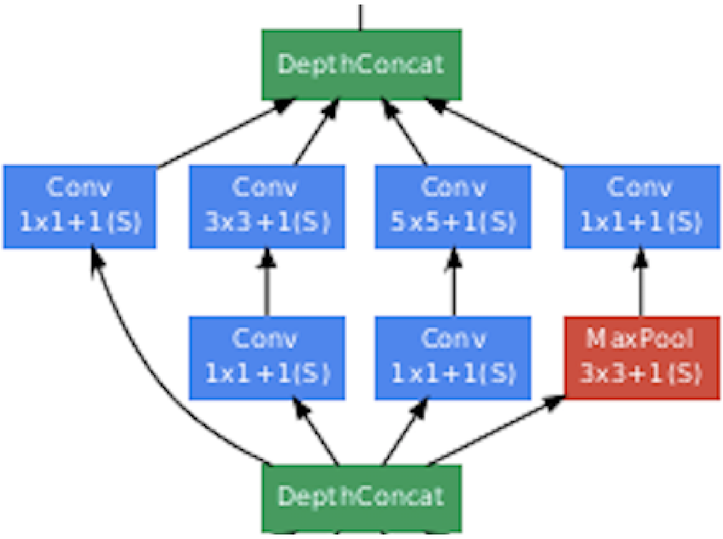


Figure 9.8: GooLeNet Model[32]

9.2.2 Alexnet

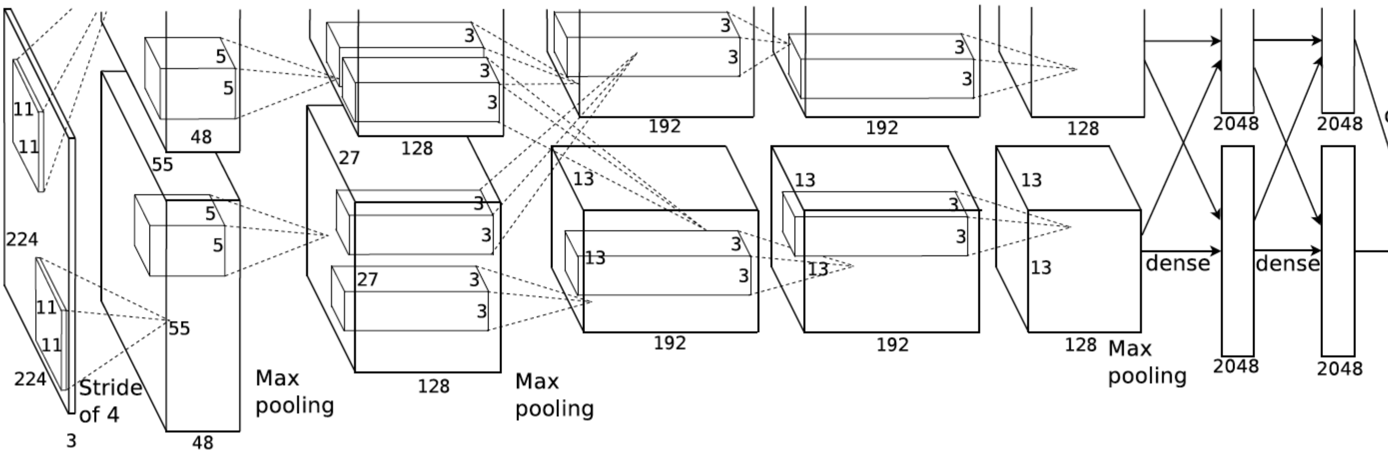
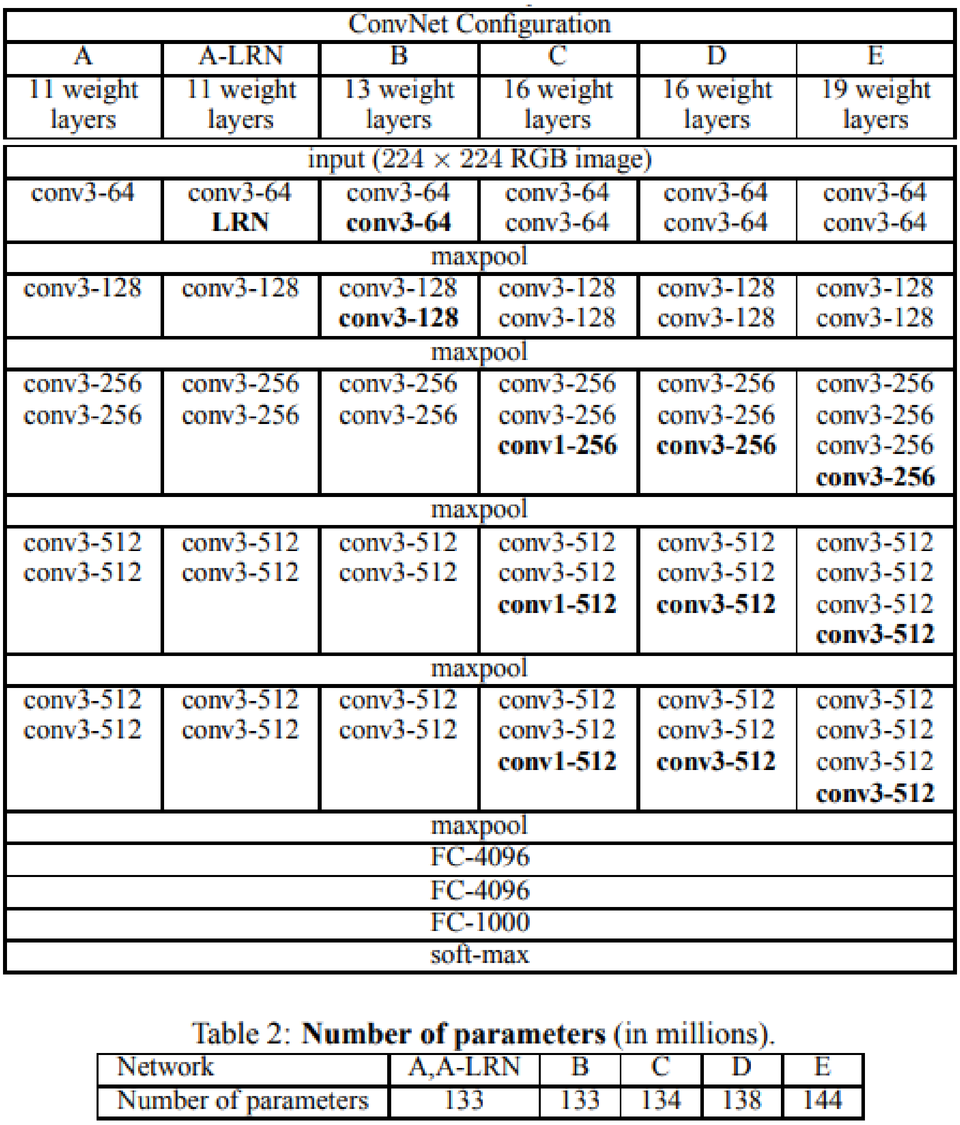


Figure 9.9: AlexNet Model[32]

9.2.3 VGG



ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Table 2: **Number of parameters** (in millions).

Network	A,A-LRN	B	C	D	E
Number of parameters	133	133	134	138	144

Figure 9.10: VGG Table[32]

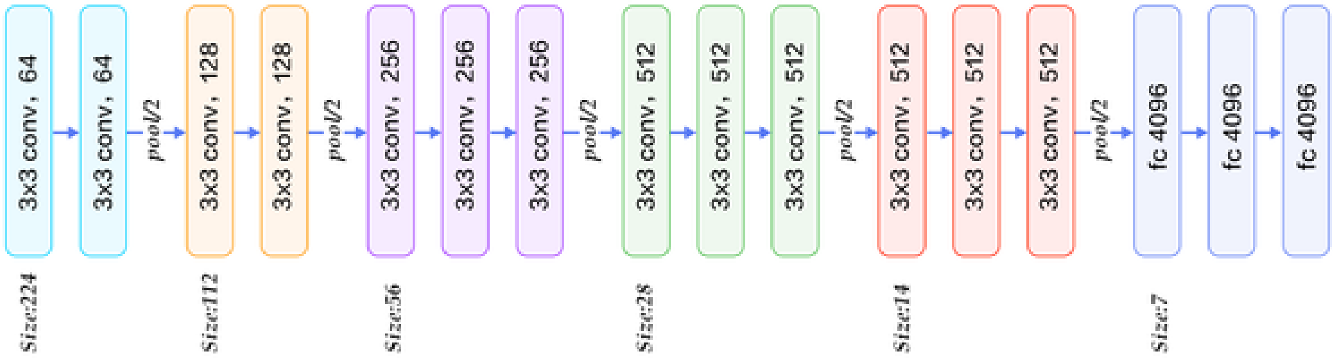
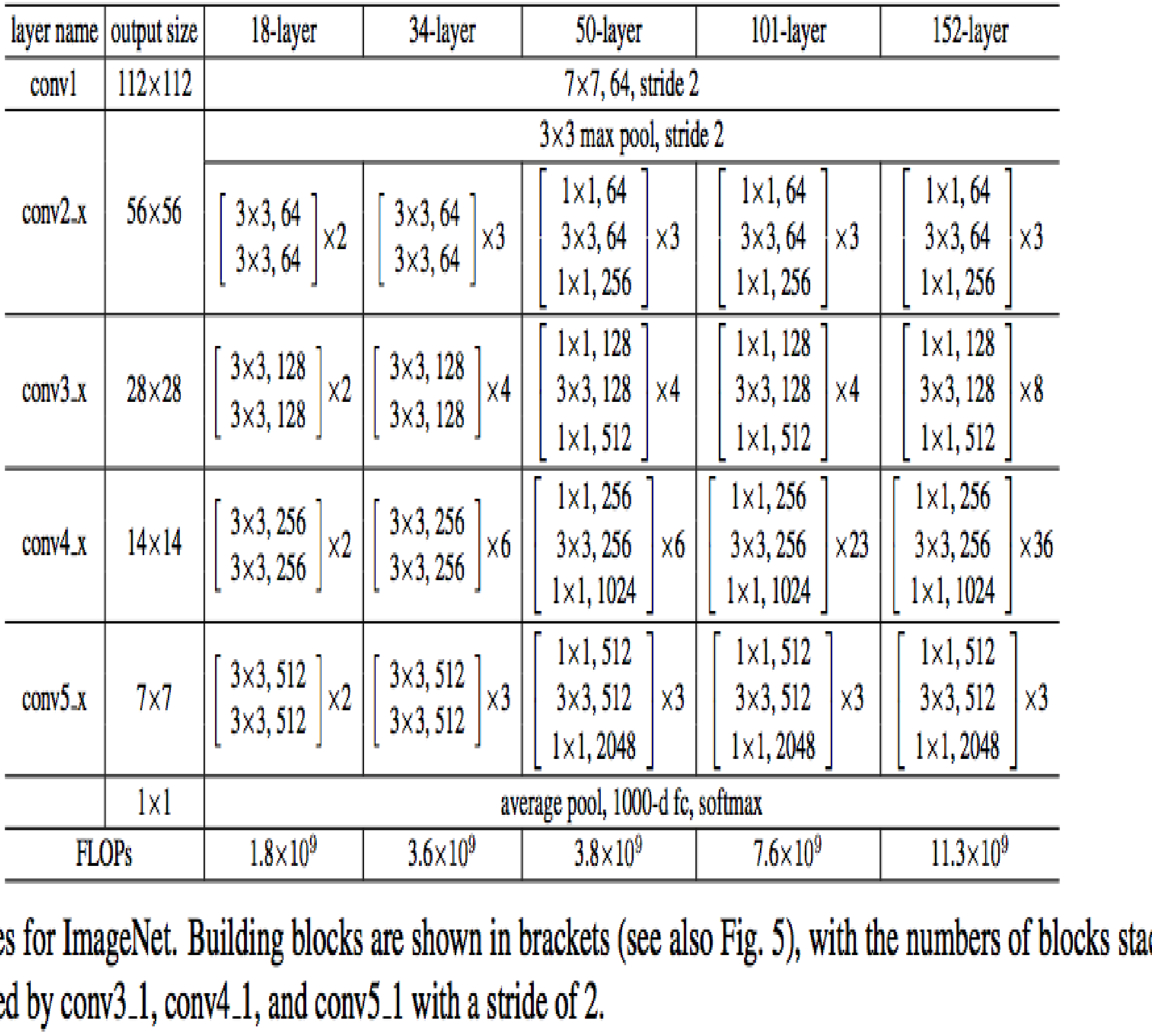


Figure 9.11: VGG Model[32]

9.2.4 ResNet



layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix}3\times3, 64\\3\times3, 64\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 64\\3\times3, 64\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 64\\3\times3, 64\\1\times1, 256\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 64\\3\times3, 64\\1\times1, 256\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 64\\3\times3, 64\\1\times1, 256\end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix}3\times3, 128\\3\times3, 128\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 128\\3\times3, 128\end{bmatrix}\times4$	$\begin{bmatrix}1\times1, 128\\3\times3, 128\\1\times1, 512\end{bmatrix}\times4$	$\begin{bmatrix}1\times1, 128\\3\times3, 128\\1\times1, 512\end{bmatrix}\times4$	$\begin{bmatrix}1\times1, 128\\3\times3, 128\\1\times1, 512\end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix}3\times3, 256\\3\times3, 256\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 256\\3\times3, 256\end{bmatrix}\times6$	$\begin{bmatrix}1\times1, 256\\3\times3, 256\\1\times1, 1024\end{bmatrix}\times6$	$\begin{bmatrix}1\times1, 256\\3\times3, 256\\1\times1, 1024\end{bmatrix}\times23$	$\begin{bmatrix}1\times1, 256\\3\times3, 256\\1\times1, 1024\end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix}3\times3, 512\\3\times3, 512\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 512\\3\times3, 512\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 512\\3\times3, 512\\1\times1, 2048\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 512\\3\times3, 512\\1\times1, 2048\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 512\\3\times3, 512\\1\times1, 2048\end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

Table 1. Architectures for ImageNet. Building blocks are shown in brackets (see also Fig. 5), with the numbers of blocks stac sampling is performed by conv3_1, conv4_1, and conv5_1 with a stride of 2.

Figure 9.12: ResNet Table[32]

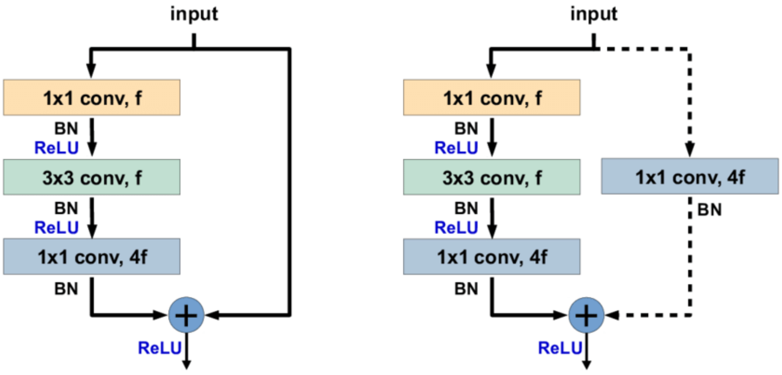


Figure 9.13: ResNet Model[32]

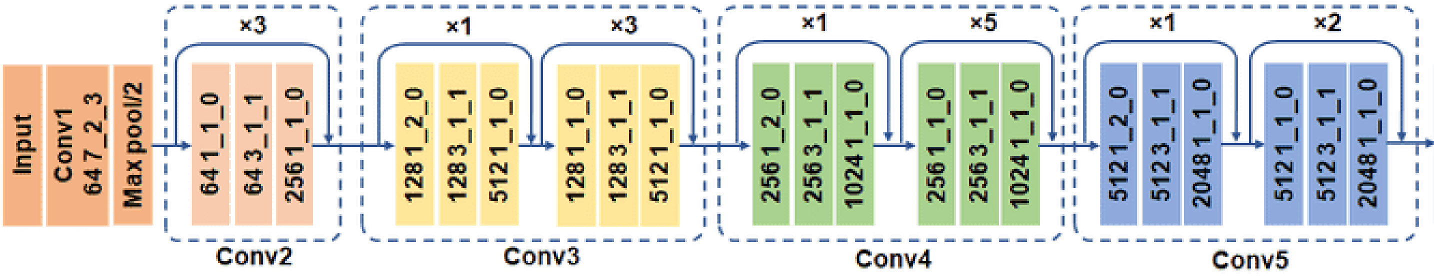


Figure 9.14: ResNet Model[32]