University of Windsor

Masters of Applied Computing
COMP4730-1-R-2022F
MACHINE LEARNING

PROJECT REPORT

For

IMPLEMENTATION OF DIFFERENT CNN ARCHITECTURES ON MNIST DATASET

PREPARED BY

SUBHAM GUPTA	110093436
ABY KURUVILA	110089795
AJAY PALLIVATHUKKAL MANOJ	110089283

SUBMITTED TO

DR. ROBIN GRAS

OCTOBER 17th, 2022

University of Windsor

Masters of Applied Computing
COMP4730-1-R-2022F
MACHINE LEARNING

TABLE OF CONTENTS

1. INTRODUCTION

The field of image classification is vast. Despite the fact that there have already been a lot of research studies performed, new and improved methodologies are always welcome so that accuracy can be improved and run time as well as complexity can be minimized. In the area of image classification, handwritten digit classification is of particular interest due to its use in various fields, such as the recognition of handwritten digits on cheques and postal cards by computers. Individuals write digits differently, adding complexity to classification. Several Deep Learning based classifiers have been proposed for the same.

In this report we treat LeCun's Convolutional Neural Network Architecture also known as LeNet 5 as the baseline model and we compare this baseline model with 2 different versions of Convolutional Neural Networks(CNN). Overall idea is to classify an input image into one of 10 classes i.e digits ranging from 0-9. In depth analysis of both the CNN architectures along with the outputs produced by both these models gives us an estimate of which CNN architecture works better and how we can improve the accuracy by increasing the depth.

2. MNIST DATA SET

The MNIST DATA SET is most widely used by Computer Vision scientists, specially by individuals who have just started exploring around image processing and NIST's original database samples were modified to create it. This dataset has images of handwritten digits from 0-9. In total there are 70,000 such digital images and it's further segregated into 60,000 training image data set and 10,000 testing image data set. All these images are normalized 28*28 grayscale images. Below is a sample of MNIST data set.

Since we are using MNIST data set from Tensorflow's dataset collection our data comes segregated into 60,000 training samples and 10,000 testing samples. In order to ensure the generalization of training data samples we need to have a validation data set and hence we split the training data into 2 sets i.e. actual training data(80%) and validation data(20%). We use this actual training data to train different CNN architectures and then test these models and LeNet 5 model on unknown data set i.e. the test data set.

3. DATA CLEANING

Before proceeding ahead with the modelling part we have primarily performed three cleaning operations. In order to represent the grayscale nature of the image on a separate axis we have resized the image from 28 * 28 to 28 * 28 * 1 size thereby allowing the CNN architecture to consider the input image as single channel image. We have even normalized the pixel values of the image by dividing the pixels by 255, thus limiting the range of each pixel value between 0 to 1. Finally one hot encoding of the output label i.e. the y value is also done.

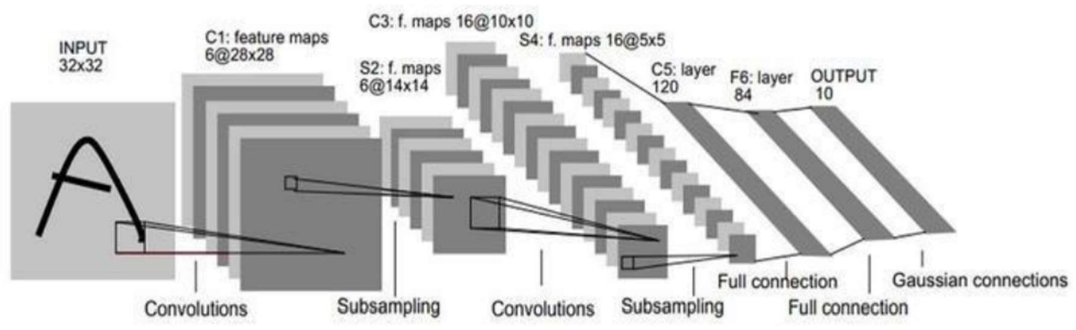
4. Le Net 5 ARCHITECTURE

Le Net 5 CNN Architecture was proposed by Yann LeCun in the year 1989 and is one of the oldest CNN architecture and we have used this CNN architecture as our baseline model. The first layer is also known as the convolutional layer and it takes a 28 * 28 * 1 image as input and using 6 Feature Map with each filter of size 5 * 5 to produce 6 maps of size 28 * 28 as output. Tanh is the activation function which is used in all the convolutional layer. The second layer which is also know as the pooling layer reduces the size from 28 * 28 to 14 * 14 using filter of size 2 * 2, hence we get 6 Feature Maps of size 14 * 14 as the output. The third layer is again a convolutional layer and it takes the incoming 14 * 14 * 6 image as input and using 16 Feature Maps with each filter of size 5 * 5, it produces 16 maps of size 10 * 10 as output. The fourth layer again reduces the size of the input image using 2 * 2 filter to give 16 maps of 5 * 5 image as output. The fifth layer is a fully connected layer with 120 units and it produces 120 maps of 1 * 1 as output. The sixth layer again is a fully connected layer with 84 units and it produces 84 maps of 1 * 1 as output. Finally the last most layer i.e the output layer is a softmax layer that produces 10 possible values between 0 to 9 corresponding to 10 different digits. LeNet 5 achieves an accuracy of 98.54 % and records a loss of 82.09%.

<u>LeNet 5 Architecture by LeCun</u>

LAYER	LAYER TYPE	FILTER SIZE	NO OF FEATURE MAP	OUTPUT IMAGE SIZE	NO OF PARAMETERS TO TRAIN	ACTIVATION FUNCTION USED
INPUT	IMAGE		1	32 * 32		
C1	CONV	5 * 5	6	28 * 28	156	Tanh
P2	POOL	2 * 2	6	14 * 14		
C3	CONV	5 * 5	16	10 * 10	2416	Tanh
P4	POOL	2 * 2	16	5 * 5		
	FLATTEN			400		
F5	FULLY CONNECTED		120	120	48120	Tanh
F6	FULLY CONNECTED		84	84	10164	Tanh

OUTPUT	SOFTMAX		10	10	850	Softmax



Lecun, Y.; Bottu, L. ;Bengio, Y.; Haffner, P. (1998). "Gradient-based learning applied to document recognition" Proceedings of IEEE. 86(11):2278-2324 Vol.: 86, Issue: 11, Nov. 1998

5. DIFFERENT MODELS & ARCHITECTURES

MODEL 1:

In the First CNN Architecture that we propose has a CONV layer as the 1st layer which accepts input of size 28 * 28 *1 and than with the help of 32 feature maps with each filter of size 3 * 3 it produces an output image of size 26 * 26 * 32. All the CONV layers use Relu as the activation function. The second layer i.e the POOL layer takes this input image and it reduces it's size by half t 13 * 13 * 32 with the help of 2 * 2 filter. Now we flatten this output to a 5408 dimension vector. Third layer is a FULLY CONNECTED layer with 100 units and it takes this 5408 dimension vector as input and produces 100 feature maps of size 1 * 1 as the output. Finally, we have a SOFTMAX layer which produces 10 possible values corresponding to 10 digits ranging from 0-9. Model 1 achieves an accuracy of 98.55% and records a loss of 6.12%.

MODEL 1 Architecture

LAYER	LAYER TYPE	FILTER SIZE	NO OF FEATURE MAP	OUTPUT IMAGE SIZE	NO OF PARAMETERS TO TRAIN	ACTIVATION FUNCTION USED

INPUT	IMAGE		1	28 * 28		
C1	CONV	3 * 3	32	26 * 26	320	Relu
P2	POOL	2 * 2	32	13 * 13		
	FLATTEN			5408		
F3	FULLY CONNECTED		100	100	540900	Relu
OUTPUT	SOFTMAX		10	10	1010	Softmax

MODEL 2:

Addition of extra layers i.e. by increasing the model depth we help the model in achieving greater accuracy. Hence in this CNN Architecture that we propose has a CONV layer as the 1st layer which accepts input of size 28 * 28 *1 and than with the help of 32 feature maps with each filter of size 3 * 3 it produces an output image of size 26 * 26 * 32. All the CONV layers use Relu as the activation function. The second layer i.e the POOL layer takes this input image and it reduces it's size by half t 13 * 13 * 32 with the help of 2 * 2 filter. The third layer is a CONV layer which with the help of 64 feature maps and with each filter of size 3 * 3 produces an output image of size 11 * 11 * 64. Fourth layer is again a CONV layer which with the help of 64 feature maps and with each filter of size 3 * 3 produces an output image of size 9 * 9 * 64. Fifth layer is a POOL layer which reduces the size of input image by half i.e from 9 * 9 to 4 * 4 by using 2 * 2 filter. Now we flatten this output to a 1024 dimension vector. Sixth layer is a FULLY CONNECTED layer with 100 units and it takes this 1024 dimension vector as input and produces 100 feature maps of size 1 * 1 as the output. Finally, we have a SOFTMAX layer which produces 10 possible values corresponding to 10 digits ranging from 0-9. Model 2 achieves an accuracy of 99.05% and records a loss of 4.56%.

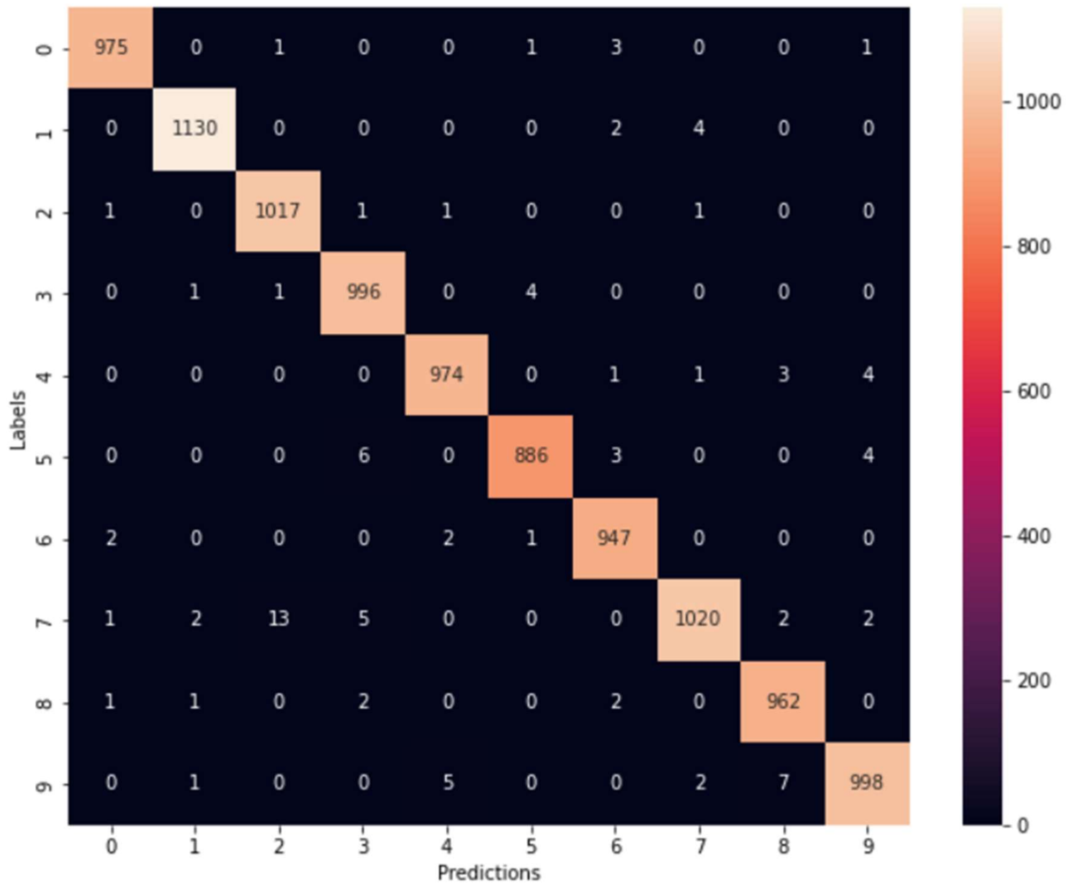
MODEL 2 Architecture

LAYER	LAYER TYPE	FILTER SIZE	NO OF FEATURE MAP	OUTPUT IMAGE SIZE	NO OF PARAMETERS TO TRAIN	ACTIVATION FUNCTION USED
INPUT	IMAGE		1	28 * 28		
C1	CONV	3 * 3	32	26 * 26	320	Relu
P2	POOL	2 * 2	32	13 * 13		
C3	CONV	3 * 3	64	11 * 11	18496	Relu
C4	CONV	3 * 3	64	9 * 9	36928	Relu
P5	POOL	2 * 2	64	4 * 4		
	FLATTEN			1024		
F6	FULLY CONNECTED		100	100	540900	Relu
OUTPUT	SOFTMAX		10	10	1010	Softmax

6. MODEL ACCURACY

Based on the above experiment that we have performed with the CNN by implementing 2 different CNN architectures, we have observed with adding more no of layers i.e. by increasing the depth we achieve better performance on MNIST data set. Hence we use the 2nd CNN Architecture and compute the Confusion Matrix and thereafter calculate the precision, recall and F1 score for all the classes i.e. digits ranging from 0 – 9.

CONFUSION MATRIX



PRECISION, RECALL and F1 Score

```
   Class  Precision    Recall    Fscore
0      0   0.993884  0.994898  0.994391
1      1   0.994718  0.995595  0.995156
2      2   0.996082  0.985465  0.990745
3      3   0.994012  0.986139  0.990060
4      4   0.990844  0.991853  0.991349
5      5   0.985539  0.993274  0.989391
6      6   0.994748  0.988518  0.991623
7      7   0.976077  0.992218  0.984081
8      8   0.993802  0.987680  0.990731
9      9   0.985192  0.989098  0.987141
```

7. REFERENCES

a. All lecture slides of chapter 1 -5 covered in class.

b. https://en.wikipedia.org/wiki/MNIST_database

c. https://en.wikipedia.org/wiki/LeNet

d. https://medium.com/geekculture/implementing-lenet-architecture-using-the-research-paper-d780acf02e1e

e. https://towardsdatascience.com/understanding-and-implementing-lenet-5-cnn-architecture-deep-learning-a2d531ebc342

f. http://yann.lecun.com/exdb/publis/pdf/lecun-01a.pdf