Handwritten Digit Classification using Machine Learning Models

Vidushi Garg

Student, Department of Information Technology, Maharaja Agrasen Institute of Technology, New Delhi, India

---

Abstract - *The rapid growth of new documents and multimedia news has created new challenges in pattern recognition and machine learning. The problem of handwritten digit recognition has long been an open problem in the field of pattern classification. Handwriting recognition is one of the compelling research works going on because every individual in this world has their own style of writing. It is the capability of the computer to identify and understand handwritten digits or characters automatically. The main objective of this paper is to provide efficient and reliable techniques for recognition of handwritten numerals by comparing various classification models. MNIST dataset is widely used for this recognition process and it has 70000 handwritten digits. This paper performs the analysis of accuracies and performance measures of algorithms Support Vector Classification model (SVC), Logistic Regression model and Random Forest Classification model.*

***Key Words*: Support Vector Classification Model(SVC), Modified National Institute of Standards and Technology database(MNIST), Binarization, Logistic Regression Model, Random Forest Classifier**

1. INTRODUCTION

Handwritten digit recognition is an important problem in optical character recognition, and it can be used as a test case for theories of pattern recognition and machine learning algorithms. To promote research of machine learning and pattern recognition, several standard databases have emerged. Handwriting recognition is one of the compelling and fascinating works because every individual in this world has their own style of writing. The main difficulty of handwritten numerals recognition is the serious variance in size, translation, stroke thickness, rotation and deformation of the numeral image because of handwritten digits are written by different users and their writing style is different from one user to another user.[2] In real-time applications like the conversion of handwritten information into digital format, postal code recognition, bank check processing, verification of signatures, this recognition is required.

This research aims to recognize the handwritten digits by using tools from Machine Learning to train the classifiers, so it produces a high recognition performance. The MNIST data set is widely used for this recognition process. The MNIST data set has 70,000 handwritten digits. Each image in this data set is represented as an array of 28x28. The array has 784 pixels having values ranging from 0 to 255.[6] if the pixel value is '0' it indicates that the background is black and if it is '1' the background is white.

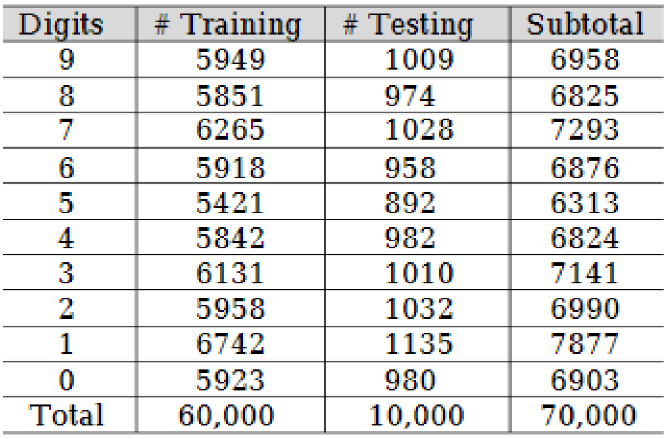
This study focuses on feature extraction and classification. The performance of a classifier can rely as much on the quality of the features as on the classifier itself. In this study, we compare the performance of three different machine learning classifiers for recognition of digits. The three classifiers namely Support Vector Classification model (SVC), Logistic Regression, and Random Forest Classification model. The main purpose of this research is to build a reliable method for the recognition of handwritten digit strings. The main contribution in this work is that Support Vector Classification model gives the highest accuracy while compared to the other classification models. Yet 100% accuracy is something that is to be achieved and the research is still actively going on in order to reduce the error rate. The accuracy and correctness are very crucial in handwritten digit recognition applications. Even 1% error may lead to inappropriate results in real-time applications.

2. RESEARCH METHODOLOGY

2.1 Description of the dataset

The MNIST dataset, a subset of a larger set NIST, is a database of 70,000 handwritten digits, divided into 60,000 training examples and 10,000 testing samples. The images in the MNIST dataset are present in form of an array consisting of 28x28 values representing an image along with their labels.[1] This is also the same in case of the testing images. The gray level values of each pixel are coded in this work in the [0,255] interval, using a 0 value for white pixels and 255 for black ones.

Table -1: Dataset



Digits	# Training	# Testing	Subtotal
9	5949	1009	6958
8	5851	974	6825
7	6265	1028	7293
6	5918	958	6876
5	5421	892	6313
4	5842	982	6824
3	6131	1010	7141
2	5958	1032	6990
1	6742	1135	7877
0	5923	980	6903
Total	60,000	10,000	70,000

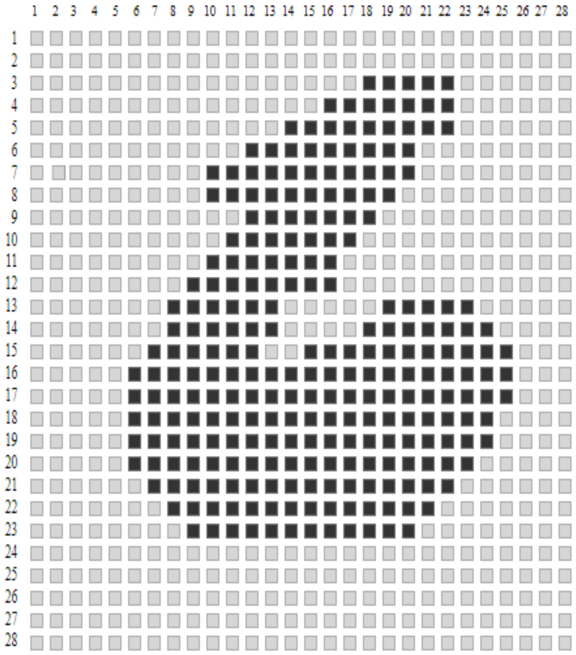


Fig -1: Digit 6 in 28x28 format in MNIST

2.2 Data Preprocessing

An important point for managing a high performance in the learning process is the construction of a useful training set. The 70000 different patterns contained in the MNIST database can be seen as a rather generous set, but evidence shows that the usual learning algorithms run into serious trouble for about one hundred or more of the test set samples. Therefore, some strategy is needed in order to increase the training set cardinality and variability. Usual actions comprise geometric transformations such as displacements, rotation, scaling and other distortions.

The variables proposed in this paper to make handwritten digit classification require images in binary level. The binarization process assumes that images contain two classes of pixel: the foreground (or white pixels, with maximum intensity, i.e., equal to 1) and the background (or black pixels with minimum intensity, i.e., equal to 0). The goal of the method is to classify all pixels with values above of the given threshold as white, and all other pixels as black. That is, given a threshold value t and an image X with pixels denoted as x(i, j), the binarized image X_b with elements x_b (i, j) is obtained as follows:

If x(i, j) > t $\quad x_b(i, j) = 1$

Else $\quad x_b(i, j) = 0$

Then, the key problem in the binarization is how to select the correct threshold, t, for a given image. We observe that the shape of any object in the image is sensitive to variations in the threshold value, and even more sensitive in the case of handwritten digit. Therefore, we consider that a binary handwritten number is better recognized computationally if its trace is complete and continuous, this is the criterion that we use to the threshold, being its choice of crucial importance.[3]

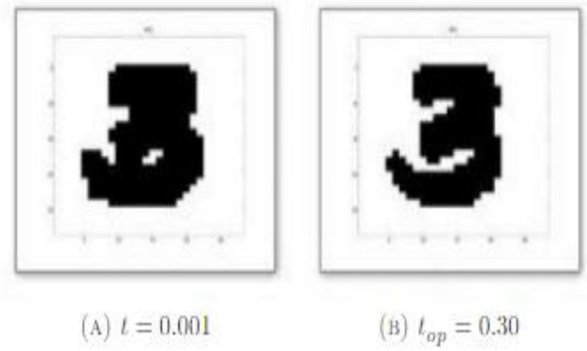

(A) $t = 0.001$ (B) $t_{op} = 0.30$

Fig -2: The same digit with different threshold value

2.3 Implementation

2.3.1 Logistic Regression Model

Logistic Regression is a Machine Learning classification algorithm that is used to predict the probability of a categorical dependent variable. In logistic regression, the dependent variable is a binary variable that contains data coded as 1 (yes, success, etc.) or 0 (no, failure, etc.). In other words, the logistic regression model predicts P(Y=1) as a function of X.

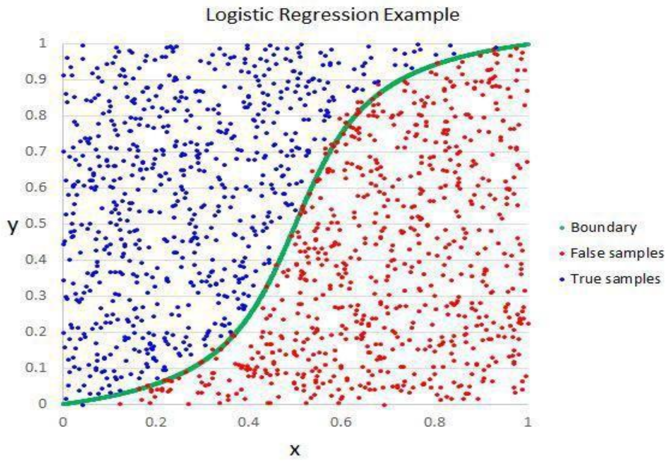


Fig -3: Logistic Regression Model

After that training set data is fed as input to the Logistic Regression model so that the classifier gets trained. The guessed label is matched with the original to get the accuracy of the trained classifier. Once the training is done, the testing data is given to the classifier to predict the labels and testing accuracy is obtained. The confusion matrix is generated which gives the probability between the actual data and the predicted data. Using the confusion matrix, the performance measures like precision, recall and f1 score can be calculated. Using Logistic Regression, an accuracy of 88.97% is obtained on test data.

Precision = TP/(TP+FP)

where TP = True Positive, FP = False Positive

Recall = TP/(TP+FN)

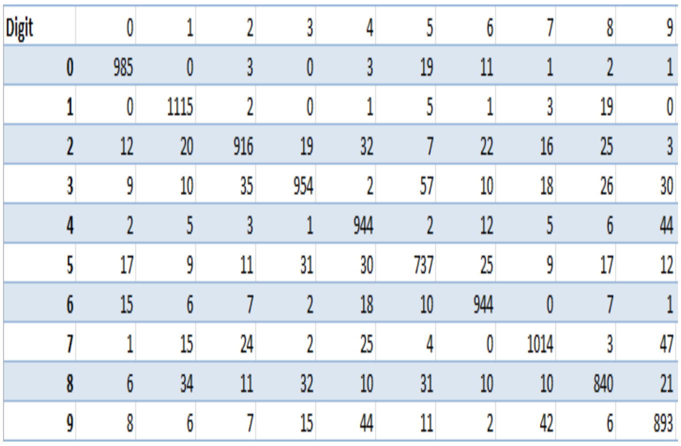
where FN = False Negative

F1score = 2*Precision+Recall/(Precission+Recall)

The confusion matrix obtained here is in the form of matrix M 10x10 because it is a multi-class classification (0-9).

The below Table 2 is the confusion matrix obtained for the trained data set using Logistic Regression Model.

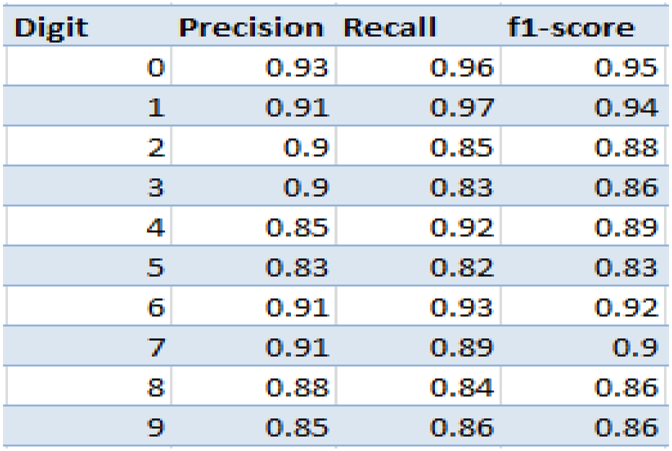
Table -2: Confusion matrix using Logistic Regression model



Digit	0	1	2	3	4	5	6	7	8	9
0	985	0	3	0	3	19	11	1	2	1
1	0	1115	2	0	1	5	1	3	19	0
2	12	20	916	19	32	7	22	16	25	3
3	9	10	35	954	2	57	10	18	26	30
4	2	5	3	1	944	2	12	5	6	44
5	17	9	11	31	30	737	25	9	17	12
6	15	6	7	2	18	10	944	0	7	1
7	1	15	24	2	25	4	0	1014	3	47
8	6	34	11	32	10	31	10	10	840	21
9	8	6	7	15	44	11	2	42	6	893

The below table 3 shows the precision, recall and f1 score values obtained for the trained data set using the Logistic Regression Model.

Table -3: Precision, Recall and F1 score for Logistic Regression on trained dataset



Digit	Precision	Recall	f1-score
0	0.93	0.96	0.95
1	0.91	0.97	0.94
2	0.9	0.85	0.88
3	0.9	0.83	0.86
4	0.85	0.92	0.89
5	0.83	0.82	0.83
6	0.91	0.93	0.92
7	0.91	0.89	0.9
8	0.88	0.84	0.86
9	0.85	0.86	0.86

The Test Data Set obtained accuracy of 88.97% using the Logistic Regression on MNIST data set.

2.3.2 Random Forest Classifier

Random forest is a supervised learning algorithm. It can be used both for classification and regression. It is also the most flexible and easy to use algorithm. A forest is comprised of trees. It is said that the more trees it has, the more robust a forest is. Random forests creates decision trees on randomly selected data samples, gets prediction from each tree and selects the best solution by means of voting. It also provides a pretty good indicator of the feature importance.

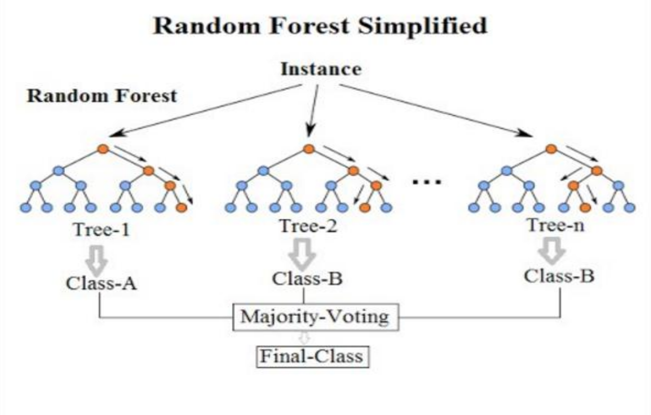
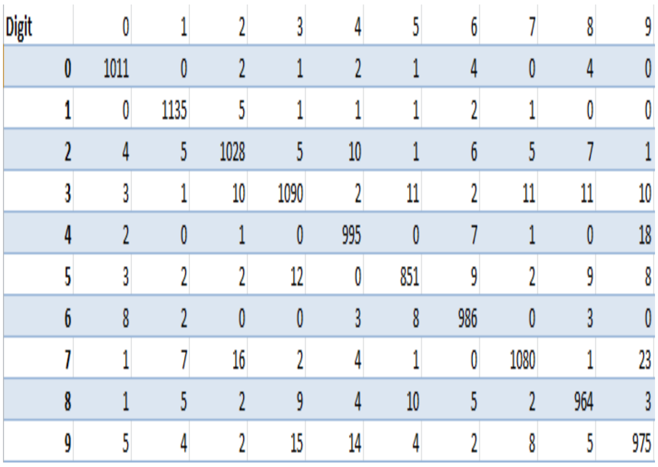


Fig -4: Random Forest Classifier

Here also, the confusion matrix is obtained and precision and recall values are computed.

Table 4: Confusion matrix using Random Forest Classifier

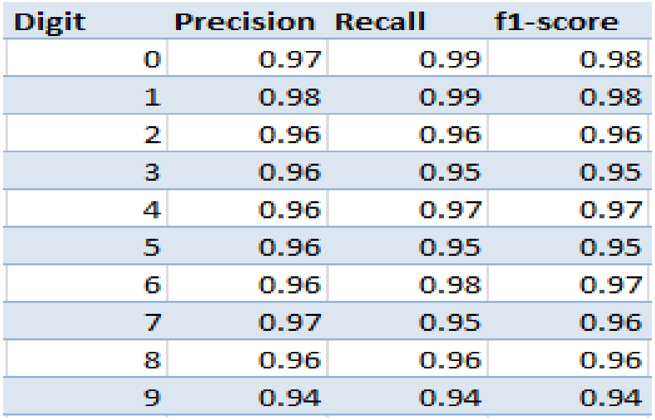


Digit	0	1	2	3	4	5	6	7	8	9
0	1011	0	2	1	2	1	4	0	4	0
1	0	1135	5	1	1	1	2	1	0	0
2	4	5	1028	5	10	1	6	5	7	1
3	3	1	10	1090	2	11	2	11	11	10
4	2	0	1	0	995	0	7	1	0	18
5	3	2	2	12	0	851	9	2	9	8
6	8	2	0	0	3	8	986	0	3	0
7	1	7	16	2	4	1	0	1080	1	23
8	1	5	2	9	4	10	5	2	964	3
9	5	4	2	15	14	4	2	8	5	975

The above Table 4 shows the confusion matrix obtained for the trained data set using the Random Forest Classifier.

The below table 5 shows the precision, recall and f1 score values obtained for the trained data set using the Random Forest Classifier.

Table 5: Precision, Recall and F1 score for Random Forest Classifier on trained dataset



Digit	Precision	Recall	f1-score
0	0.97	0.99	0.98
1	0.98	0.99	0.98
2	0.96	0.96	0.96
3	0.96	0.95	0.95
4	0.96	0.97	0.97
5	0.96	0.95	0.95
6	0.96	0.98	0.97
7	0.97	0.95	0.96
8	0.96	0.96	0.96
9	0.94	0.94	0.94

The Test Data Set obtained accuracy of 96.3% using the Random Forest Classifier on MNIST data set.

2.3.3 Support Vector Machine (SVM) Classifier for digit recognition

Support Vector Machine is a supervised machine learning technique which is applied for classification and regression. It is nothing but the representation of the input data into points in the space and mapped thus classifying them into classes. The SVM classifies or separates the classes using the hyper-plane concept.[9] The separation margin should be equidistant from the classes.

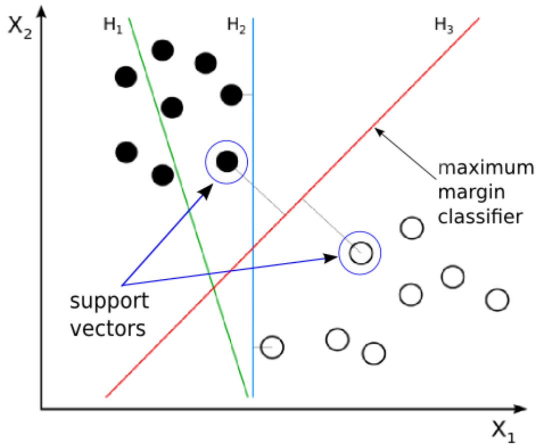
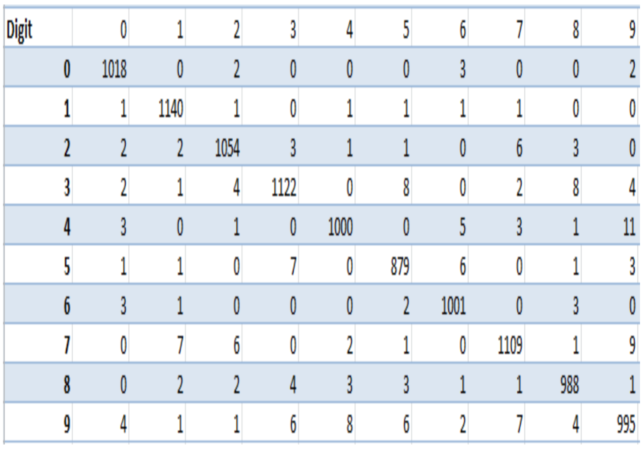


Fig-5: Support Vector Machine Classifier

Here also, the confusion matrix is obtained and precision and recall values are computed.

Table 6: Confusion matrix using Support Vector Machine Classifier

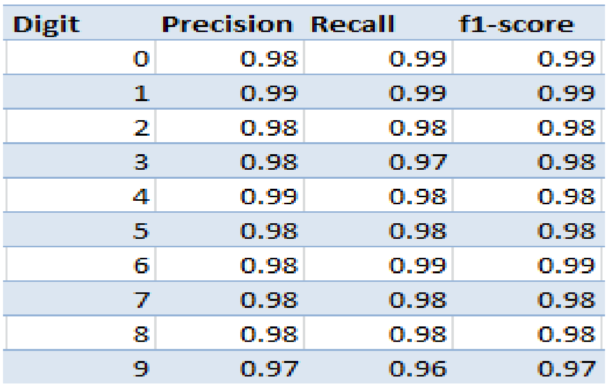


Digit	0	1	2	3	4	5	6	7	8	9
0	1018	0	2	0	0	0	3	0	0	2
1	1	1140	1	0	1	1	1	1	0	0
2	2	2	1054	3	1	1	0	6	3	0
3	2	1	4	1122	0	8	0	2	8	4
4	3	0	1	0	1000	0	5	3	1	11
5	1	1	0	7	0	879	6	0	1	3
6	3	1	0	0	0	2	1001	0	3	0
7	0	7	6	0	2	1	0	1109	1	9
8	0	2	2	4	3	3	1	1	988	1
9	4	1	1	6	8	6	2	7	4	995

The above Table 6 shows the confusion matrix obtained for the trained data set using the Support Vector Machine Classifier.

The below table 7 shows the precision, recall and f1 score values obtained for the trained data set using the Support Vector Machine Classifier.

Table 7: Precision, Recall and F1 score for Support Vector Machine Classifier on trained dataset



Digit	Precision	Recall	f1-score
0	0.98	0.99	0.99
1	0.99	0.99	0.99
2	0.98	0.98	0.98
3	0.98	0.97	0.98
4	0.99	0.98	0.98
5	0.98	0.98	0.98
6	0.98	0.99	0.99
7	0.98	0.98	0.98
8	0.98	0.98	0.98
9	0.97	0.96	0.97

The Test Data Set obtained accuracy of 98.3% using the Random Forest Classifier on MNIST data set.

3. CONCLUSIONS

The accuracies of the algorithms Logistic Regression, Random Forest Classifier and Support Vector Machine Classifier are tabulated below in table 8.

Table 8: Data Accuracy of different models

Algorithm	Data Accuracy
Logistic Regression	88.97%
Random Forest Classifier	96.3%
Support Vector Machine Classifier	98.3%

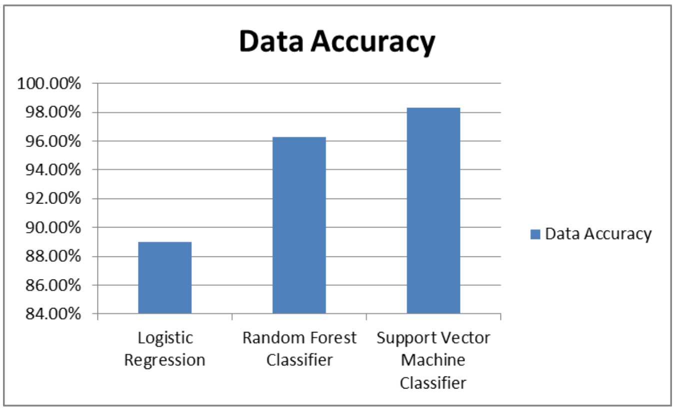


Chart -1: Data accuracy chart of different models

It can be clearly observed that Support Vector Machine Classifier has more accuracy compared to Logistic Regression and Random Forest Classifier.

REFERENCES

[1] Wu, Ming & Zhang, Zhen. (2019)paper on "Handwritten Digit Classification using the MNIST Data Set"

[2] S M Shamim, Mohammad Badrul Alam Miah, Angona Sarker, Masud Rana & Abdullah Al Jobair paper on "Handwritten Digit Recognition using Machine Learning Algorithms"

[3] Andrea Giuliodori, Rosa Lillo and Daniel Peña paper on "HANDWRITTEN DIGIT CLASSIFICATION"

[4] Plamondon, R., & Srihari, S. N. Online and off-line handwriting recognition: a comprehensive survey. IEEE Transactions on Pattern Analysis and Machine Intelligence, 22(1), 63-84

[5] Liu, C. L., Yin, F., Wang, D. H., & Wang, Q. F. (2013). Online and offline handwritten Chinese character recognition benchmarking on new databases. Pattern Recognition, 46(1), 155-162

[6] Handwritten Digit Recognition using Convolutional Neural Networks in Python with Keras

[7] Al-Wzwazy, Haider& M Albehadili, Hayder&Alwan, Younes& Islam, Naz& E Student, M &, Usa. (2016). "Handwritten Digit Recognition Using Convolutional Neural Networks. International Journal of Innovative Research in Computer and Communication Engineering"

[8] AL-Mansoori, Saeed. (2015). "Intelligent Handwritten Digit Recognition using Artificial Neural Network"

[9] Meer Zohra, D.Rajeswara Rao paper on "A Comprehensive Data Analysis on Handwritten Digit Recognition using Machine Learning" Approach by. International Journal of Innovative Technology and Exploring Engineering (IJITEE) ISSN: 2278-3075, Volume-8 Issue-6, April 2019