HANDWRITING DETECTION USING NEURAL NETWORKS

Project report submitted in partial fulfillment of the requirement for the degree of

Bachelor of Technology
in

Information Technology

By

Rishabh Tewari (161460)

Under the supervision

of

Ms Monika Bharti

to

Department of Computer Science & Engineering and Information Technology

Jaypee University of Information Technology Waknaghat, Solan173234, Himachal Prades
h

Department of Computer Science & Engineering and Information Technology

Jaypee University of Information Technology Waknaghat, Solan-173234,Himachal Pradesh

Certificate

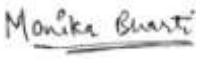
Candidate's Declaration

I hereby declare that the work presented in this report entitled "**Handwriting Detection Using Neural Networks**" in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Information Technology** submitted in the department of Computer Science & Engineering and Information Technology,Jaypee University of Information Technology, Waknaghat is an authentic record of my own work carried out over a period from July 2019 to May 2020 under the supervision of Ms. Monika Bharti .

The matter embodied in the report has not been submitted for the award of any other degree or diploma.

Rishabh Tewari(161460)

This is to certify that the above statement made by the candidate is true to the best of my knowledge.



Ms. Monika Bharti
(Assistant Professor-Grade II)

ACKNOWLEDGEMENT

I take this opportunity to express my profound gratitude and deep regards to my project guide Ms. Monika Bharti for her exemplary guidance , monitoring and constant encouragement throughout the course of this project .The blessing ,help and guidance given by her ,time to time shall carry me a long way in the journey of life on which I am about to embark. I am also obliged to staff members of JUIT college , for the valuable information provided by them in their respective fields. I am grateful for their cooperation during the period of my assignment.

Lastly, I thank almighty, my parents ,my classmates for their constant encouragement without which this project would not have been possible.

TABLE OF CONTENTS

LIST OF FIGURES

Random forest

LIST OF TABLES

ABSTRACT

This submitted report draws a comparison between 2 separate models ,to detect the hand written text while making use of:-
"MNIST"EMNIST" datasets. The upshot will be a system-structure which almost accurately inform about :

i) Given digit/s ii)

Given alphabet/s

The "2 models" in which we make comparision:

*"Artificial Neural Network Model"

* "RandomForest Model"

Also done planning to additionally experiment with some different hyperparameters in order to max. the test_accuracies & also lower "overfittingness" as attainable.

"EMNIST"/"MNIST" are the most common datasets available on the internet which can beused for hand writing analysis & detection.The MNIST datasets contains about "60000 training-imgs & 10000 test images". The EMNIST datasets have about "124800 training images &20800 test images".

“CHAPTER-1”

“INTRO-DUCTION”

#1.1”Introduction to Handwriting Detection”

“Handwriting detection” techniques can be identified as the capability of a computer/s to take input and co mputing the handwritten text from the sources such as touchscreens, photo’s , papersheet/s, and other similar device . The OPTICAL CHARACTER RECOGNITION(OCR) principles can be put into use while sensing the image of handwritten text from the paper as ‘offline’. Otherwise, for instance, take a pen/pencil based “comp. screen surface” is perfect example of motion s of pen-tip which can be sensed over as ‘online’. It is usually easier work to do as there are more clues for us.

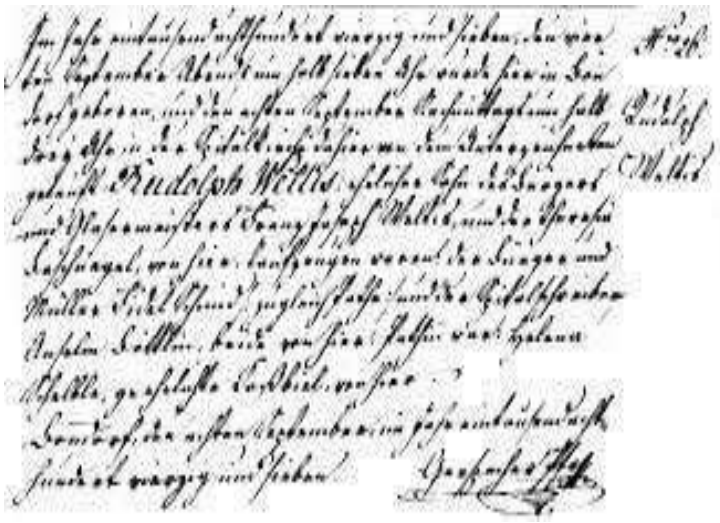
The facility to store the hand-written content digitally, and also converting the digitalizedsignatures which computer can easily interp ret, etc. are amongst the uses of it .

Here only the work to classify the text utilizing “artificial neural networks” is done. The large databases o f handwriting digit/numbers such as “MNIST” are used for the task. Handling of the task such as to form at,segment it to characters is also done here. The perfectly suited word is searched.

Fig.#1.1” Image_of_ handwriting- detection”

#1.1.1"Examples of handwritten text"

Below given some "handwritten texts" samples. Surely, exists a number of applications/service grown using mentioned tech.



(#a)

(#b)

(#c)

(#d)

Fig#1.1.1"Handwritten-text img" (#a) Img-1 (#b) Img-2 (#c) Img-3 (#d) Img-4 **#1..1.2"Use-case :*handwriting-detection In Postal-Service"**

The "Postal Service Dept." of the USA introduced a software sys. named "Hand written address interpretation" which sorts I and all about ninty-five percent written (handwritten) mails automatically. Mentioned Use_case explains tech-working.

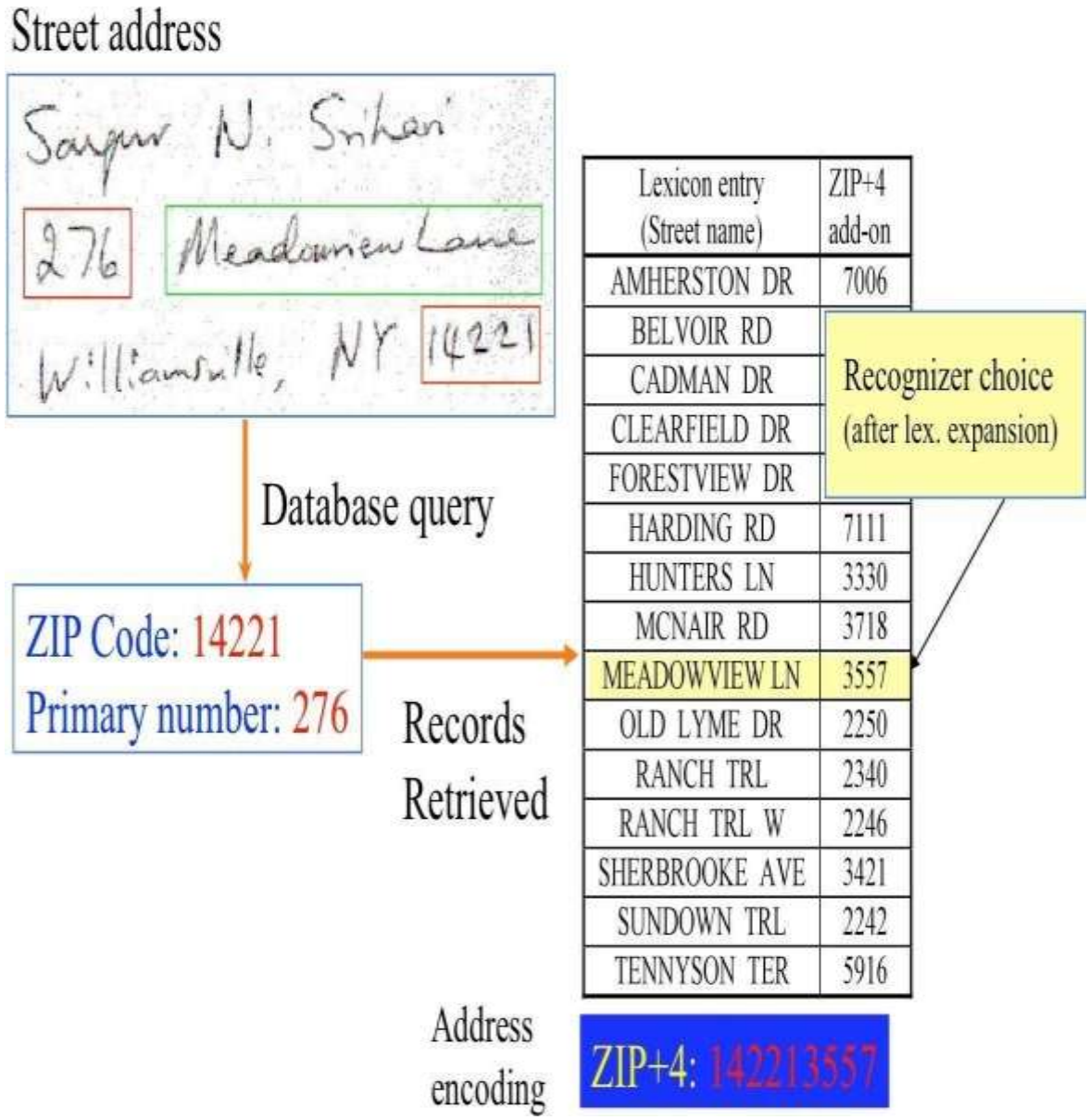


Fig..#1.1.2"Use-case :Handwriting-detection postal—serv." **#1..1.3"Detecting the hand-writing in "AUTHOR IDENTIFICATION-SYSTEM"Use Case"**

We can easily recognize the writer of the writtentexts(i.e. author) by just examining the presented handwritten-texts.

The above also goes by the name "handwritten biometric recognition".

#1..2Steps:"handwriting-detection"

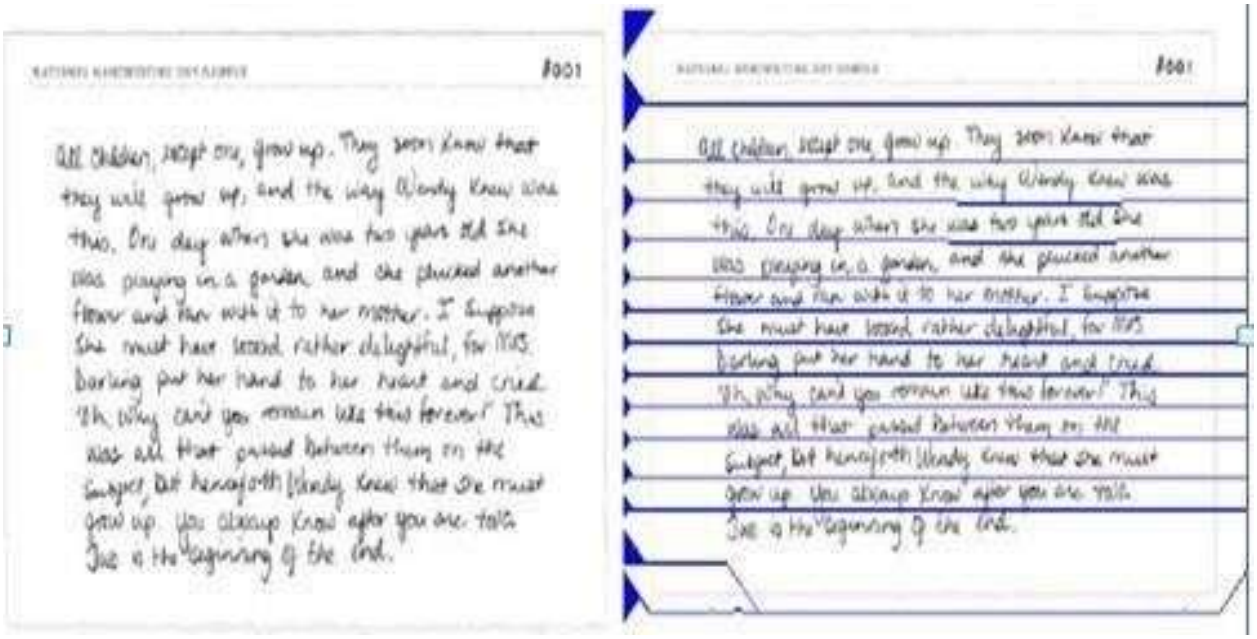
*1)."Line--Segmentation"

*2)."Character--Segmentation"

*3)."Pre-process the image in-order to pull out noise"

*4)."Try fitting methodology to enact Handwriting_detection"

#1.2..1"Line--segmentation"



(#a) (#b)

Img#1..2.1)Line_Segmentation (#a)) Org.-Image (#b))Result line_segmentation img.

The process of separation of lines in the given text, is "line-

segmentation". Lets assume the given lines are ten in the handwrittentext. Foremost step is the, labelling

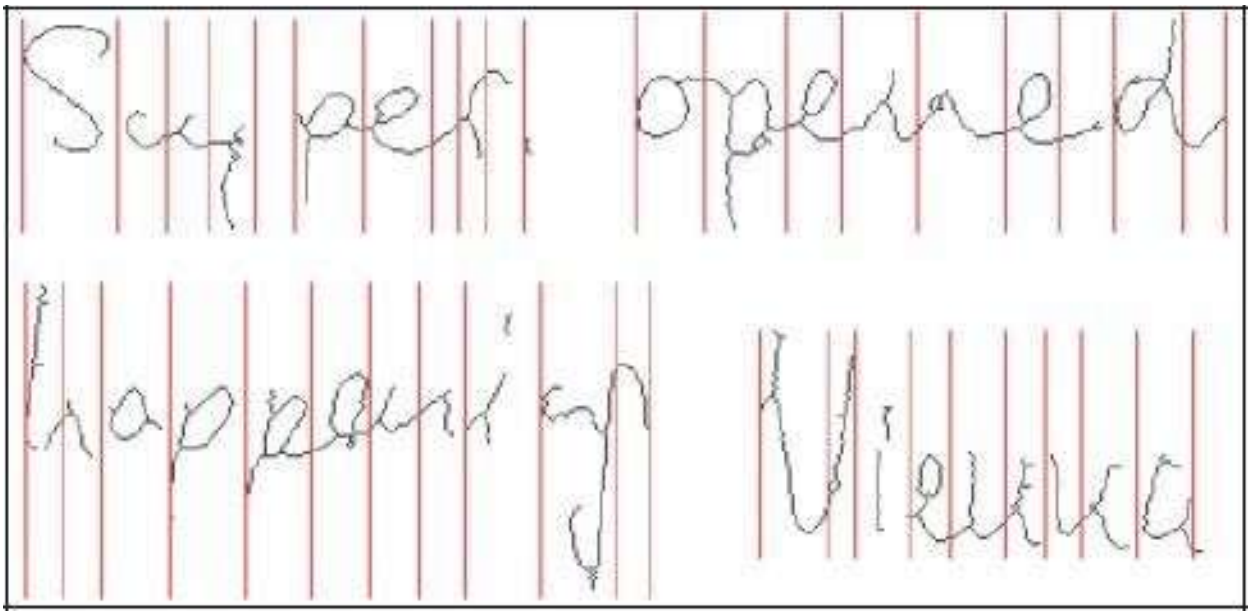
every obtained line. The above step is of great importance as characters in hand writing texts

be differentiated prior use-of the “recognition techniques”.

As distict person owns distinct ways/style of hand-
writing. Some posses angledwriting of words and gap inbetween the line/s is unequal, creating diff iculties in linesegmentation.

“Edge Detection Techniques” is utilized here. Every “edge detection technique” can be change d to do “line segmentation” as per need.

#1.2.2)”CharacterSegmentation”



Fig#(1.2..2)”Character-Segmentation”

Now as the“linesegmentation” completes we’ll plan to split character from character in this line. The attempt to
disintegrate the image containing strings of character to smaller parts is “character segmentation”.

Size of texts differ from large to small, with character spacing very much to less. All based on persons handwriting”. Template Matching” helps in obtaining unique property of characterimage.

The result of this project will be as suppose, we possess a character and then we understand, then we gain knowledge of it and then we find what character it is & which letter possess it. Thats one of the segment in “handwriting-detection”.

#1..2.2)”The preprocessing technique/s” in order to remove noise

Now lets get to “pre-processing tech.” which we’ll perform on hnd writing text.

Firstly we’ll remove the noise using ’noise removal tech.’

.

Noise occur in various forms ,as viewable in white-
black dotted form on images,known as salt pepper noises.
It occurs because of clicking noise,particles of dust,etc

.

(#b)) (#a)

Fig.#1..2.3)Median filtering (#a)Result-Image(#b)Original-Image

In order to control and lessen these sort of noises, “median filtering” is used

.

Its working is done by selecting a set no. of “pixels”, further alloting it’s val. to the nearby pi xel/s.

Say instance : -

Lets set a no. of pixels in a group “y”

y =[3 90 7 4]

So, the median filtered output signal “x” will be :- x[1] = Median[3 3 90] = 3

x[2] = Median[3 90 7] = Median[3 7 90] = 7

x[3] = Median[90 7 4] = Median[4 7 90] = 7

x[4] = Median[7 4 4] = Median[4 4 7] = 4

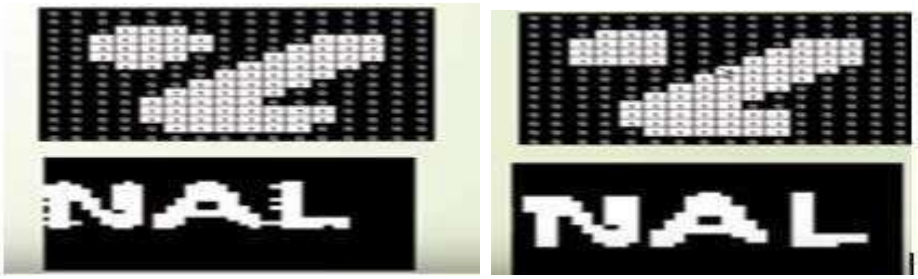
thus , x=[3 7 7 4]

The output signal of 90 changes to 7 & by this white-pixels on the image gets removed.

#1.2..3.1)“The Morphological operation”

Next action which could take execution, “morphological operation” in order cancel noises. C larify & smoothening by excluding tiny “protusians” in the character is done.

It helps in distinctly identify character from the other character.



(#b) (#a)

#Fig.1.2..3.1)”Morphological-Operation”(#a) Result-Img (#b)Actual-Img

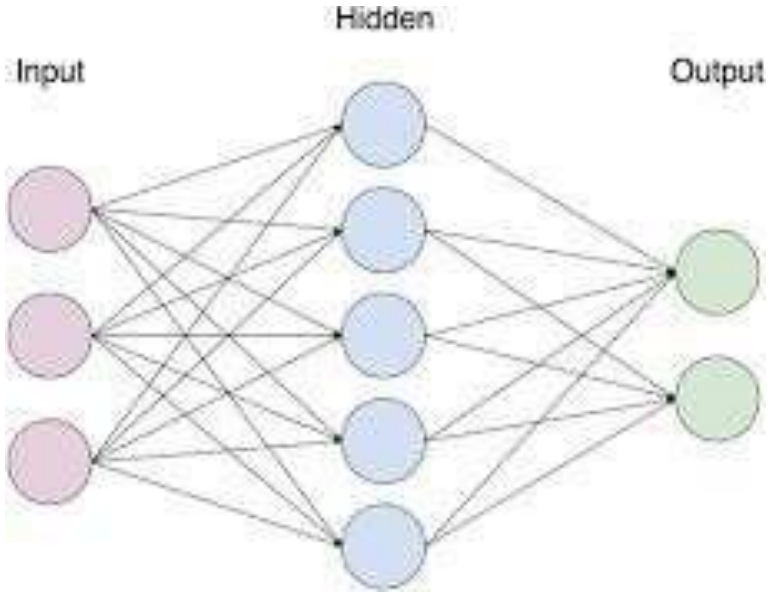
#1..3)”An Intro to the Neural-network”

Interconnecting group of nodes can be referenced as “neural networks”.
Similar to the connected-network of neurons within central nervous system.

Let artificial neuron within the nervous system be represented by distinct ovalnode & to connect from o ne o/p neuron to i/p lets assume arrows.

“The 2 basic parts is incorporated by NNs” of central nervous system nets:

(i)The nodes in NN be Neurons in central nervous system.
(ii)The weighs in NN be ‘Synapses’ in central nervous system.



#Fig.1..3)”Neural-networks”

It is inspired by biological neural networks within the brain and works in similar manner. It's like having multiple signal to a node and it's a choice creating capability and the outputs may be one or multiple again and this output relies on the previous logic and weight that's assigned. We can always assign weight to the nodes and at the node assigned perform some kind of calculation and the output of this could be a calculated weighted value of the set of inputs. Therefore in this way using calculations we are able to expect it to do a logic for pattern recognition.

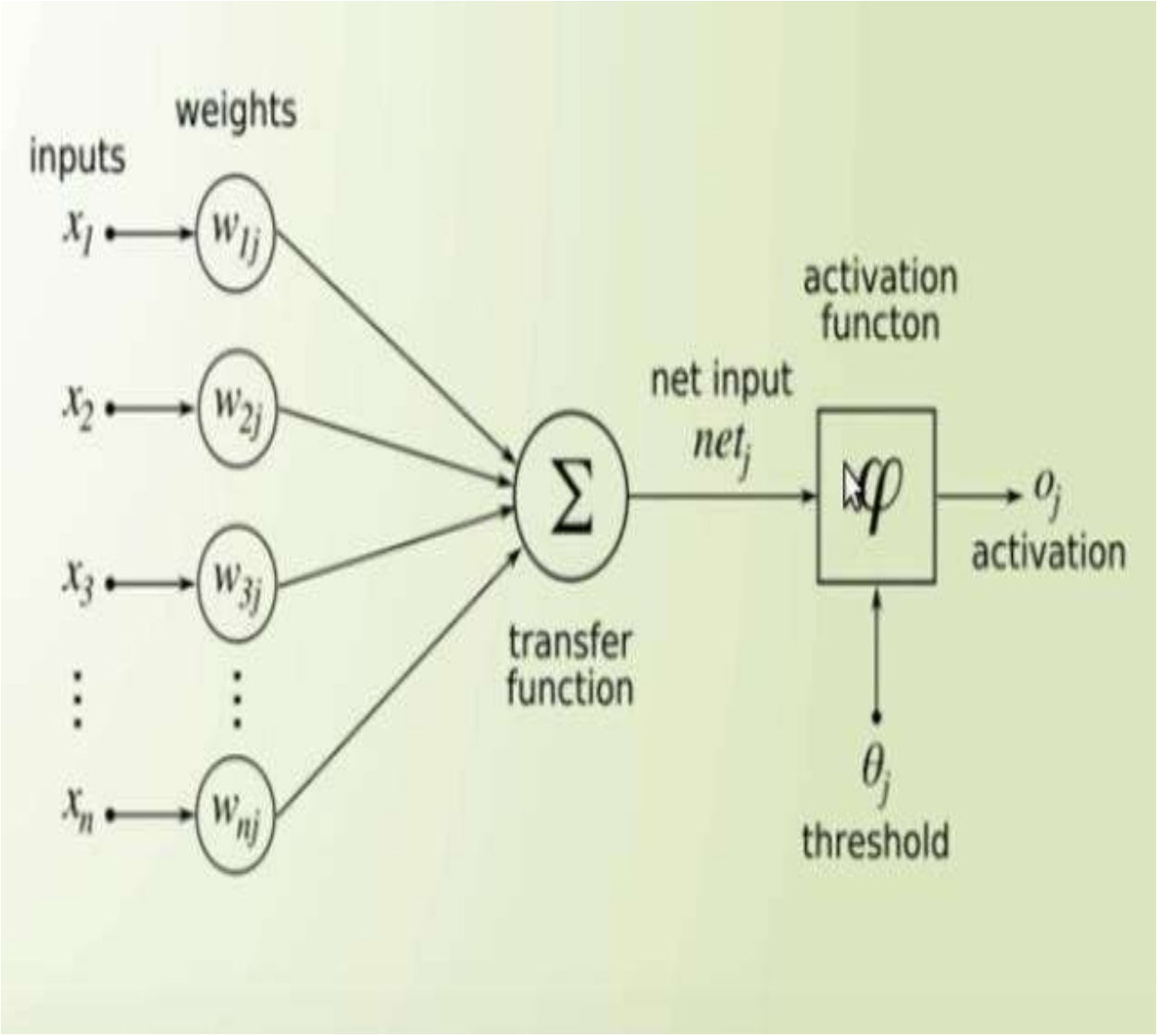
Neural network work in three main paradigms :-

1. Supervised learning :- It is where we know the cost function , input data and all the data i.e. given ,so we can calculate the cost function based on input data i.e. given. Cost function is basically the difference between the input and output data. It is basically a feedback network where we have the output, then you are taking part of output and feeding it the input and thus we calculate the cost function.

2. Un-supervised learning :- Here the cost function may or may not be the function of the input data.

3. Reinforcement learning :- How your input interacts with environment is known but you actually don't know what your input is. Only thing we know is how it behaves with the environment, what is the result ,and what are the effects of it. So, in this learning you can estimate a cost function but you might not be able to calculate cost function.

#1.3..1)"Basic Neural -- Networks"



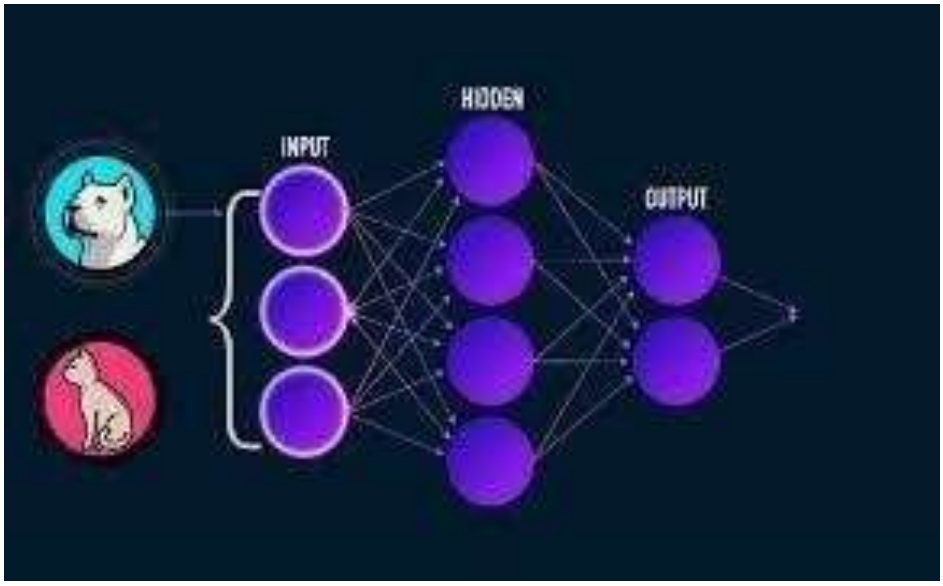
#Fig. 1..3.1)"Basic Neural-networks"

This is the
weights ass
combines a
or covertin

.

#1.3..2)"Neural-network: 'A Classifier-systems'"

Artificial neural networks are very effective classifiers. For example, we can train an ANN to classify cats versus dogs. In this example, cats and dogs are our 2 classes. It would be a binary classification problem.



#Fig.)1.3..2"Neural-networks:classifier-system"

#1.3..3)"Why deploy the Neural-Networks?"

Neural network with its ability can be used to derive or determine the very meaning of complicated data that can be use to detect trend and also can be used to extract pattern that are too difficult for human to be notice or some other computer techniques. A trained neural network can be thought of as an genius or as an expert in the category of information it has been given to analyze. This trained neural network can be used to provide projections given new situations of interest and answer what if questions. Other Advantages Include:

1. Adaptive Learning : It is an ability to learn how to do tasks based on the data given for experience in beginnin g or training .

2. Real Time Operation :- ANN computations may be carried out in parallel, and special hardware devices are being designed and manufactured which is of this capability.

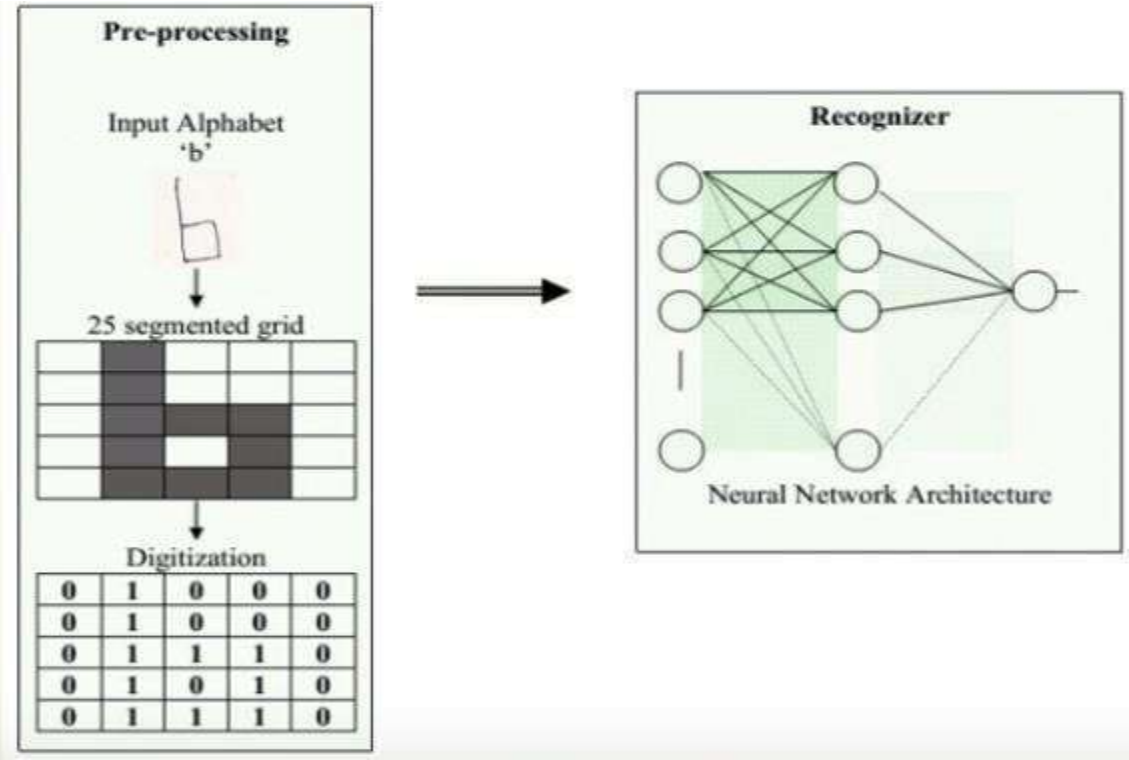
3. Self-Organization :- An ANN can create its own representation or organization of the information which receives during the learning time.

4. Fault Tolerance :- Partial destruction of network can leads to corresponding degradation of the performance. However, here some network capabilities can be retained even during major network damages.

1.3.4 Applications of Neural Network

Neural networks is where you have a set of samples of a particular alphabet 'a' and we suppose to take at least 5-10 different samples of 'a' i.e. 10 different handwritings of 'a'. So we take 10 different samples and we perform neural network , and we write a neural network algorithm and we run the neural network algorithm on this 10 images and then when we have a 11^{th} image or something we know whether it is an 'a' another alphabet.

#1..3.5)"2-phase character/s recogn."



#Fig.1..3.5)'2-phase character/s-recogn.'

These are the two phases of character recognition. The preprocessing technique as I said and you are separating the alphabet and you are doing preprocessing technique on it. This is one of the ways in which your image will be converted in a typical scenario in a MATLAB or opens a way whichever the software you are using. So this is represented in form of grids, and this is in digitized form. As shown in the above figure the grids which are filled with the alphabet are written as 1 in digitized form and the rest are written as 0. This is in fact fed to the neural network where all the processing is done. Neural network architecture has basically three layers

- Input, Hidden and Output layer.

Hidden Layer could be any number of hidden layer nodes the more the number of hidden layer nodes and hidden layer, the more the complex the neural network is, the more computational requirement is necessary for this system that you are using. It all depends on how we go about

balancing, how many nodes you need for the hidden layers.

#1.3..6)'Multi-layered Perceptrons[M.L.P] clasifier'

The next technique we use here for character recognition is MLP classifier. This is a s ort of classifier that may be used alternative to a neural network. Neural network might get computationally intensive based on kind of calculation and also the complexity of t he problem need to increase the quantity of hidden layers simply to attain higher accuracy. Ther efore, in order to do that we go on increasing the hidden layers or the quantity of hidden nodes. These are number of hidden nodes in figure. we might want to have another layer like this, we m ight have 3-
4 hidden layer based on complexity of the program. Complexity is to do a character recognition for all English alphabets. We've twenty-
six different alphabets and to do this twentysix different alphabets, neural network has got to be complex. We have 26 different alphabets a
nd to do this 26 different alphabets, neural network has to be complex. We can have vari ous different sizes, different styles of handwriting. "A" simply could be written in thous ands of styles and to understand all these things, we are working to store and to see how the p attern is changing between the various ways. We might want to possess lot of this network. Ther efore this cannot can be decided. This is the limitation of Neural Network where your network si ze as to increasing. You could be guaranteed of getting good results or you may not but the thin g is, if you are increasing the size of network your calculations are increasing. When once t he calculations are increasing, your computational capacity need to be increased or you might not be even able to run it on home PC to just test how it works. So that is where multilayered p erceptron classifier comes in.

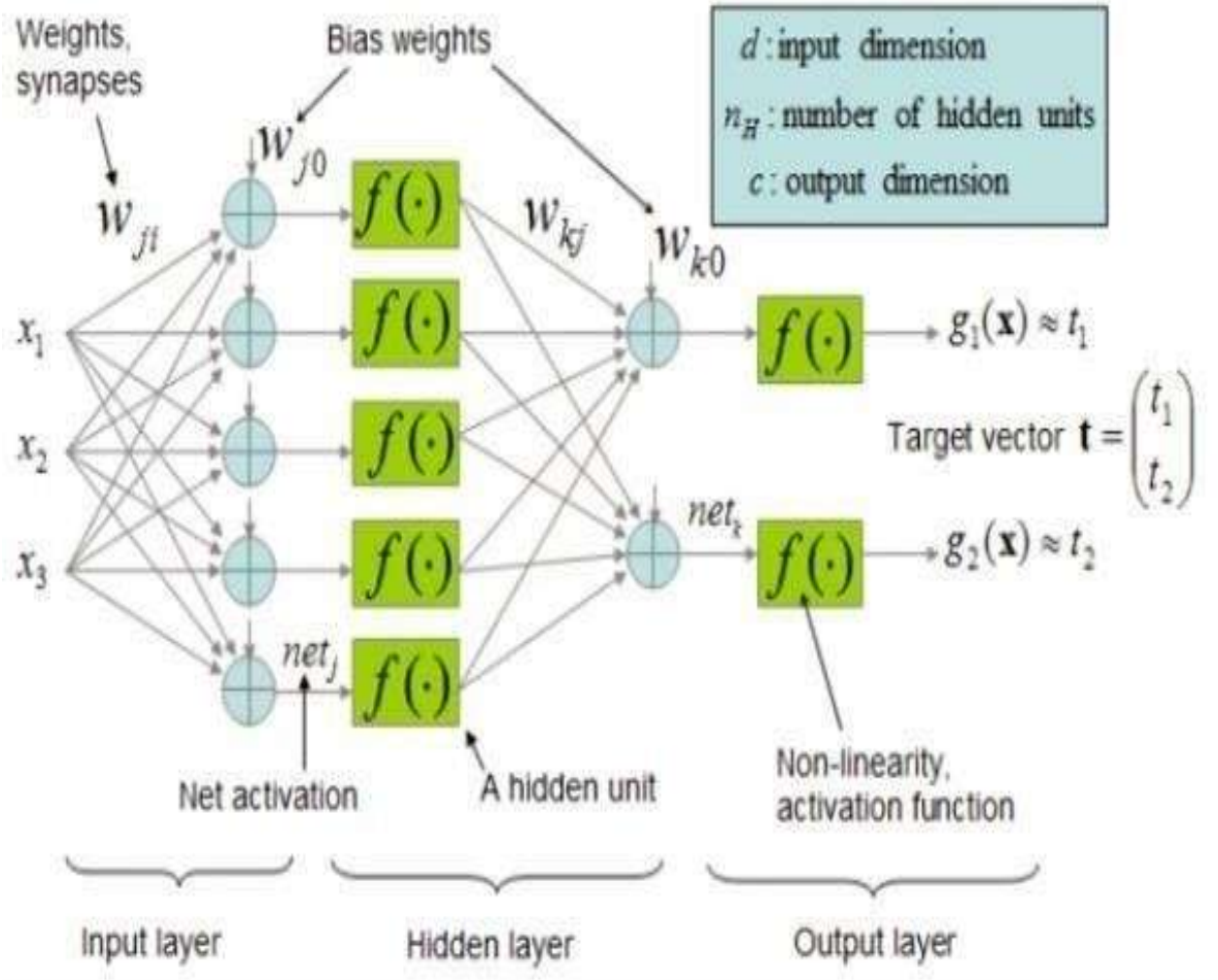
1. It is a feed forward neural network :- Feed forward in the sense you do not have anything that is coming back from the output you are not using the output to basically control the input over here which is the not the case in neural network

2. Data is propagated from input to output units in a feed forward way :- i.e there is no feed back form the output

3. Data processing may extend over multiple layers :- So there can be multiple MLP classifier that can be used but there can be only one hidden layer

4. A one hidden layer MLP can approximate any function to any desired accuracy:- This is the advantage of MLP classifier ,we can use any number of MLP classifiers we can all process it in parallel but at the same time we are restricting the hidden layer to one hidden layer which reduces the competition calculations to a great extent and we can also be sure of getting any disaccurate results.

5. Because of these advantages MLP classifiers can avoid the bulk of calculations performed in conventional neural networks . As we demonstrated before in the above point.



#Fig.)1..3.6"Multiple-layered Perceptrons clasifier"

So this is a typical diagram of a multilayers MLP classifier. This Wji represents weights synapses. Weight synapses refer to the strength or amplitude of connection between the nodes how strongly X1 and next node is connected. We have biased weight of each node and the function – The hidden unit. Net activation is the transfer function in the hidden unit. We have final activation function.

It is basically similar to neural network but only difference we do not have a feedback and we can have many multiple classifiers of hidden unit running parallel. So it is a feed forward network and it is much faster because we just have one hidden layer.

1.4 Problem Statement

Currently handwriting detection is done using the Optical character recognition (OCR) approach Optical character recognition (OCR) is a technology that enables you to convert different types of documents, such as scanned paper documents, PDF files or images captured by digital camera editable and searchable data.

But Optical character recognition is not a complete success. Because Optical character recognition technology is not always perfect. In Optical character recognition hundred percent accuracy rate does not exist. The Optical character recognition engine makes use of pattern recognition and complex algorithms of mathematical nature to give a good text file output. We can optimize the settings of an Optical character recognition engine to help us but this may put us in many more problems as the solving of one problem can lead to the creation of another problem and many other such disadvantages are linked with Optical character recognition.

#1..4.1Solution :An Answer'

The above problem can be solved by applying New feature extraction techniques and by using a rtificial neural network. By applying and using above techniques we are able to expect hundred percent accuracy of character recognition and may sort out all kind of problems that occurs due t o the Optical character recognition. We have done compared Artificial neural network, ra ndom forest, XGBoost to see which techniques perform better.

#1..5)'Objectives'

1. To Understand problem of handwriting detection.

2. We aim to covert old literature into digitized form.

3. System serve or act as guide and working in character recognition areas.

4. To describe solution using Artificial neural network, Random forest, XGBoost and compar e the results among them to see which algorithm performs better.

5. Applying the solution to MNIST and EMNIST dataset.

6. To demonstrate various techniques that are used to perform handwriting detection.

7. Emphasize on the limitation of various techniques.

CHAPTER 2

LITERATURE SURVEY

In this chapter, we have looked into different research papers that have investigated the different studies regarding handwriting detection using Neural Network.

Gil Levi and Tal Hassner, Haider A. Alwzwazy1 [1,2] Neural networks are playing a major role in machine learning applications, since neural networks work like human brain they try to mimic the working of neurons in human brain that is why they are nowadays used to build self-driven cars that can park themselves without any driver. Google deep minds are also working in this field to develop board playing games.

Dr. Kusum Gupta [3] In machine learning applications, we need different features for the prediction. So the feature recognition helps us in extracting features and their parameters. If we have more features, then it will sometimes lead to over fitting. When there is large difference between train accuracy and test accuracy, then model will over fit that is why feature specification is very important. We have to select those feature only which will help us in better prediction in the application.

Faisal Tehseen Shah, Kamran Yousaf [4] Before 1980's and 1990's the data sets are very less and the neural networks are fringe but in 21st century the data sets are present in large amount that is why neural networks are making a big come back and neural networks helps us to make large number of machine learning applications. Later in 1980's developing of speech recognition was a very difficult task but now this is not a very difficult task because of the increase in the data set.

Sherif Abdel Azeem, Maha El Meseery, HanyAhmed [5] Neural networks can be used as powerful tool for signal and image processing. In this research paper the online Arabic handwritten digits' recognition is done. The neural networks have lot more potential to do things by increasing the number of hidden layer in the network, all the processing is done by the hidden layer nodes.

Yusuf Perwez, Ashish Chaturvedi [6] To recognize the alphabets we can represent the English alphabets in the binary format in which the input is given to the simple feature extraction system and the output is fed the neural network that we have created.

Gregory Cohen, Saeed Afshar, Jonathan Tapson [7] In this research paper MNIST is introduced which is a dataset which is extended version of the NIST. This dataset contains about 60k images and the images are only the black and white images and because of its large data set this dataset help us to train our algorithm more accurately. MNIST also solves the problem of data set for the neural networks.

Diederik Kingma, Jimmy Ba [8] Adam Optimizer is a variant of stochastic gradient descent algorithm. Adam is an abbreviation for adaptive moment estimation. It is the current best choice among gradient based convex optimization algorithm. One advantage is that hyperparameters require less tuning. This method is very efficient, requires very little memory requirements and is best for problems which are having large data or parameters.

C. Zhang & P. C. Woodland [9] The creation of the hidden layers is always an issue and the function that can be used in the hidden layer is also an issue, the most common functions are the sigmoid function and ReLU function. In this paper we have studied about the various parameters of the ReLu function that has also helped in our project.

X. Glorot and Y. Bengio [10] In this research paper we have studied that there are some problems in finding the local minima of the algorithm and there can be problem of over fitting and under fitting, the training of deep neural networks is also a major issue for this we have to do random initialization.

Chapter-#3

'SYSTEMS-DEVELOPMENTS'

#3..1)'MNIST—Dataset'

1. MNIST is a database of handwritten digits.

2. It has training set of 60,000 images and test set of 10,000 images.

3. It has black and white images

4. It has 28x28 pixel bounding box

#Fig.)3..1'MNIST-- dataset-img'

#3..2)"EMNIST—dataset"

1. EMNIST is a database of handwritten character digits.

2. It has training set of 1,24,800 images and test set of 20,800 images

3. It has black and white images

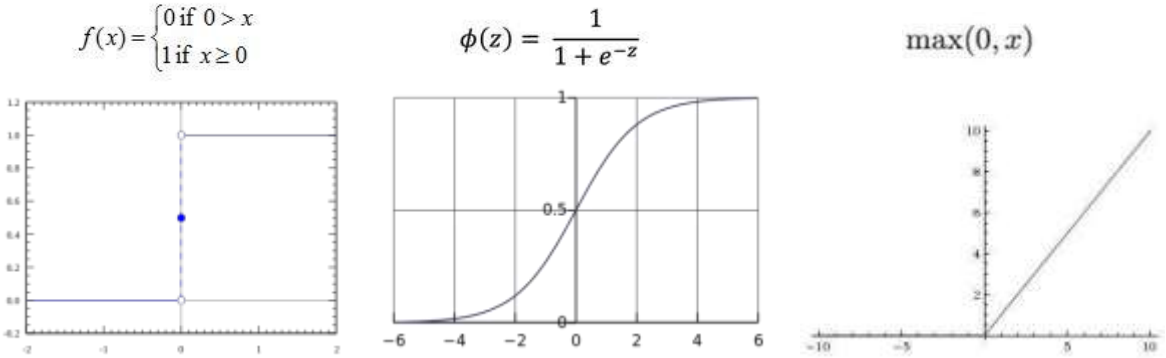
4. It has 28x28 pixel bounding box

#Fig.3.2)'EMNIST:Dataset-img'

#3..3"Activation-Functions"

1. The Activation function introduces non-linear properties to the Network.

2. They convert an input signal of a node in a Neural Network to an output signal.



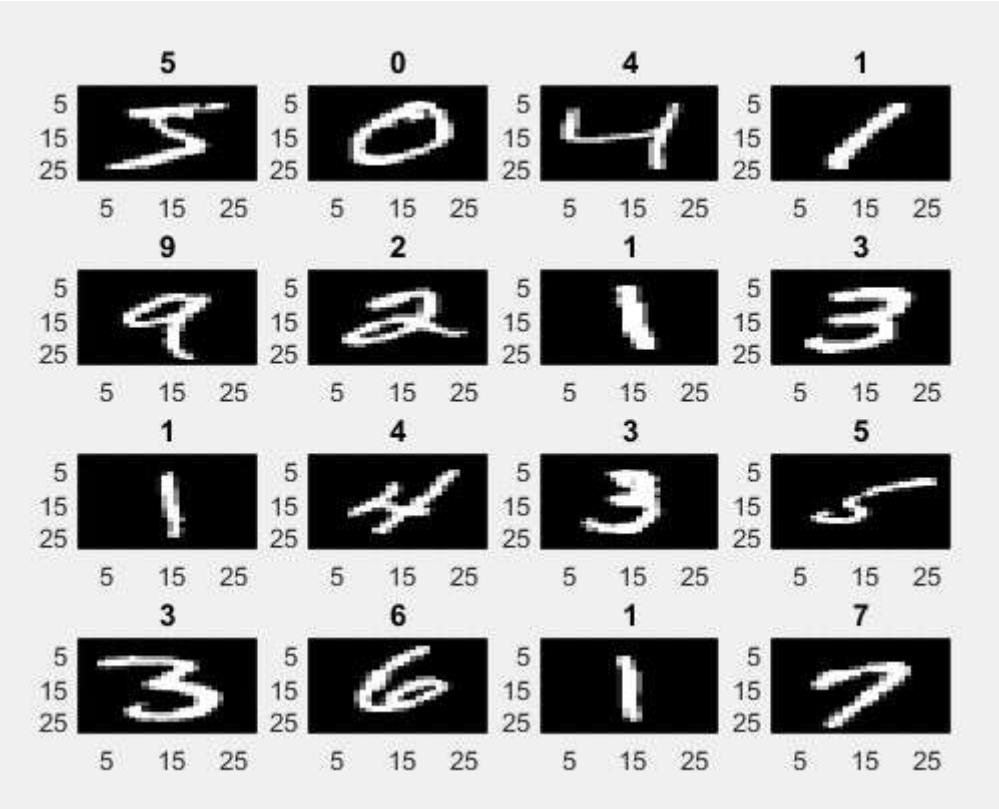
(a) (b) (c)

#Fig.3..3)'activation-func.(#a)step-func.(#b)sigmoid

(#c) Re-Lu func.'

#3..4)"Design sys. : MNIST-dataset"

Data set is divided into two parts-training set and test set. Training set has 60000 images of 28*28 grid size and test set has 10000 images of same size



. First 20 images from training set are : --

#fig .3..4)' 20 img. training-sets: MNIST-dataset'

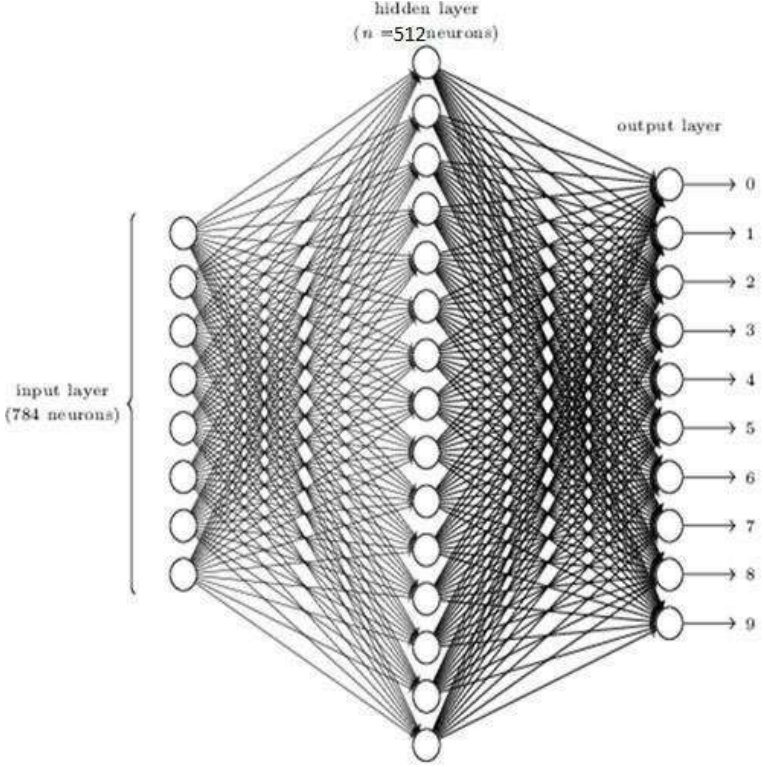
#3.4..1)"Artificial-Neural Networks"

2-layer's "neural-network"

1. Input layer :- It has input shape 28 *28.

2. Hidden layer : A single hidden layer is used with 128 nodes and ReLu activation function is used.

3. Output layer :- It has 10 nodes. Each node represents a class. The 10 classes are digits (0-9).



#fig.3.4..1)"2-layer:'neural-networks'"

'**Optimizer'**

Adam optimizer is used. Adam is an abbreviation for adaptive moment estimation. Adam Optimizer is a variant of stochastic gradient descent algorithm that's been designed specifically for training deep neural networks. It is the current best choice amongst gradient based convex o ptimization algorithm. An advantage of Adam Optimiser is that the hyperparameters require less tuning.

'**Lossfunction'**

Logarithmic loss function (sparse_categorical_crossentropy) is used. The goal of each training st ep is to minimize this loss function. This loss function is used in multiclass classification when the classes we are predicting are mutually exclusive.

'**Metric:Evaluation'**

Accuracy metrics or classification accuracy is used for evaluating performance of our model .

#3.4..2)"<u>Random-Forest"</u>

Before implementing random forest, reshape function is used. Reshape function is used to change dimensions of an array without changing its data. In our project, reshape function is use

d to convert three dimensional array to two dimensional array. In our project, 6 hyperparameters are used in random forest.

i. n_estimators :-

n_estimators defines the no. of trees in the forest. The default value of n_estimators is "100".

ii. max_depth :-

The max_depth is the maximum depth of the tree. The default value of max_depth is "none".

iii. max_features :-- The number of features to consider when looking for the best split:

a) If 'int', then consider max_features = features at each split.

b) If 'float', then max_features is a fraction and int(max_features * n_features) features are considered at each split.

c) If 'auto', then max_features=sqrt(n_features).

d) If 'sqrt', then max_features=sqrt(n_features) (same as "auto").

e) If 'log2', then max_features=log2(n_features).

f) If 'None', then max_features=n_features.

iv. max_leaf_nodes :-

Grows trees with max_leaf_nodes in bestfirst fashion. Best nodes are defined as relative reduction in impurity. If None then unlimited number of leaf nodes. The default value of max_leaf_nodes is none.

v. n_jobs : The number of jobs running in parallel for both fit and predict. None means program is using only '1' job and '-1' using all processors. The default value of n_jobs is 'none'.

vi. random_state : random_state is the seed required by the random number generator. It is used to make the results of the experiment can be reproduced.

#3..5)"Design's of sys.:' EMNIS- dataset'"

Data set is divided into two parts-training set and test set. Training set has 124800 images of 28*28 grid size and test set has 20800 images of same size

. First 25 images from training set are :--

fig#.3..5:"first-25 imgs training sets: EMNIST-dataset"

#3..5.1)'The <u>Artificial Neural-Networks'</u>

'3-Layer:neural_network'

1. <u>Input layer</u> :- It has input shape 28*28.

2. Hidden layer : Two hidden layers are used with 256 nodes and ReLu activation function is used.

3. Output layer : It has 26 nodes. Each node represents a class. The 10 classes are alphabets (A- Z).

'Optimizer'

Adam optimizer is used. Adam is an abbreviation for adaptive moment estimation. Ada m Optimizer is a variant of stochastic gradient descent algorithm that's been designed specifical ly for training deep neural networks. It is the current best choice amongst gradient based conve x optimization algorithm. An advantage of Adam Optimiser is that the hyperparameters require l ess tuning.

'loss-function'

Logarithmic loss function (sparse_categorical_crossentropy) is used. The goal of each training st ep is to minimize this loss function. This loss function is used in multiclass classification when t he classes we are predicting are mutually exclusive.

'Metric:Evaluation'

Accuracy metrics or classification accuracy is used for evaluating performance of our model .

#3.5.2)Randomforest

System design for random forest is same as described for MNIST.

“CHAPTER#4”

“Performance-Analysis”

#4..1)” Hyper-parameter tunings:Neural-networks in MNIST-dataset”:--

Hyperparameter tuning for MNIST dataset					
Expt no.	hidden layer nodes	Train accuracy	Train loss	Test accuracy	Test Loss
1	16	0.9523	0.1661	0.9452	0.1832
2	32	0.973	0.0936	0.9648	0.1193
3	64	0.9851	0.0526	0.9738	0.0875
4	128	0.9907	0.0332	0.9768	0.0747
5	256	0.9956	0.0174	0.9798	0.0691
6	512	0.9975	0.0101	0.9815	0.0648
7	1024	0.9966	0.0109	0.9809	0.0688

Tab.#4.1)’Hyperparameter tuning of Neural Network for MNIST dst

#4..2)’Hyper_parameter-tuning: Random-forest in MNIST-dataset’:--

Hyperparameter tuning of random forest for MNIST dataset						
Expt No.	n_estimators	max_depth	max_features	max_leaf_nodes	Train accuracy	Test accuracy
1	100	None	None	None	1	0.9676
2	200	None	None	None	1	0.9704
3	300	None	None	None	1	0.9714
4	500	None	None	None	1	0.9711
5	300	6	None	None	0.8925	0.8973
6	300	7	None	None	0.9158	0.918
7	300	8	None	None	0.9341	0.9279
8	300	9	None	None	0.9506	0.9414
9	300	10	None	None	0.9653	0.9476
10	300	12	None	None	0.987	0.9593
11	300	15	None	None	0.9973	0.9675
12	300	20	None	None	0.9993	0.9709
13	300	25	None	None	0.9998	0.9705
14	300	None	0.8	None	1	0.9636
15	300	20	0.8	None	0.9979	0.9635
16	300	None	None	15	0.8176	0.8254
17	300	None	None	28	0.8651	0.871
18	300	None	None	56	0.898	0.9027

Tab.#4.2)”Hyper_parameter-tuning Random Forest:MNIST-dataset” **#4..3)”Hyper-parameter tuning:Neural-Networks in EMNIST—dataset”:--**

Hyperparameter tuning for EMNIST dataset						
Expt no.	hidden layer nodes1	hidden layer nodes 2	Train accuracy	Train loss	Test accuracy	Test Loss
1	16	16	0.7725	0.7703	0.7677	0.788
2	32	32	0.8527	0.4844	0.8426	0.5208
3	64	64	0.9022	0.3033	0.8812	0.3839
4	128	128	0.9311	0.2012	0.899	0.3306
5	256	256	0.9514	0.1296	0.9117	0.3072
6	512	512	0.9602	0.1007	0.9164	0.3109
7	1024	1024	0.9634	0.0926	0.9162	0.3315

Tab.)#4..3)"Hyper-parameter tuning: neural-networks EMNIST dataset"

#4..4)"Hyperparameter-tuning:Random-forest EMNIST dataset":--

Hyperparameter tuning of random forest for EMNIST dataset						
Expt No.	n_estimators	max_depth	max_features	max_leaf_nodes	Train accuracy	Test accuracy
1	100	None	None	None	1	0.8837
2	200	None	None	None	1	0.8876
3	300	None	None	None	1	0.8885
4	100	6	None	None	0.6476	0.6452
5	100	8	None	None	0.7503	0.7393
6	100	10	None	None	0.8336	0.8044
7	100	12	None	None	0.9078	0.8457
8	100	15	None	None	0.9698	0.8724
9	100	20	None	None	0.9885	0.884
10	100	25	None	None	0.996	0.8859
11	100	None	None	15	0.502	0.5036
12	100	None	None	28	0.573	0.5728
13	100	None	None	56	0.6427	0.6412
14	100	None	None	100	0.6977	0.6939

#Tab.4.4)"hyper-parameter tune Random-forest:EMNIST-dataset"

#4..5)"Results of 'MNIST':neural-network"

#4.5.1)'Training-steps'

60000 images are used in training step.

```
Epoch 1/10
60000/60000 [==============================] - 4s 75us/sample - loss: 0.2621 - acc: 0.9252
Epoch 2/10
60000/60000 [==============================] - 5s 80us/sample - loss: 0.1076 - acc: 0.9681
Epoch 3/10
60000/60000 [==============================] - 4s 70us/sample - loss: 0.0704 - acc: 0.9800
Epoch 4/10
60000/60000 [==============================] - 4s 71us/sample - loss: 0.0503 - acc: 0.9852
Epoch 5/10
60000/60000 [==============================] - 5s 79us/sample - loss: 0.0377 - acc: 0.9886
Epoch 6/10
60000/60000 [==============================] - 5s 80us/sample - loss: 0.0273 - acc: 0.9923
Epoch 7/10
60000/60000 [==============================] - 5s 82us/sample - loss: 0.0204 - acc: 0.9947
Epoch 8/10
60000/60000 [==============================] - 5s 80us/sample - loss: 0.0152 - acc: 0.9961
Epoch 9/10
60000/60000 [==============================] - 5s 79us/sample - loss: 0.0120 - acc: 0.9973
Epoch 10/10
60000/60000 [==============================] - 5s 84us/sample - loss: 0.0101 - acc: 0.9975
```

fig.#4.5.1)"Train-loss & train-accuracy:MNIST-dataset"

In this loss is 0.0101 and accuracy is 0.9975.

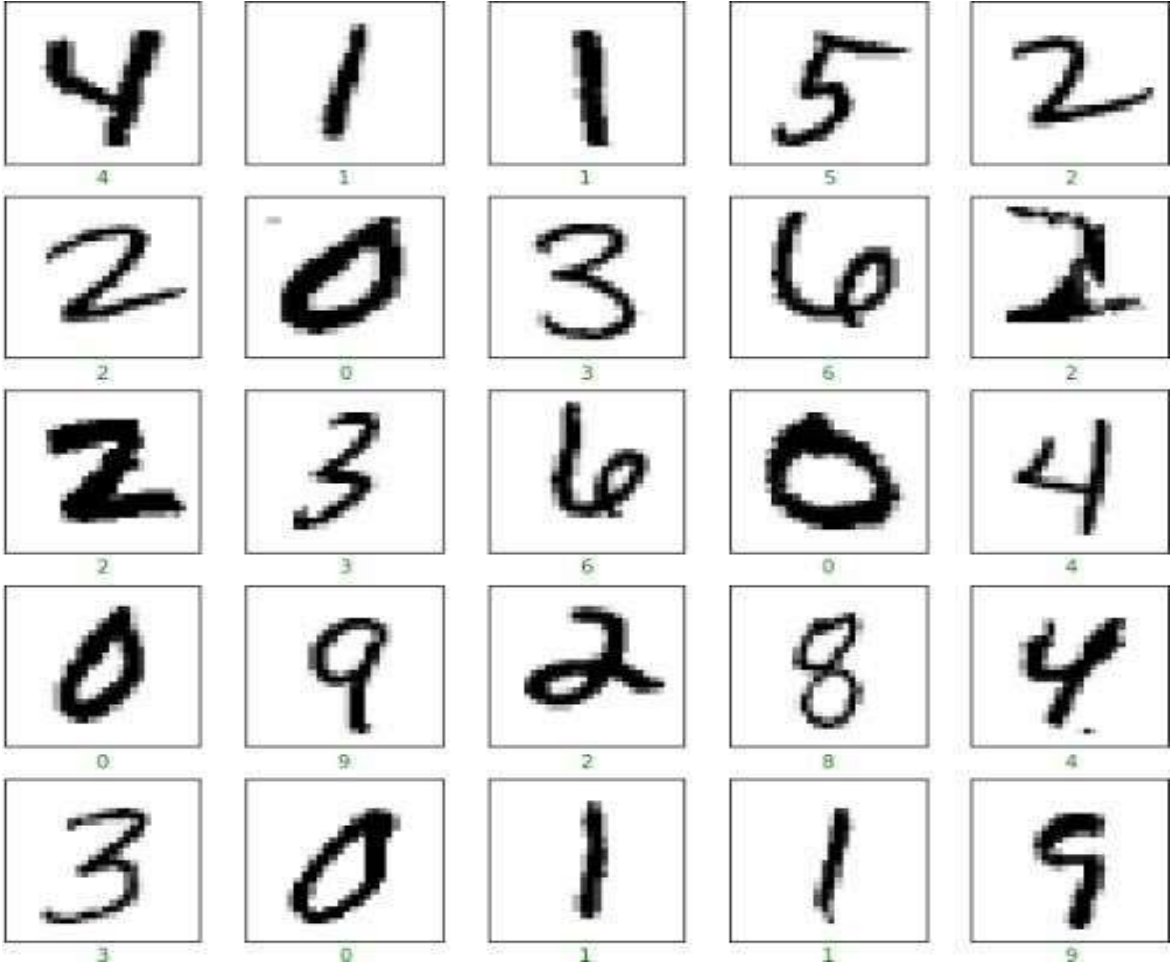
#4.5..2)"Testing-steps"

10000 images are used in testing step.

In this loss is 0.0684 and accuracy is 0.9785.

#4..5.3)"Visualizing of prediction"

First 25 images from test set along with predictions are shown as below :-_



#fig.4.5.3)"first-25 img's : test-set Neural-networks"

#4..6)"Results:MNIST Using Random-Forest"

#4.6..1)"Training-step"

60000 images are used in training step. In this accuracy is 1.0

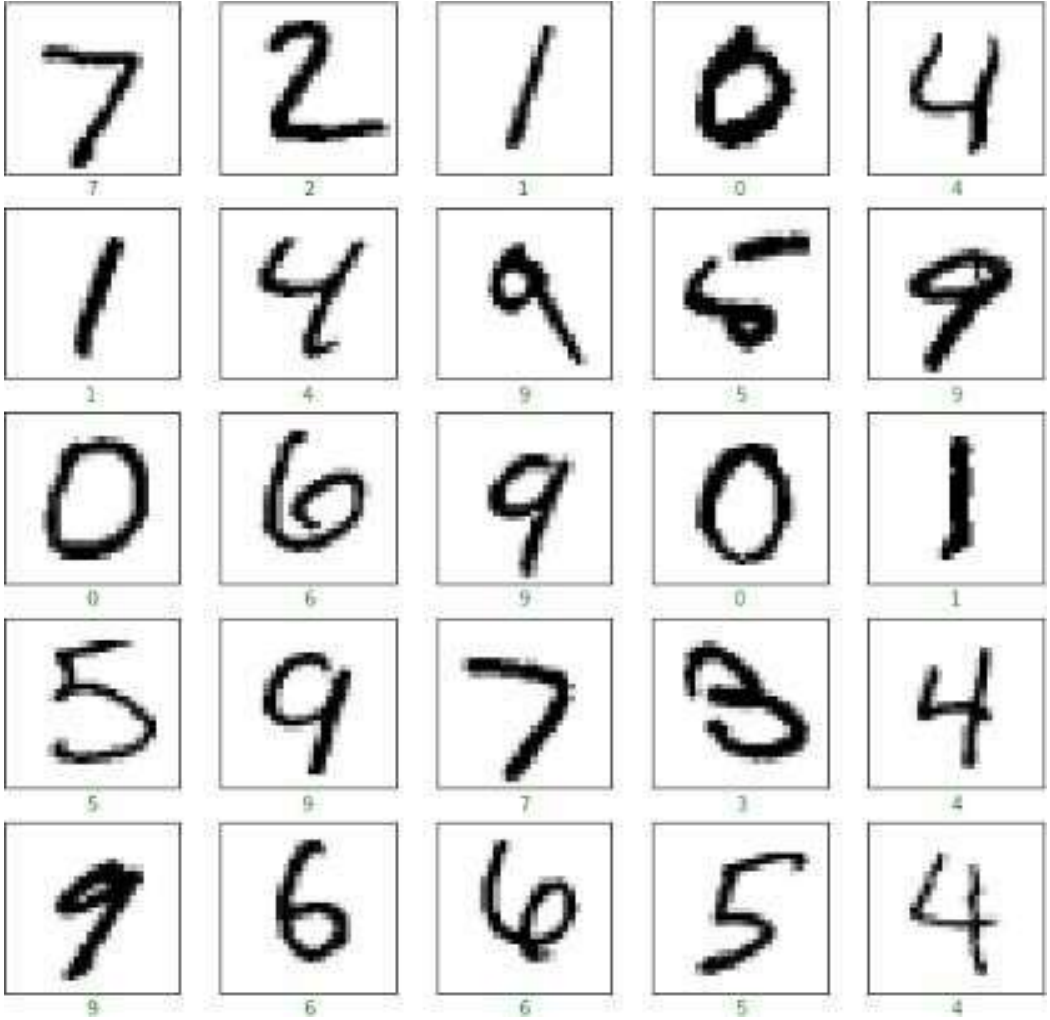
#4.6..2)"Testing-Step"

10000 images are used in testing step

. In this accuracy is 0.9721.

#4.6..3)"Visualising :Predictions"

25 random images from test set along with predictions are shown as below :--



#fig.4.6.3)"25 random img's test-set: random-forest"

#4..7)"Results:EMNIST used Neural-Networks"

#4.7..1)"Training-step"

124800 images are used in training step.

```
Epoch 1/10
124800/124800 [==============================] - 10s 77us/sample - loss: 0.6727 - acc: 0.7989
Epoch 2/10
124800/124800 [==============================] - 9s 73us/sample - loss: 0.3410 - acc: 0.8911
Epoch 3/10
124800/124800 [==============================] - 9s 71us/sample - loss: 0.2746 - acc: 0.9098
Epoch 4/10
124800/124800 [==============================] - 9s 72us/sample - loss: 0.2360 - acc: 0.9201
Epoch 5/10
124800/124800 [==============================] - 9s 69us/sample - loss: 0.2086 - acc: 0.9273
Epoch 6/10
124800/124800 [==============================] - 9s 70us/sample - loss: 0.1865 - acc: 0.9344
Epoch 7/10
124800/124800 [==============================] - 9s 70us/sample - loss: 0.1703 - acc: 0.9389
Epoch 8/10
124800/124800 [==============================] - 9s 70us/sample - loss: 0.1531 - acc: 0.9441
Epoch 9/10
124800/124800 [==============================] - 9s 69us/sample - loss: 0.1423 - acc: 0.9475
Epoch 10/10
124800/124800 [==============================] - 9s 70us/sample - loss: 0.1296 - acc: 0.9514
```

fig.#4.8..1)"Trainloss & train-accuracy:EMNIST-dataset"

In this loss is 0.1296 and accuracy is 0.9514.

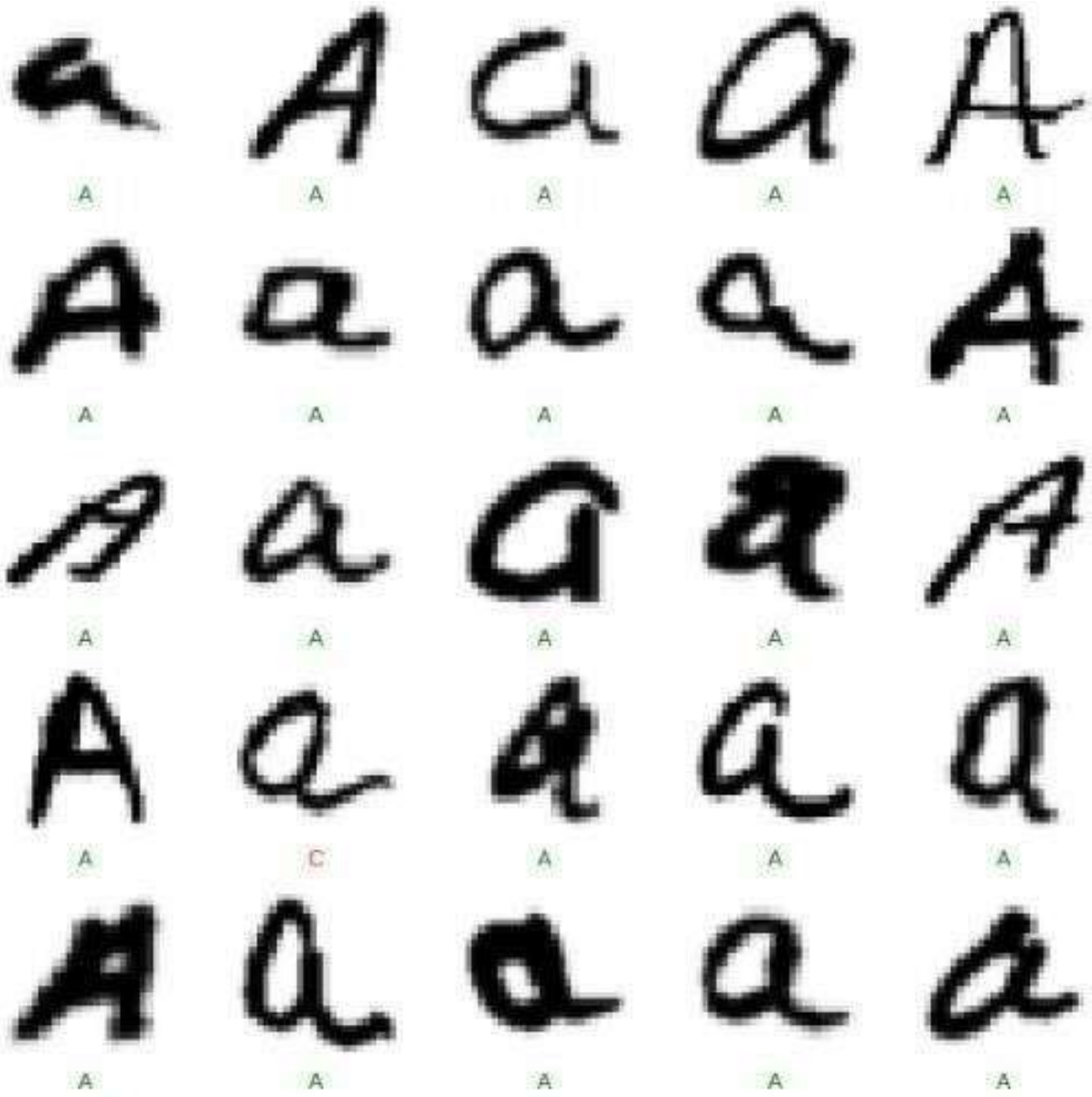
#4.7..2)"Testing-steps"

20800 images are used in testing step.

In this loss is 0.3085 and accuracy is 0.9107.

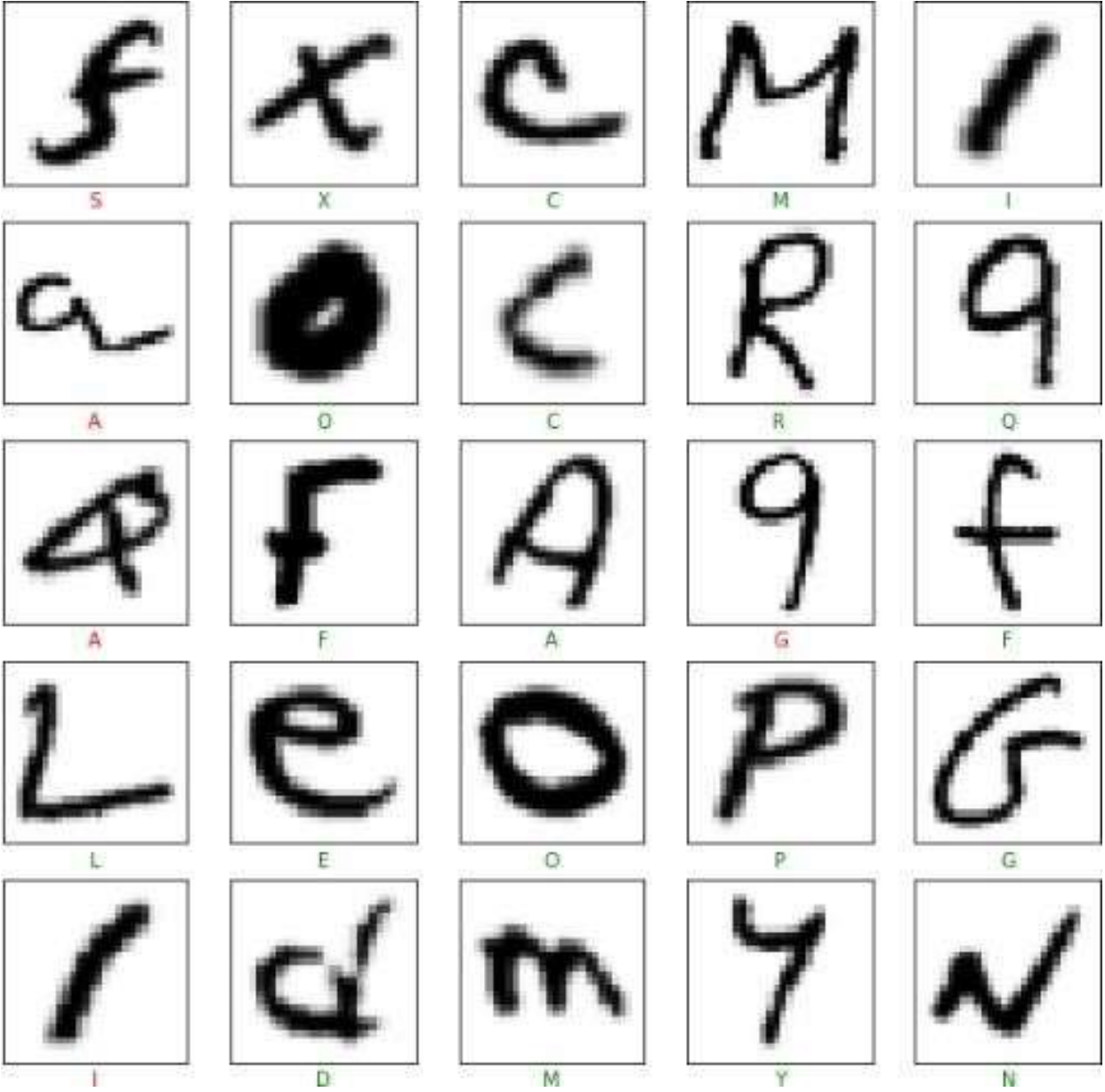
#4.7..3)"Visualizing the: predictions"

First 25 images from test set along with predictions are shown as below :-



#a

25 random images from test set along with predictions are shown as below :-



#b

fig.#4..8.3)"Visualising-predictions:Neural-networks"#a)First-25-imgs:test-set

#b)"25random-imgs:test-set"

#4..8)"Results-EMNIST:used Random-Forest"

#4.8..1)"Training-steps"

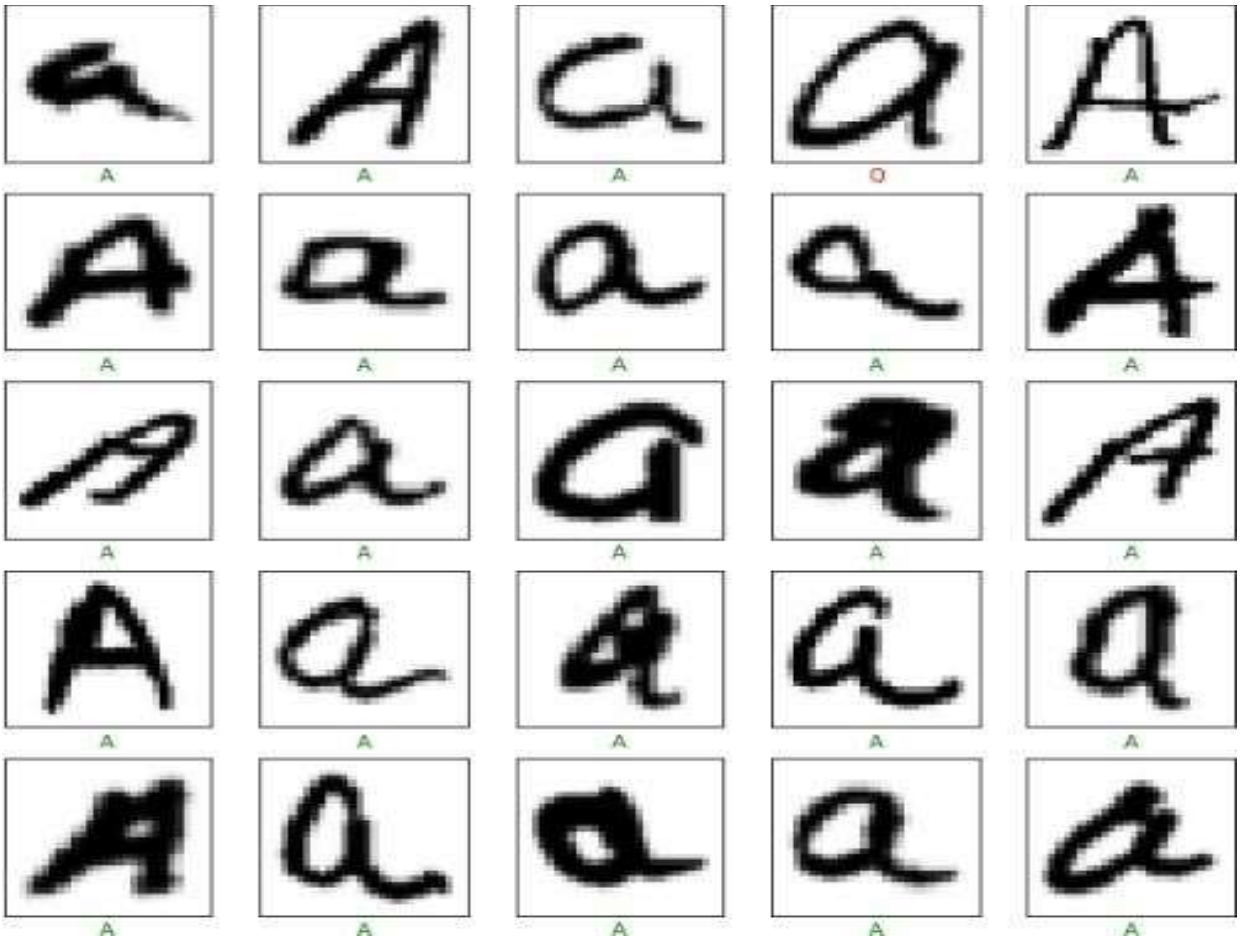
124800 images are used in training step. In this accuracy is 1.0

#4.8..2)"Testing-Steps"

20800 images are used in testing step

. In this accuracy is 0.8858. **#4.8..3)"Visualising-predictionss"**

First 25 images from test set along with predictions are shown as below :-



#a

25 random images from test set along with predictions are shown as below :-

#b

fig.#4.9..3)"Visualising-predictionss:Random-forests"#a)'First-25-imgs:test-set

#b)25-random-imgs:test-set

#4..9)"Results-compare"

We used ‘MNIST’ and ‘EMNIST’ dataset to create a handwriting detection system. Using MNI ST dataset, it can learn to recognize numbers from 0 to 9. Using EMNIST dataset, it can learn t o recognize alphabets from A to Z. We compared two different models namely artificial neura l network & random forest on both ‘MNIST’ and ‘EMNIST’ dataset. We experimented with different hyperparameters to tune these models. The results are as follows :-

Result Comparison

S.No	Dataset	Model	Best Hyperparameters	Test accuracy
1	MNIST	Artificial Neural Network	1 hidden layer, no. of nodes in hidden layer=512	0.9815
2	MNIST	Random Forest	n_estimators=300, max_depth=None, max_features=None, max_leaf_nodes=None	0.9714
3	EMNIST	Artificial Neural Network	2 hidden layer, no. of nodes in hidden layer=256	0.9117
4	EMNIST	Random Forest	n_estimators=300, max_depth=None, max_features=None, max_leaf_nodes=None	0.8885

#Tab.4..9)’Result-compare’

CHAPTER-#5

“THE_CONCLUSION”

#5..1)’Conclusions’

For MNIST dataset, the performance of Artificial Neural Network is ‘98.15%. F

or EMNIST dataset, the performance of Artificial Neural Network is ‘91.17%’.

Going further we can use Artificial Neural Network for handwriting detection on mor
e challenging cases.

#5..2)'Future-scopes'

There are several new techniques like Convolutional neural networks, Recurrent neura
l networks, Long-Short Term Memory units etc.

We can also use convolution neural network for handwriting detection.

REFERENCES

[1] Gil Levi and Tal Hassner, "Offline handwritten digit recognition using neural network" , International Journal of Advanced Research in Electrical, Electronics and Instrumentatio n Engineering, vol. 2, no. 9, pp. 4373-4377, 2013.

[2] Haider A. Alwzwazy1 , Hayder M. Albehadili2 , Younes S. Alwan3 , Naz E. Islam4 ,"Handwritten Digit Recognition Using Convolutional Neural Networks", International Journal of Innovative Research in Computer and Communication Engineering, vol. 4, no. 2, pp. 1101- 1106, 2016.

[3] DR.KUSUMGUPTA2 ,"A comprehensive review on handwritten digit recognition usi ng various neural network approaches", International journal of enhanced research in manageme nt & computer applications, vol. 5, no. 5, pp. 22-25, 2016.

[4] Faisal Tehseen Shah, Kamran Yousaf,"Handwritten Digit Recognition Using Ima ge Processing and Neural Networks", Proceedings of the World Congress on Engineering, vol., 200 7.

[5] Sherif Abdel Azeem, Maha El Meseery, HanyAhmed, ”Online Arabic Handwritten Di gits Recognition “,Frontiers in Handwriting Recognition (ICFHR), 2012

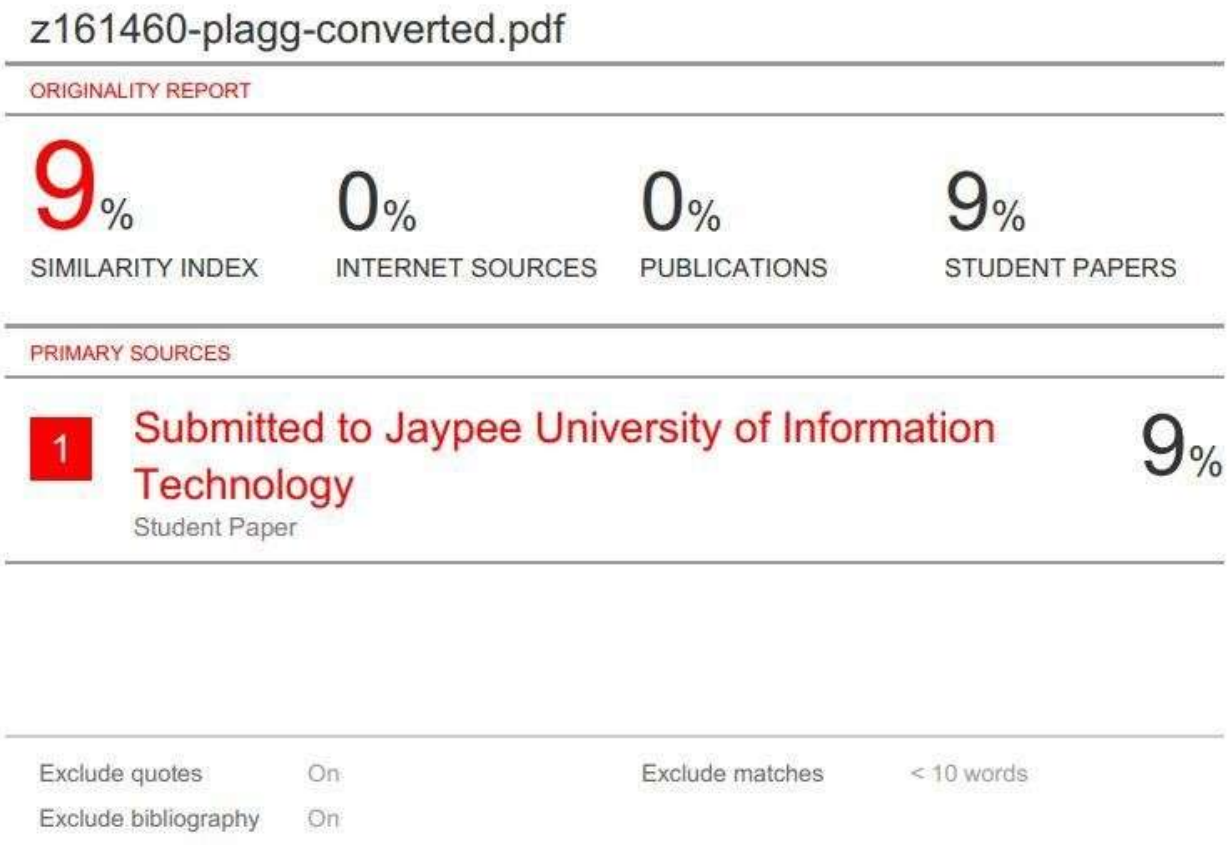
[6] Yusuf Perwez, Ashish Chaturwedi, ”Neural Networks for Handwritten English Alphab et Recognition”, 2011

[7] Gregory Cohen, Saeed Afshar, Jonathan Tapson, Andre van Schaik, “EMNIST: an extension of MNIST to handwritten letters”, 2017

[8] Diederik Kingma, Jimmy Ba, “Adam: A Method for Stochastic Optimization”, 2017

[9] C. Zhang & P. C. Woodland, "Parameterised Sigmoid and ReLU Hidden Activation Functio ns for DNN Acoustic Modelling", 2015

[10] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," Proc. International conference on artificial intelligence and statistics, 2010.



JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT

PLAGIARISM VERIFICATION REPORT

Date: 15/07/2020

Type of Document (Tick): PhD Thesis | M.Tech Dissertation/ Report | B.Tech Project Report ✓ | Paper

Name: RISHABH TEWARI **Department:** CSE & I.T **Enrolment No** 161460

Contact No. 7236002980 **E-mail.** rishabhtewari.juit@gmail.com

Name of the Supervisor: MS. MONIKA BHARTI

Title of the Thesis/Dissertation/Project Report/Paper (In Capital letters):

HANDWRITING DETECTION USING NEURAL NETWORKS

UNDERTAKING

I undertake that I am aware of the plagiarism related norms/ regulations, if I found guilty of any plagiarism and copyright violations in the above thesis/report even after award of degree, the University reserves the rights to withdraw/revoke my degree/report. Kindly allow me to avail Plagiarism verification report for the document mentioned above.

Complete Thesis/Report Pages Detail:

- Total No. of Pages = 49
- Total No. of Preliminary pages = 10
- Total No. of pages accommodate bibliography/references = 1

Rishabh Tewary

(Signature of Student)

FOR DEPARTMENT USE

We have checked the thesis/report as per norms and found **Similarity Index** at 09(%). Therefore, we are forwarding the complete thesis/report for final plagiarism check. The plagiarism verification report may be handed over to the candidate.

Monika Bharti

(Signature of Guide/Supervisor) **Signature of HOD**

FOR LRC USE

The above document was scanned for plagiarism check. The outcome of the same is reported below:

Copy Received on	Excluded	Similarity Index (%)	Generated Plagiarism Report Details (Title, Abstract & Chapters)	
	• All Preliminary Pages • Bibliography/Images/Quotes • 14 Words String		Word Counts	
Report Generated on			Character Counts	
		Submission ID	Total Pages Scanned	
			File Size	

Checked by
Name & Signature **Librarian**

Please send your complete thesis/report in (PDF) with Title Page, Abstract and Chapters in (Word File) through the supervisor at plagcheck.juit@gmail.com