ISSN No:-2456-2165

Recognizing Handwritten Alphabets using Neural Networks

Mimoh Kumar Jaiswal[1], Manisha Kasyap[2], Jatin Mulchandani[3], Netra Patil[4]
[1,2,3]Student, Bharati Vidyapeeth Deemed to be University, College of Engineering, Pune
[4]Associate Professor, Bharati Vidyapeeth Deemed to be University, College of Engineering, Pune

Abstract:- Handwriting recognition has been one of the most intriguing problems of the modern era. It can be considered as a superset of the character recognition problem and an entry point in machine learning. Throughout the years, various approaches have been designed to tackle the problem. One such approach is formulated in this paper. We have designed our system on Artificial Neural Networks which mimics the neurons of the human brain. The network uses either of gradient feature extraction and geometric feature extraction techniques to train and to generate output. The efficiency and accuracy of the output is superior than the other approaches used in the field. It is concluded that the problem of character recognition is best tackled by Neural Networks. Neural Networks have been applied to a range of problems like stock market prediction, image processing, medicine. An elaborated view of the system is provided in the following paper.

Keywords:- *Handwriting Recognition; Neural Networks; Gradient feature extraction; Geometric feature extraction.*

I. INTRODUCTION

The human brain is one of the most fascinating objects in the world. It is rightfully the epitome of what modern day computers strive for. Since the advent of technology, humans have been obsessed with making systems that can work at par compared to the human brain. The K from Fujitsu, regarded as the world's fastest computer, is four times faster than the human brain and holds 10 times more information. Sure these are impressive figures but what makes the brain win the comparison is that it can do this by taking a lot less power, lesser than a light bulb in fact. Though computers can calculate a lot faster than an average human being, the real gem that human beings have is their ability to recognize and learn from that recognition. Babies, still in their wombs can understand their mother's voice and can even respond. Similarly, our brain excels in recognizing images. Even babies can identify a product correctly even if subtle changes are applied to it. The same case is with handwriting. Millions of people have different styles of handwritings. Yet, the human brain can recognize, least take a guess on the character, which most of the time turns out to be correct. Human brain works in a manner that is so complex that scientists are yet to figure out a way to correctly annotate it with mathematics. In the case of recognition, the brain tries to learn from its surroundings. For example, if the brain see a word which is incomprehensible at the first look, it tends to look at the other characters surrounding that word. This makes our brain understand the meaning of the sentence through which it make a guess for that word or character. This however, may seem easy because of our inherent ability to do it without even blinking, but the challenge of making a computer understand is a whole another discussion. The brain does all the above work using a cell known as Neuron. Neurons are the basic element of the Central Nervous System and are responsible for all the learnings we gain. Electrical impulses generated, transfer information to other neurons. The actions we perform as a response is governed by neurons. The working of neurons was described by the work of Warren McCulloch and Walter Pitts in 1943[1]. The work described how interconnections between neurons work inside the brain and paved the way for further research in the field. This is how Artificial Intelligence came into existence. Today's Artificial Intelligence is best accomplished by Neural Networks which is an extension of the brain's working into the field of Computer Science.

II. ARTIFICIAL NEURAL NETWORKS

Neural Networks are a programming archetype which is inspired by the biological functioning of neurons. They learn by learning from data which is usually observational in nature. Since it is inspired by the neurons, a similar entity is created to replicate the functioning of neurons. Neurons learn by sending short electrical pulses which passes down to other neurons. The neuron is composed of dendrites and axon. The axon can be considered a conducting path through which the impulses pass and the dendrites are a set of fine structures that receive these signals. Synapses occur at the end of axon which distinguishes which type of action is to be performed. When the signal received by a dendrite has a larger input then, it is considered as an excitatory input.

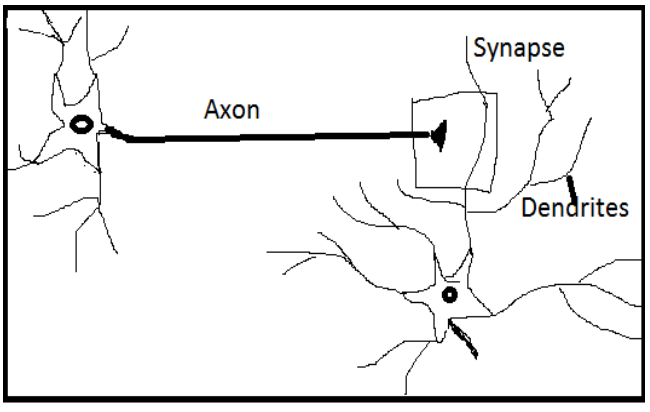


Fig 1:- Neurons sending signals to other neurons

Neural Networks have perceptrons which emulate the neurons. It has two important characteristics which explains their concept. It has sets of weights that adapt themselves according to the learning algorithm and can approximate non - linear functions. The weights are a measure of the connection strength between perceptrons. They get activated during training tasks and during predictive tasks. The weights

adaptiveness makes neural networks perfect for solving complex problems as it gives the programmer a control over the network and it can finely tune the weights according to the problem. This capability is of high value. It highly decreases the error rate of the output.

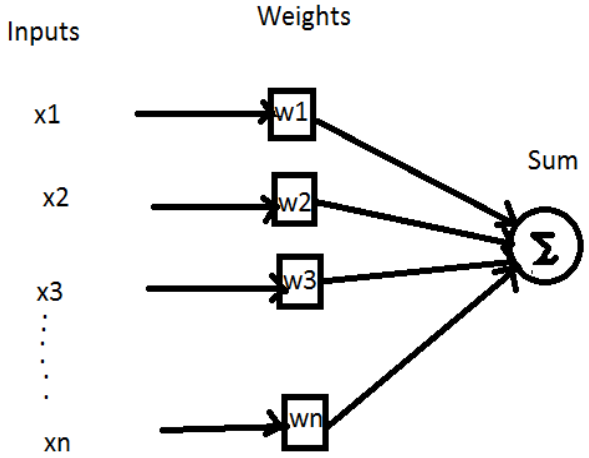


Fig 2:- Perceptron Model

$$\text{Output} = \begin{cases} 0, & \text{if } \sum_j w_j x_j + b \leq 0 \\ 1, & \text{if } \sum_j w_j x_j + b > 0 \end{cases}$$

The above expression shows the condition for output being 0 or 1. Here, the weights are represented by w_j and inputs are represented as i_j, 'b' is called the bias for the neural network. It signifies two important points;

1. The neural network will output 1 if b is much larger

2. The neural network will output 0 if b is smaller.

The disadvantage of this concept is that any change in the weights changes the output significantly. This may result in huge errors. To overcome this, we modify our neurons and use Sigmoids instead. They are similar to perceptrons but change slightly to a small change in weights. They can output all values between 0 and 1, achieving more accuracy. The sigmoid function is defined as:

$$\sigma(z) = 1 / (1 + e^{-z})$$

The sigmoid function replicates a perceptron at the extreme values but differs for intermediate values.

1. If $e^{-z} \to 0, \sigma(z) \to 1$, this means if $z = w.x + b$ is large and positive, the output will be 1.

2. If $e^{-z} \to -\infty, \sigma(z) \to 0$, this means that if $z = w.x + b$ is negative, then the output is 0.

The σ is called the sigmoid function. The plot of the sigmoid function gives us a smooth curve and not a step function. In further research, various types of activation function were devised of the form $f(w.x + b)$ of some other activation function f(.).

III. PROPOSED MODEL

Based on this concept we created this model where we use the neural network for training and then testing it with a dataset. The problem is a machine learning problem and hence needs to be tackled in steps.

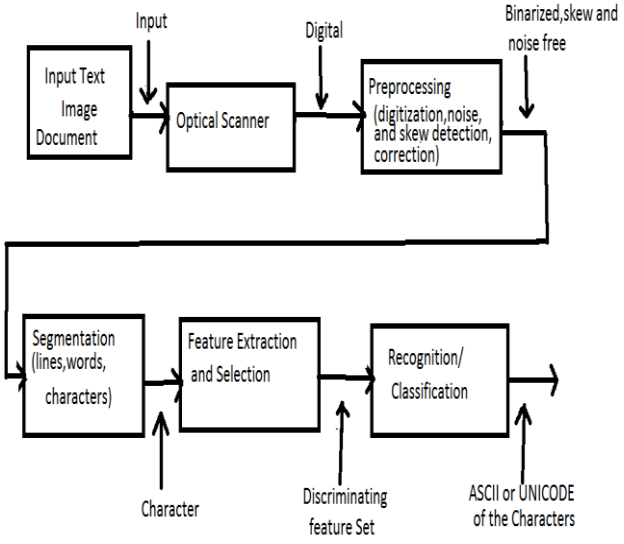


Fig 3:- Flow diagram of the proposed model

A. *Input text acquisition*

In this step, the image of the handwritten text is acquired. These are the scanned files of the actual written characters and are usually in JPEG/PNG format. They can be scanned through any medium be it a scanner or a digital camera[5].

B. *Preprocess of the Input text*

The second step of the recognition system is to process the image file acquired in the first step. This process is crucial to the whole system as it affects the accuracy as well as efficiency of the system. In this step, the image is digitized and then any sort of noise is removed. Noise may be present around the text that is to be recognized which may hinder in the recognition phase. Here, the image is now converted to gray scale. The aim is to make the text stand out from the background. After this, the image is binarized which converts the image from grayscale to binary image. Edge detection algorithm is applied to detect edges using Sobel technique. Its aim is to detect discontinuities in brightness.

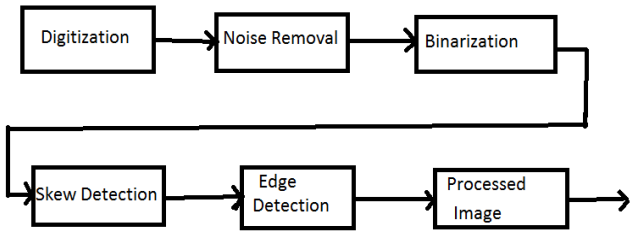


Fig 4:- Preprocess of the Input text

C. *Segmentation*

The third step of the task is Segmentation. In this task, the image is read and decomposed into sub images of characters and are detected one by one. The aim is to segregate every character such that they may be individually recognized. This is done using the blob detection feature. But first, the image is normalized. Normalization of the image makes every character fit into the same block as every other character. The blob is then detected.

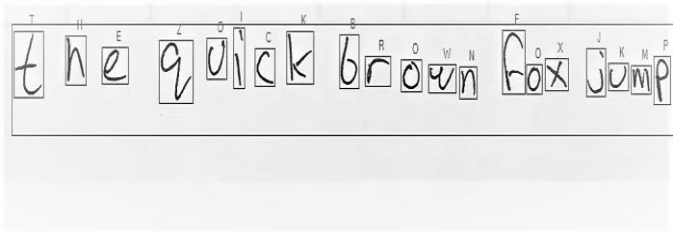


Fig 5:- Blob detection for a text

D. Feature Detection and Extraction

Feature Detection methods is one of the most important aspect of any recognition system. It extracts features that are needed to be recognized and its accuracy determines the overall accuracy of the system and reduces misclassification. In this section we propose two methods for feature extraction :

1. Gradient feature extraction

2. Geometric extraction based on character geometry.

- *Gradient feature extraction*

This method of feature extraction identifies feature within an image using the Gradient Vector [3]. The gradient vector is defined as a vector having both Magnitude and Direction. It is calculated by applying derivatives along horizontal and vertical axes. It is shown by $\mathbf{[G_x, G_y]^T}$, where $\mathbf{G_x}$, $\mathbf{G_y}$ are Gradient Vector components along the Horizontal and Vertical direction. In the model, it is calculated using the Sobel Operator. It is defined as,

1	2	1
0	0	0
-1	-2	-1

Horizontal Component

1	0	-1
2	0	-2
1	0	-1

Vertical Component

Fig 6:- Sobel Operator Masks

(i-1,j-1)	(i-1,j)	(i-1,j+1)
(i,j-1)	(i,j)	(i,j+1)
(i+1,j-1)	(i+1,j)	(i+1,j+1)

Fig 7:- 8- Neighbourhood of pixel (i, j)

Using the sobel masks, we determine G_x, G_y. If an input image is given, then for a pixel at (i, j), it's neighbouring pixels are calculated by convolving sobel masks with the 8 neighbourhood pixels as,

$G_x(x,y) = 1(i-1, j-1)+2(i-1, j)+1(i-1, j+1)-1(i+1, j-1)+2(i+1, j)-1(i+1, j+1)$

$G_y(x,y) = 1(i-1, j-1)+2(i, j-1)+1(i+1, j-1)-1(i-1, j+1)+2(i, j+1)-1(i+1,j+1)$

The Gradient Strength and Direction can be further calculated as:

$$Magnitude = |G(i,j)| = \sqrt{Gx(i,j)^2 + Gy(i,j)^2}$$

$$Direction = |G(i,j)| = tan^{-1}[Gy(i,j) / Gx(i,j)]$$

The magnitude and direction are then separated into 8 Chaincode directions.

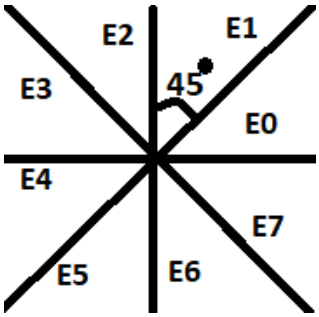


Fig 8:- Chaincode Directions with equal angles between vectors

After this, a Gaussian Filter is applied to the 8 chaincode to downsample the 9*9 vector to 5*5 gaussian vector. This produces a feature count of 200(5 Horizontal, 5 Vertical, 8 Directional resolution). Further steps are taken to reduce aliasing which occurs due to down sampling.

- *Geometry based feature extraction*

This technique takes inspiration from how humans can perform visually challenging tasks by extracting perceptual information of the surroundings. In this method, we try to generalize the features of all the 26 alphabets by segregating them in terms of lines and curves[2]. First, we define a matrix which exactly contains the alphabet in question. Then we partition the matrix into a fixed number of smaller matrices. This partitioning helps to go into the depth of character geometry. The smaller matrices are termed as "zones"[4]. These zones can help identify a character at a higher rate and with more accuracy. The concept is introduced because whatever be the style of handwriting a person possesses, there will be some common features that can distinguish alphabets from one another. For example, E and F can be distinguished based on the extra line segment E has at its bottom. This extra line segment can be found out through the zones we created.

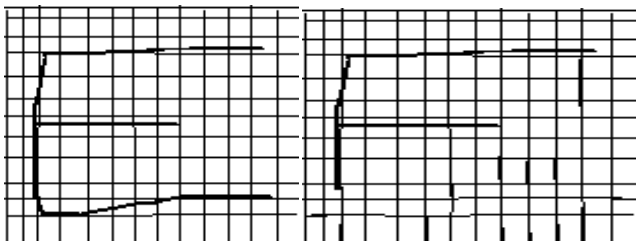


Fig 9:- Differentiating characters based on zones

After creating zones, we define a feature vector of individual zones. Each feature vector has certain parameters to count, which decides the character. These parameters are then normalized for their length. The parameters determined are, the number of horizontal, vertical, right and left diagonals along with their normalized lengths.

Normalization of length occurs using the formula, Normalized length = number of pixels of the line / total pixels

in that zone. Feature vectors were then calculated for each zone.

```
% feature_vectors
column_zone1_features=lineclassifier(column_zone1);
column_zone2_features=lineclassifier(column_zone2);
column_zone3_features=lineclassifier(column_zone3);

row_zone1_features=lineclassifier(row_zone1);
row_zone2_features=lineclassifier(row_zone2);
row_zone3_features=lineclassifier(row_zone3);
```

Recognition of the image

After features are extracted, the network needs to be trained for the input dataset. As the neural networks are explained in detail in the previous sections, here we will have a look on how the neural network of our system reduces its error rate for other inputs[6].

In the neural network, the error rate defines the deviation between the desired output and the actual output. Backpropagation algorithm tries to reduce the error rate. It works towards an optimal error rate at each layer so that their it results in an optimal output. We define some terms for this,

1. Error derivative rate for weights: This defines the rate of error when the weights are changed i.e., increased or decreased slightly.

2. Error rate for activity of units: This defines the error rate when the units in the layer output differently according to the weights. For example, this error rate for the output layer is the difference between the actual output and the calculated output. But for the hidden layers, it gets slightly complicated. Assume a hidden layer just behind the output layer. To calculate the error rate for activity of this unit, we find the weights between the output layer and this layer, then multiply by the error rates of the output units and add all the products. This gives the activity error rate of the hidden layer. This method can go on till the first layer i.e., the input layer is reached. The algorithm tries to reduce the error derivative rate for the weights consecutively and reaches a global optimum.

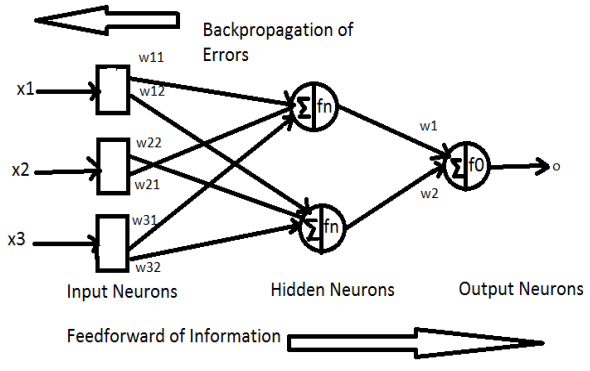


Fig 10:- Errors backpropagating to previous layers

IV. CONCLUSION

This paper has demonstrated a way to identify handwritten alphabets in an Offline mode. The crucial component being the Feature Extraction methodologies and the reduction of errors using the Backpropagation algorithm in the training as well as testing of the Neural Network. The system creates 9 features using Geometric feature extraction algorithm and 12 features using gradient feature extraction. Below are the performance comparison of the model using the Gradient vector technique and Geometric feature extraction technique.

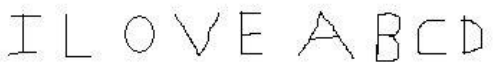
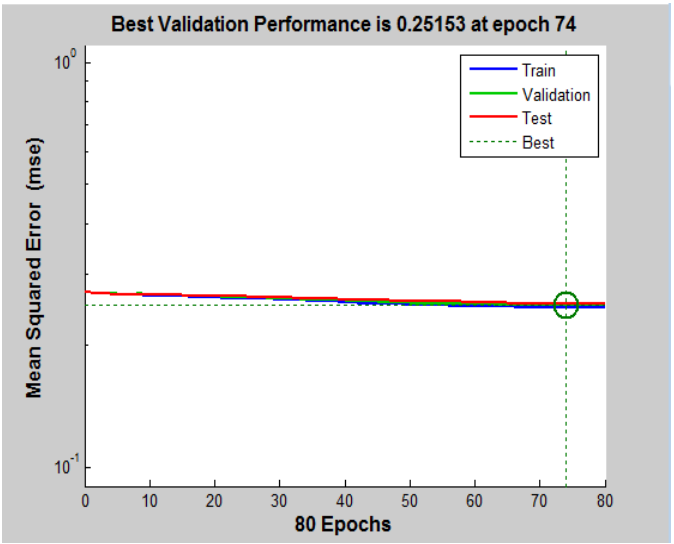
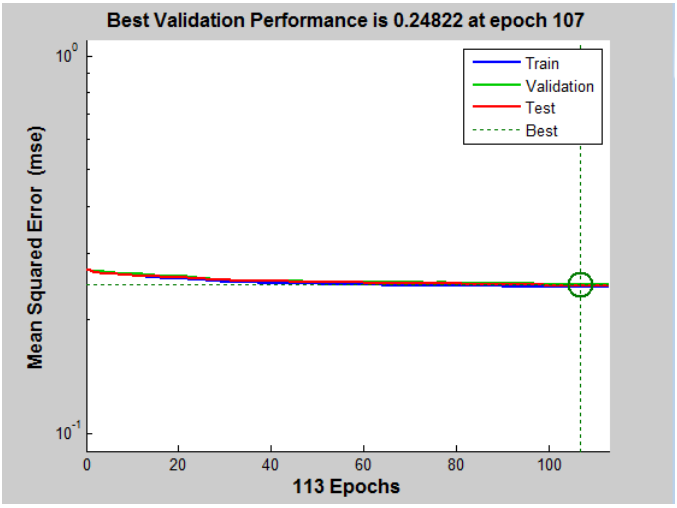


Fig 11:- Input image



Gradient technique accuracy on test set = 88.5%

Output in plain text = Z EOVE AACO



Geometric feature extraction accuracy on test set = 85.8%

Output in plain text = I LOVE AECO

The comparison shows that Gradient technique performs better than the Geometric feature extraction technique. But from the output we see that the latter technique achieves better results than the former. This is mainly because

of the more number of features created by the geometric feature extraction for training and testing sets.

REFERENCES

[1]. Warren McCulloch, Walter Pitts, "A Logical Calculus of the ideas Imminent in Nervous Activity," University of Illinois.

[2]. M. Blumenstein, B. K. Verma and H. Basli, "A Novel Feature Extraction Technique for the Recognition of Segmented Handwritten Characters", 7th International Conference on Document Analysis and Recognition (ICDAR '03) Edinburgh, Scotland.

[3]. Ashutosh Agarwal, Karamjeet Singh, Kamaljeet Singh, "Use of Gradient Technique for Extracting Features from Handwritten Gurmukhi Characters and Numerals, International Conference on Information and Communication Technologies, 2014.

[4]. Dinesh Dileep, "A Feature Extraction Technique Based On Character Geometry For Character Recognition".

[5]. Plamondon, Réjean, and Sargur N. Srihari. "Online and offline handwriting recognition: a comprehensive survey." Pattern Analysis and Machine Intelligence, IEEE Transactions (2000).

[6]. J.Pradeep, E.Srinivasan, S.Himavathi, "Diagonal Based Feature Extraction For Handwritten Alphabets Recognition System Using Neural Network",International Journal of Computer Science & Information Technology (IJCSIT), Vol 3, No 1, Feb 2011.

[7]. Christos Stergiou and Dimitrios Siganos, "Neural Networks".