International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 06 Issue: 07 | July - 2022 Impact Factor: 7.185 ISSN: 2582-3930

Implementation of English Handwritten Character Recognition System using Neural Network

Madhav Sharma

Research Scholar (Ph.D.)

Computer science & Engineering

Faculty of Engineering & Technology

Jagannath University, Jaipur

Abstract: The technique of figuring out the handwritten textual content using device interface is known as handwritten man or woman popularity. Handwritten characters gift demanding situations due to the fact they range from one author to the other; even though the same author writes the identical character, the form, size, and place of the character vary. Handwritten character recognition has been one of the difficult areas in the field of picture processing. It has numerous applications which consist of, financial institution cheques and conversion of any handwritten report into digital text shape. There's a outstanding want forstoring statistics to desktop garage f rom the facts to be had in handwritten images to later reprocess this facts utilizing computers. But to re-system this facts, it might be very hard to examine the other information from these picture files. Consequently, a way to automatically clear up and save information, in particular text, from photograph documents is needed. This paper ambition to classify a character handwritten using dropout changes to 0.10 to 0.75 and implement the result as per epochs changes. Model will trained and softmax technique is used to train model using modified gradient descent technique.

Keywords: Handwritten, Neural Network, Gradient Descent, Softmax, Trained

I. Introduction:

Character recognition is a machine that's an artwork of detecting, segmenting and identifying characters from picture. An remaining objective of hand written character popularity is to simulate the human reading skills simply so the pc can examine understand edit and paintings as human do with text. Handwriting recognition has been one of

the principal charming and difficult research regions in discipline of image processing and sample recognition within the latest years. HCR contributes immensely to the advancement of automation process and thereby improves the interface among man and machine in numerous packages. several research works are specializing in new techniques and strategies that would lessen the time interval while supplying higher reputation accuracy Handwriting reputation is that the potential of a machine to obtain and interpret handwritten enter from more than one resources like paper files, pics, touch display screen devices and so forth. spotting the textual content of a record could be useful in many diverse applications like analyzing medical prescriptions, financial institution cheques and different professional documents.

Handwriting reputation are often damaged into kind of exceedingly impartial modules. The purpose of this venture is to fashion hand written character reputation device the usage of neural networks if you want to effectively apprehend a particular individual of type layout using the synthetic Neural network approach. reputation and classificatio n of characters are achieved by Neural community. The most aim of this project is to correctly recognize a selected man or woman of type layout the use of the artificial Neural community method this is helpful in recognizing characters of English . one the various first manner by which computers are endowed with human like competencies is thru the utilization of a neural community. Neural networks are particularly beneficial for solving issues that cannot be expressed as a sequence of steps, together with spotting styles and classifying

II. PROBLEM STATEMENT

To design and develop a model that recognizes and classifies the hand written characters of English alphabets and apply modified gradient descent approach.

III. IMPLEMENTATION DETAILS

1) Image Acquisition: In Image acquisition, the opencv provides the flexibility to capture the image from the camera input feed or through the directory path . The image needs to have a specific format such as JPEG, png etc. In the Opencv the images are converted into multi dimensional arrays.

2) Pre-processing: The pre-processing are a series of operations carried out on an input image. It essentially enhances the input image, thereby making it more suitable for segmentation. This uses opencv which is a free open source library employed in real time image pre-processing. Convolutional Neural Networks(CNN) is the main architecture used for pre-processing. The various tasks performed on the image in the stage of pre-processing are, Binarization: In this process a gray scale image is converted into a binary image by the global thresholding technique. Dilation of edges: The binarized image undergoes the dilation of edges using sobel technique, filling the holes present in image and dilation are the operations that are performed in the last two stages which produce the pre-processed image which is suitable for segmentation.

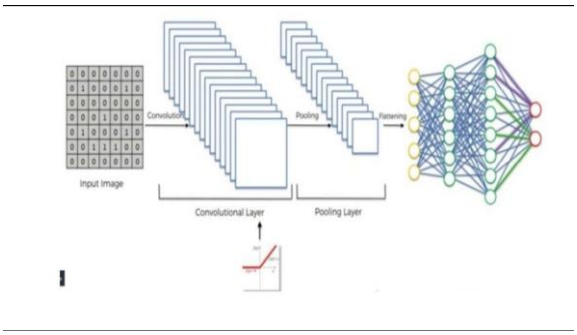

Fig 1: Preprocessing Technique

3) Segmentation: In the segmentation stage, the sequence of characters in an image has to be decomposed into sub-images of individual character. The input image that is pre-processed is segmented into isolated characters using labeling process which assigns a number to each character. Therefore the information about the number of characters in the image is given by the labeling process.

4) Classification and Recognition: The decision making part of the recognition system is the classification stage .In the classification stage output of CNN is fed as an input to the ANN. The dataset is trained by the ANN model. It has two algorithms ie, feed forward neural network(FFNN) and back propagation(BP). The dataset is trained by the FFNN and error minimization through the BP. The neural classifier consists of two hidden layers besides an input layer and an output layer . Input layer makes use of linear activation function. The hidden layers and output layer use sigmoid activation function.

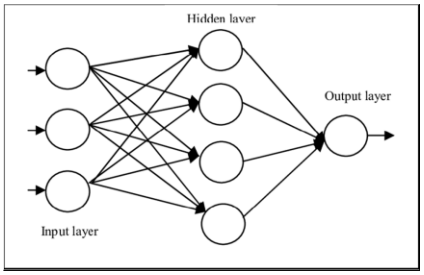

Fig2 : ANN Architecture

5) Post-Processing: This is the final stage of the recognition system. In this the recognized characters are printed correspondingly in the structured text form by calculating the equivalent ASCII value with the help of recognition index of the test samples .

IV. Training and Recognition Techniques:

Many image analysis program make considerable use of character recognition systems. The features of stored character pictures are compared to the features of the character to be identified in a character recognition system. I/P characters are identified as that particular stored character when maximal matching is found. The study of how robots can examine their surroundings, learn to discern patterns of interest from their backgrounds, and make appropriate and acceptable decisions is known as pattern recognition.[2]

V. Softmax Regression Technique:

Softmax regression (or multinomial logistic regression) is a generalization of logistic regression to the case where we want to handle multiple classes. In logistic regression we assumed that the labels were binary: $y(i)\in\{0,1\}$y(i)∈{0,1}. We used such a classifier to distinguish between two kinds of hand-written digits. Softmax regression allows us to handle $y(i)\in\{1,\ldots,K\}$y(i)∈{1,…,K} where KK is the number of classes.

Recall that in logistic regression, we had a training set $\{(x(1),y(1)),\ldots,(x(m),y(m))\}$ of mm labeled examples, where the input features are $x(i)\in Rn$x(i)∈ℜn. With logistic regression, we were in the binary classification setting, so the labels were $y(i)\in\{0,1\}$y(i)∈{0,1}. Our hypothesis took the form:

$h\theta(x)=11+\exp(-\theta\top x)$,

and the model parameters θθ were trained to minimize the cost function

$J(\theta)=-[\sum i=1 m y(i)\log h\theta(x(i))+(1-y(i))\log(1-h\theta(x(i)))]$J(θ)=−[∑i=1my(i)log hθ(x(i))+(1−y(i))log (1−hθ(x(i)))]

In the softmax regression setting, we are interested in multi-class classification (as opposed to only binary classification), and so the label yy can take on KK different values, rather than only two. Thus, in our training set $\{(x(1),y(1)),\ldots,(x(m),y(m))\}$, we now have that $y(i)\in\{1,2,\ldots,K\}$y(i)∈{1,2,…,K}. (Note that our convention will be to index the classes starting from 1, rather than from 0.) For example, in the MNIST digit recognition task, we would have $K=10$K=10 different classes.

Given a test input xx, we want our hypothesis to estimate the probability that $P(y=k|x)$P(y=k|x) for each value of $k=1,\ldots,K$k=1,…,K. I.e., we want to estimate the probability of the class label taking on each of the KK different possible values. Thus, our hypothesis will output a KK-dimensional vector (whose elements sum to 1) giving us our KK estimated probabilities. Concretely, our hypothesis $h\theta(x)$hθ(x) takes the form:

$h\theta(x)=\begin{bmatrix} P(y=1|x;\theta) \\ P(y=2|x;\theta) \\ \vdots \\ P(y=K|x;\theta) \end{bmatrix}=1\sum Kj=1\exp(\theta(j)\top x)\begin{bmatrix} \exp(\theta(1)\top x) \\ \exp(\theta(2)\top x) \\ \vdots \\ \exp(\theta(K)\top x) \end{bmatrix}$hθ(x)=[P(y=1|x;θ)P(y=2|x;θ)⋮P(y=K|x;θ)]=1∑j=1Kexp (θ(j)⊤x)[exp (θ(1)⊤x)exp (θ(2)⊤

x):exp(θ(K)⊤x)] Here θ(1),θ(2),…,θ(K)∈Rnθ(1),θ(2),…,θ(K)∈ℜn are the parameters of our model. Notice that the term 1∑Kj=1exp(θ(j)⊤x)1∑j=1Kexp(θ(j)⊤x) normalizes the distribution, so that it sums to one.

For convenience, we will also write θθ to denote all the parameters of our model. When you implement softmax regression, it is usually convenient to represent θθ as a nn-by-KK matrix obtained by concatenating θ(1),θ(2),…,θ(K)θ(1),θ(2),…,θ(K) into columns, so that

θ=⎡⎣⎢⎢|θ(1)||θ(2)||⋯||θ(K)|⎤⎦⎥⎥.θ=[||||θ(1)θ(2)⋯θ(K)||||].

VI. Model

```
# RNNs.
    x = keras.layers.Bidirectional(
        keras.layers.LSTM(128, return_sequences=True, dropout=0.10)
    )(x)
    x = keras.layers.Bidirectional(
        keras.layers.LSTM(64, return_sequences=True, dropout=0.10)
    )(x)
```

Training: Train the Model for 10 Epoch using the dropout 0.10

Epoch 1/10

1357/1357 [==============================] - ETA: 0s - loss: 13.4155Mean edit distance for epoch 1: 20.3619

1357/1357 [==============================] - 836s 611ms/step - loss: 13.4155 - val_loss: 11.6365

Epoch 2/10

1357/1357 [==============================] - ETA: 0s - loss: 10.5208Mean edit distance for epoch 2: 20.0552

1357/1357 [==============================] - 823s 606ms/step - loss: 10.5208 - val_loss: 9.4642

Epoch 3/10

1357/1357 [==============================] - ETA: 0s - loss: 8.7705Mean edit distance for epoch 3: 19.6905

1357/1357 [==========] - 819s 604ms/step - loss: 8.7705 - val_loss: 7.5980

Epoch 4/10

1357/1357 [==========] - ETA: 0s - loss: 6.9062Mean edit distance for epoch 4: 18.7937

1357/1357 [==========] - 818s 603ms/step - loss: 6.9062 - val_loss: 5.6208

Epoch 5/10

1357/1357 [==========] - ETA: 0s - loss: 5.4873Mean edit distance for epoch 5: 18.3694

1357/1357 [==========] - 816s 601ms/step - loss: 5.4873 - val_loss: 4.4773

Epoch 6/10

1357/1357 [==========] - ETA: 0s - loss: 4.6234Mean edit distance for epoch 6: 18.1260

1357/1357 [==========] - 828s 610ms/step - loss: 4.6234 - val_loss: 3.8309

Epoch 7/10

1357/1357 [==========] - ETA: 0s - loss: 4.0337Mean edit distance for epoch 7: 17.9453

1357/1357 [==========] - 844s 622ms/step - loss: 4.0337 - val_loss: 3.3783

Epoch 8/10

1357/1357 [==========] - ETA: 0s - loss: 3.6235Mean edit distance for epoch 8: 17.9265

1357/1357 [==========] - 824s 607ms/step - loss: 3.6235 - val_loss: 3.1683

Epoch 9/10

1357/1357 [==========] - ETA: 0s - loss: 3.3208Mean edit distance for epoch 9: 17.7695

1357/1357 [==========] - 827s 609ms/step - loss: 3.3208 - val_loss: 2.8370

Epoch 10/10

1357/1357 [==============================] - ETA: 0s - loss: 3.1065Mean edit distance for epoch 10: 17.7347

1357/1357 [==============================] - 826s 609ms/step - loss: 3.1065 - val_loss: 2.7254

Train the Model for 10 Epoch using the dropout 0.10		
Epoch	**Mean Edit Distance**	**Loss**
Epoch 1	20.361	13.41
Epoch 2	20.0552	10.25
Epoch 3	19.6905	8.77
Epoch 4	18.7937	6.9
Epoch 5	18.3694	5.62
Epoch 6	18.126	4.62
Epoch 7	17.9453	4.03
Epoch 8	17.9265	3.62
Epoch 9	17.7695	3.32
Epoch 10	17.7347	3.1

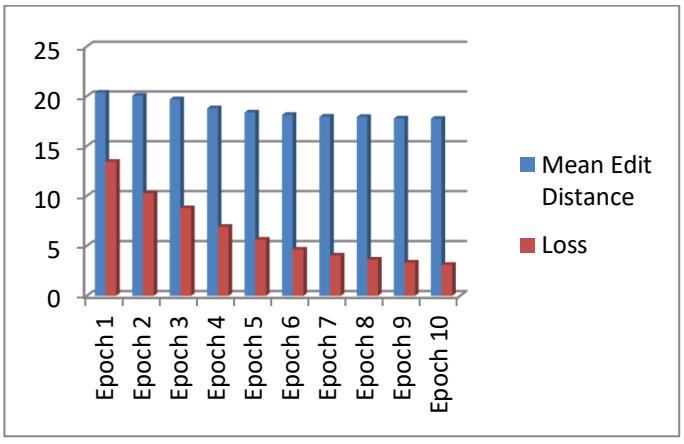

Fig 3: Training Model for 10 Epoch using the dropout 0.10

```
# RNNs.
  x = keras.layers.Bidirectional(
    keras.layers.LSTM(128, return_sequences=True, dropout=0.75)
  )(x)
```

```
    x = keras.layers.Bidirectional(
        keras.layers.LSTM(64, return_sequences=True, dropout=0.75)
    )(x)
```

Epoch 1/10

1357/1357 [==========] - ETA: 0s - loss: 13.9199Mean edit distance for epoch 1: 20.6633

1357/1357 [==========] - 895s 653ms/step - loss: 13.9199 - val_loss: 12.9479

Epoch 2/10

1357/1357 [==========] - ETA: 0s - loss: 11.9586Mean edit distance for epoch 2: 20.4318

1357/1357 [==========] - 824s 607ms/step - loss: 11.9586 - val_loss: 11.7130

Epoch 3/10

1357/1357 [==========] - ETA: 0s - loss: 10.8047Mean edit distance for epoch 3: 20.2206

1357/1357 [==========] - 819s 603ms/step - loss: 10.8047 - val_loss: 10.3421

Epoch 4/10

1357/1357 [==========] - ETA: 0s - loss: 10.0148Mean edit distance for epoch 4: 20.0254

1357/1357 [==========] - 823s 606ms/step - loss: 10.0148 - val_loss: 9.4942

Epoch 5/10

1357/1357 [==========] - ETA: 0s - loss: 9.4024Mean edit distance for epoch 5: 19.8834

1357/1357 [==========] - 815s 601ms/step - loss: 9.4024 - val_loss: 8.7701

Epoch 6/10

1357/1357 [==========] - ETA: 0s - loss: 8.8319Mean edit distance for epoch 6: 19.6930

1357/1357 [==========] - 814s 600ms/step - loss: 8.8319 - val_loss: 7.9753

Epoch 7/10

1357/1357 [==========] - ETA: 0s - loss: 8.3369Mean edit distance for epoch 7: 19.5514

1357/1357 [==========] - 814s 600ms/step - loss: 8.3369 - val_loss: 7.4355

Epoch 8/10

1357/1357 [==========] - ETA: 0s - loss: 7.8927Mean edit distance for epoch 8: 19.4093

1357/1357 [==========] - 816s 601ms/step - loss: 7.8927 - val_loss: 7.0145

Epoch 9/10

1357/1357 [==========] - ETA: 0s - loss: 7.5309Mean edit distance for epoch 9: 19.1177

1357/1357 [==========] - 815s 600ms/step - loss: 7.5309 - val_loss: 6.4622

Epoch 10/10

1357/1357 [==========] - ETA: 0s - loss: 7.1799Mean edit distance for epoch 10: 19.1189

1357/1357 [==========] - 815s 600ms/step - loss: 7.1799 - val_loss: 6.3740

Train the Model for 10 Epoch using the dropout 0.75		
Epoch	Mean Edit Distance	Loss
Epoch 1	20.66	13.91
Epoch 2	20.43	11.95
Epoch 3	20.22	10.8
Epoch 4	20.02	10.01
Epoch 5	19.88	9.4
Epoch 6	19.69	8.83
Epoch 7	19.55	8.33
Epoch 8	19.4	7.8
Epoch 9	19.11	7.5
Epoch 10	19.11	7.1

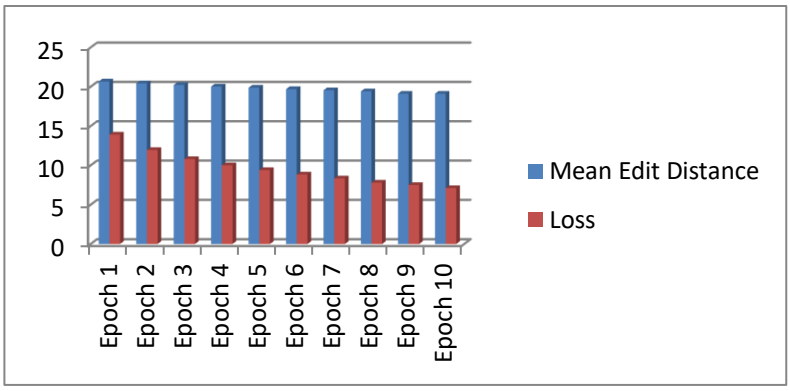

Fig 4: Train the Model for 10 Epoch using the dropout 0.75

Comparison of Train the Model for 10 Epoch using the dropout 0.10 and 0.75

Epoch	Mean Edit Distance for 0.10 dropout	Mean Edit Distance for 0.75 dropout	Loss for dropout 0.10	Loss for dropout 0.75
Epoch 1	20.361	20.66	13.41	13.91
Epoch 2	20.0552	20.43	10.25	11.95
Epoch 3	19.6905	20.22	8.77	10.8
Epoch 4	18.7937	20.02	6.9	10.01
Epoch 5	18.3694	19.88	5.62	9.4
Epoch 6	18.126	19.69	4.62	8.83
Epoch 7	17.9453	19.55	4.03	8.33
Epoch 8	17.9265	19.4	3.62	7.8
Epoch 9	17.7695	19.11	3.32	7.5
Epoch 10	17.7347	19.11	3.1	7.1

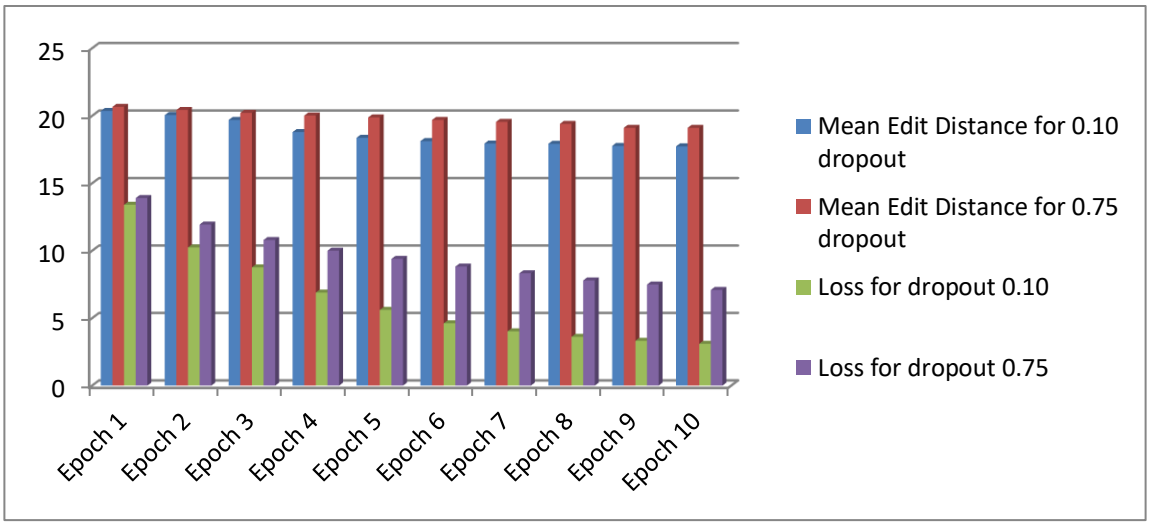

Fig 6: Comparison of Train the Model for 10 Epoch using the dropout 0.10 and 0.75

References:

[1] Yang Zong-chang (2005 – 2013) Establishing Structure For Artificial Neuralnetworks Based-On Fractal Journal of Theoretical and Applied Information Technology 10th March 2013. Vo 49 No.1 © JATIT & LLS.

[2] Selvi, P.P.; Meyyappan, T., (21-22 Feb. 2013) Recognition of Arabic numerals with grouping and ungrouping using back propagation neural network, Pattern Recognition, Informatics and Mobile Engineering (PRIME), 2013 International Conference on , vo, no., pp.322, 327.

[3] Sahu, N.; Raman, N.K., (22-23 March 2013) An efficient handwritten Devnagari character recognition system using neural network, Automation, Computing, Communication, Control and Compressed Sensing (iMac4s), 2013 International Multi-Conference on , vo, no., pp.173,177.

[4] Nguang Sing Ping; Yusoff, M.A., (12-14 June 2012) Application of 13-point feature of skeleton to neural networks-based character recognition, Computer & Information Science (ICCIS), 2012 International Conference on, vo1, no., pp. 447, 452.

[5] Pradeep, J.; Srinivasan, E.; Himavathi, S., (Oct. 30 2012-Nov. 2 2012.) Performance analysis of hybrid feature extraction technique for recognizing English handwritten characters, Information and Communication Technologies (WICT), 2012 World Congress on , vol., no., pp.373,377.

[6] Budiwati, S.D.; Haryatno, J.; Dharma, E.M., (17-19 July 2011) Japanese character (Kana) pattern recognition application using neural network, Electrical Engineering and Informatics (ICEEI), 2011 International Conference on , vo, no., pp.1,6 17-19.