**HANDWRITTEN CHARACTER RECOGNITION OF ROMAN AND DEVANAGRI CHARACTERS WITH DEEP LEARNING**

**Abstract**

Pattern recognition has taken great strides in recent years, but not much work has been done on a combined datasets of different scripts. Lacking work done on a combined dataset of two scripts has endeavoured me to take up this project of classifying the handwritten characters of both Roman and Devanagari scripts in the same project. In this project, we have worked on classification of handwritten characters of Roman and Devanagari scripts. Standard datasets of both the scripts have been combined for training the network. To achieve a superior result for classification, convolutional neural network has been used. Doing things manually in postal mail sorting, bank check processing, form data entry, etc., is quite tedious, and there are greater chances of errors. To overcome these problems, a handwritten character classification system can be used.

**Keywords**

Convolutional Neural Networks, Handwritten Digit Recognition, Artificial Neural Network, State of the Art.

**Introduction**

Recognition is identifying something from the past experiences. Recognising characters from a handwritten document is easy for human beings and this ability can be induced in machines too through deep learning models. Handwritten Character Recognition is nothing but recognizing and interpreting the handwritten characters from sources such as number plates of vehicles, photographs, touch screens, documents from banking field and postal service etc. But since handwriting depends much on the individual and because we do not always write the same character in exactly the same way our network should be able to recognise the pattern, to interpret the handwritten characters. The benefit that Handwritten Character Recognition systems offer is that they can reduce the data entry time and storage space required by documents.

There has been considerable research work done on pattern recognition based on deep learning techniques, and significant progress has been made in this area in the last few years**.** But with the growth in the amount of available handwritten data in various scripts, we can do further investigation by combining datasets of two scripts and trying to achieve a better accuracy**.**

So, here we’ll be using a convolutional neural network, which is a specific type of deep neural network for character recognition purpose.

**Literature Review**

In the last two-three decades, substantial research has been done in the area. Remarkable accuracy has already been achieved through standard neural networks [1-3] but, the real advancement has been brought by CNN which delivers the state-of-the-art performance in this domain [4-6]. CNNs have already given impressive results in offline handwritten character recognition of Arabic language [7]; handwritten Tamil character recognition [8]; handwritten Urdu text recognition [9] and Chinese handwritten text recognition [10].

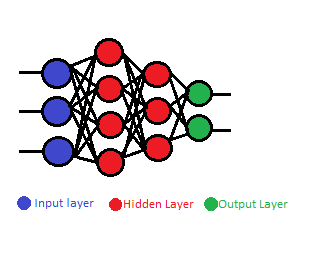
**Neural Networks**

Neural networks try to mimic the human brain. They are similar to biological neural networks that are there in the human brain and work in a similar way. Neurons in the brain are called nodes in a neural network and Synapses in the brain are called weights in a neural network. It is like having multiple signals to a node, and it has a decision making capacity, and the outputs can be one or multiple again, and this output is based on the previous logic and the weight that is assigned. We can always assign weights to the nodes and at the node we can perform some kind of calculation and the output of this would be a calculated weighted value of the set of inputs. So in this way, using calculation, we can expect it to do logic for pattern recognition.

Below is a short description about ANN and CNN and why the latter is a better performer for pattern recognition purposes.

**Standard Neural Network**

Standard Neural Network is a group of multiple neurons or nodes at each layer. It is also called as a **Feed-Forward Neural network** because inputs are processed only in the forward direction.



**Fig 1: Standard ANN.**

The input layer in standard NN accepts the inputs, the hidden layer processes the inputs, and the output layer produces the result. Essentially, each layer tries to learn certain weights.

For doing an image classification using standard NN, the first thing we have to do is to convert a 2D image into a 1D vector before training the model. This results in an increase in the number of trainable parameters drastically and the network also loses the spatial features of the image.

**Convolutional Neural Network**

Convolutional neural networks are one of the most popular models in the deep learning community today. **Filters are the building blocks of CNN and they** are used to extract the relevant features from the input using the convolution operation. CNN extracts features from the raw image in its first layers and classifies the pattern with its last layers.

The major difference between CNN and a traditional Artificial Neural Network (ANN) is that only the last layer of a CNN is fully connected whereas in Standard NN, each neuron is connected to every other neuron. Standard NNs are not suitable for images because these networks lead to over-fitting easily due to the size of the images. Consider an image of size [28x28x3]. If this image has to be passed to a Standard NN model, it must be flattened into a vector of 28x28x3= 2352 rows. Thus there will have to be 2352 weights in its first layer to receive the input. For larger images, more complex vector of weights will be there, which will need a more powerful processor to process.

|  |  |  |
| --- | --- | --- |
|  | **ANN** | **CNN** |
| **Type of Data** | Tabular data, text data | Image data |
| **Parameter Sharing** | No | Yes |
| **Spatial Relationship** | No | Yes |
| **Performance** | ANN is considered less powerful and less accurate than CNN. | CNN is considered more powerful than ANN. |

**Table 1: Comparison between ANN and CNN.**

**Training Dataset**

Dataset taken from [11] was used for roman characters and digits. For devanagri characters, dataset was picked from [12]. The training of the network has been done after converting all the images of both the datasets in .jpg form and then putting around a thousand of these images of each character in a separate folder. So, there are 26 folders which have images of roman characters, 36 folders which have images of devnagri characters and 10 folders which have images of digits.

In total, around 70 thousand sample images have been used. Of which 75% have been used for training the model and the rest for testing. Each sample image is of 28 by 28 pixel.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Number Of Training Images** | **Number Of Testing Images** | **Number Of Folders** |
| **Roman- Script Images** | 20474 | 6825 | 26 |
| **Devanagri- Script Images** | 26313 | 8771 | 36 |
| **Digit Images** | 7101 | 2367 | 10 |

**Table 2: Data used for training and testing.**

Some handwritten characters of the two scripts look very similar and one of the challenges faced in the classification is the distinction of these several similar shaped characters. Some characters have very similar variation between them and this leads to recognition complexity

and the accuracy rate of the recognition system gets reduced. Some of the sets of similar numerals are as in Fig 2.

|  |  |  |
| --- | --- | --- |
| **Numeral ‘0’ and Roman character ‘O’** |  |  |
| **Devanagri character ‘ढ’ and Numeral ‘6’** |  |  |
| **Numeral ‘1’ and Roman character ‘I’** |  |  |

**Fig 2: Examples of some similar shaped characters.**

**Proposed Network Architecture**

layers=[

imageInputLayer([28 28 3], 'Name', 'Input')

convolution2dLayer(3,8, 'Padding', 'same', 'Name', 'Conv\_1')

batchNormalizationLayer('Name', 'BN\_1')

reluLayer('Name','Relu\_1')

maxPooling2dLayer(2, 'Stride', 2, 'Name', 'Maxpool\_1')

convolution2dLayer(3,16, 'Padding', 'same', 'Name', 'Conv\_2')

batchNormalizationLayer('Name', 'BN\_2')

reluLayer('Name','Relu\_2')

maxPooling2dLayer(2, 'Stride', 2, 'Name', 'Maxpool\_2')

convolution2dLayer(3,32, 'Padding', 'same', 'Name', 'Conv\_3')

batchNormalizationLayer('Name', 'BN\_3')

reluLayer('Name','Relu\_3')

maxPooling2dLayer(2, 'Stride', 2, 'Name', 'Maxpool\_3')

convolution2dLayer(3,64, 'Padding', 'same', 'Name', 'Conv\_4')

batchNormalizationLayer('Name', 'BN\_4')

reluLayer('Name','Relu\_4')

maxPooling2dLayer(2, 'Stride', 2, 'Name', 'Maxpool\_4')

fullyConnectedLayer(72,'Name', 'FC')

softmaxLayer('Name', 'Softmax');

classificationLayer('Name', 'Output Classification');

];

The design comprises a sequential CNN architecture with an input shape (28, 28, 3). The first layer is a Convolutional layer followed by a BatchNormalization layer, a ReLU layer and a MaxPooling Layer. Same sequence of layer arrangement is repeated four times with different filter sizes. The main significance of these layers is Feature extraction .

Then there is a Fully Connected Layer, a Softmax layer and a classification layer, which is the final layer.

***Convolutional layer*:**

The convolutional layer is the core building block of a CNN. The layer's parameters consist of a set of learnable filters (or kernels), which have a small receptive field, but extend through the full depth of the input volume.

***Batch Normalization:***

Batch normalization is a technique for training very deep neural networks that standardizes the inputs to a layer for each mini-batch. This has the effect of stabilizing the learning process and dramatically reducing the number of training epochs required to train deep networks.

***ReLU Layer*:**

The Rectified Linear Unit is the most commonly used activation function in deep learning models. The function returns 0 if it receives any negative input, but for any positive value xx it returns that value back. So it can be written as f(x)=max(0,x).

***MaxPooling Layer:***

Max pooling is a pooling operation that selects the maximum element from the region of the feature map covered by the filter. Thus, the output after max-pooling layer would be a feature map containing the most prominent features of the previous feature map.

***Fully Connected Layer*:**

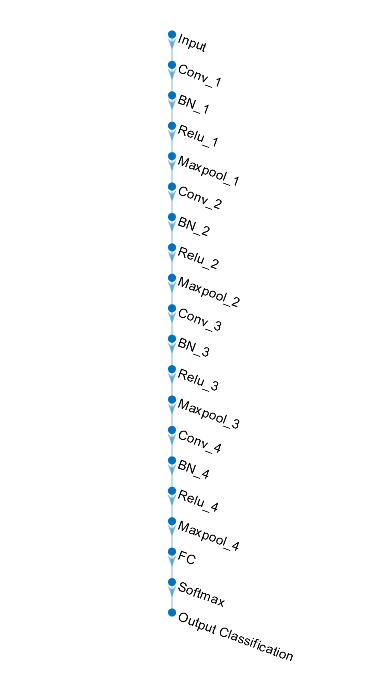
This layer takes the inputs from the feature analysis and applies weights to predict the correct label. The CNN process begins with convolution and pooling, breaking down the image into features, and analyzing them independently. The result of this process feeds into a fully connected neural network structure that drives the final classification decision.

***Softmax Layer*:**

Softmax assigns decimal probabilities to each class in a multi-class problem. Those decimal probabilities must add up to 1.0. This additional constraint helps training converge more quickly than it otherwise would. Softmax is implemented through a neural network layer just before the output layer. The Softmax layer must have the same number of nodes as the output layer.

***Classification Layer*:**

A classification layer computes the cross entropy loss for multi-class classification problems with mutually exclusive classes.

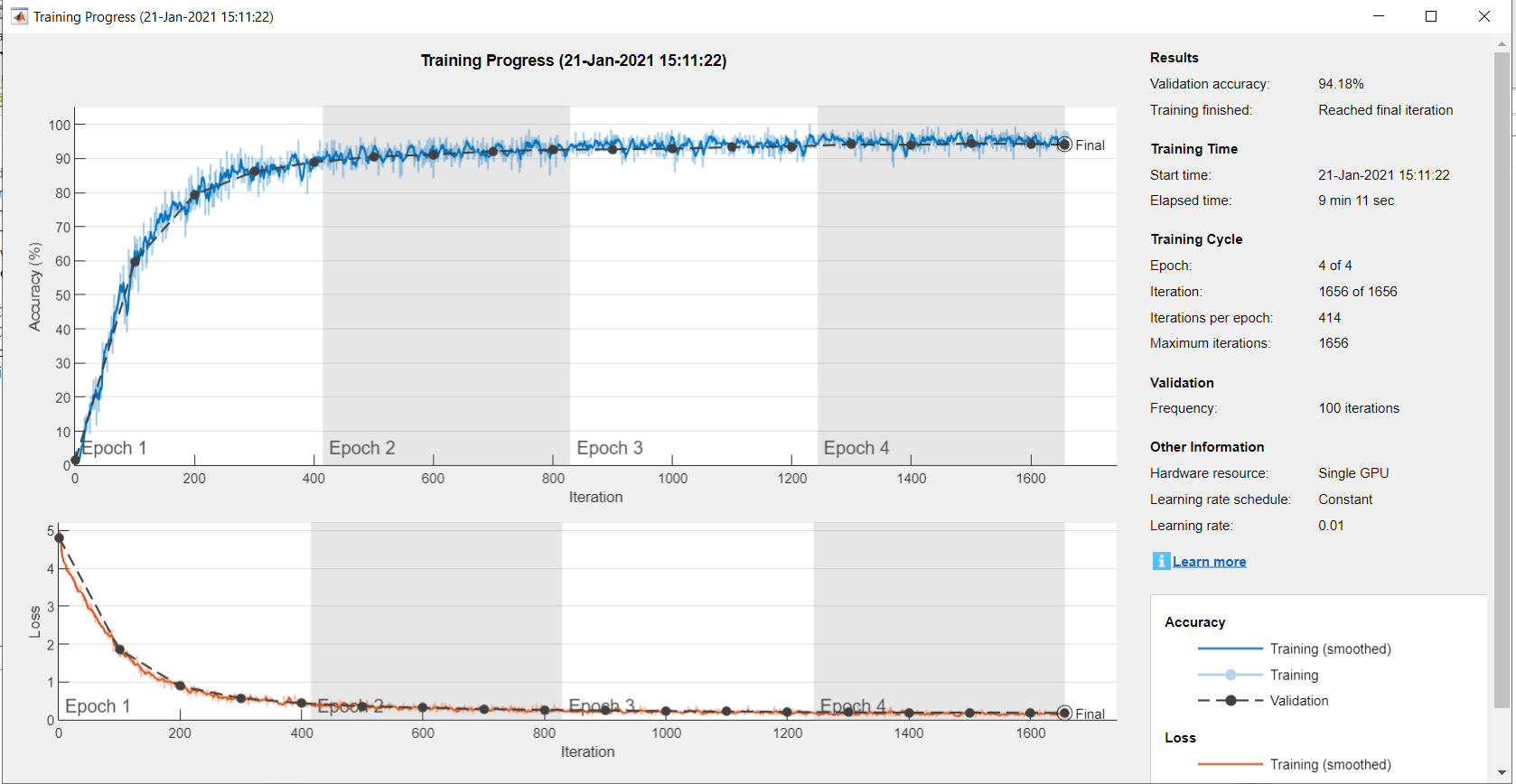
****

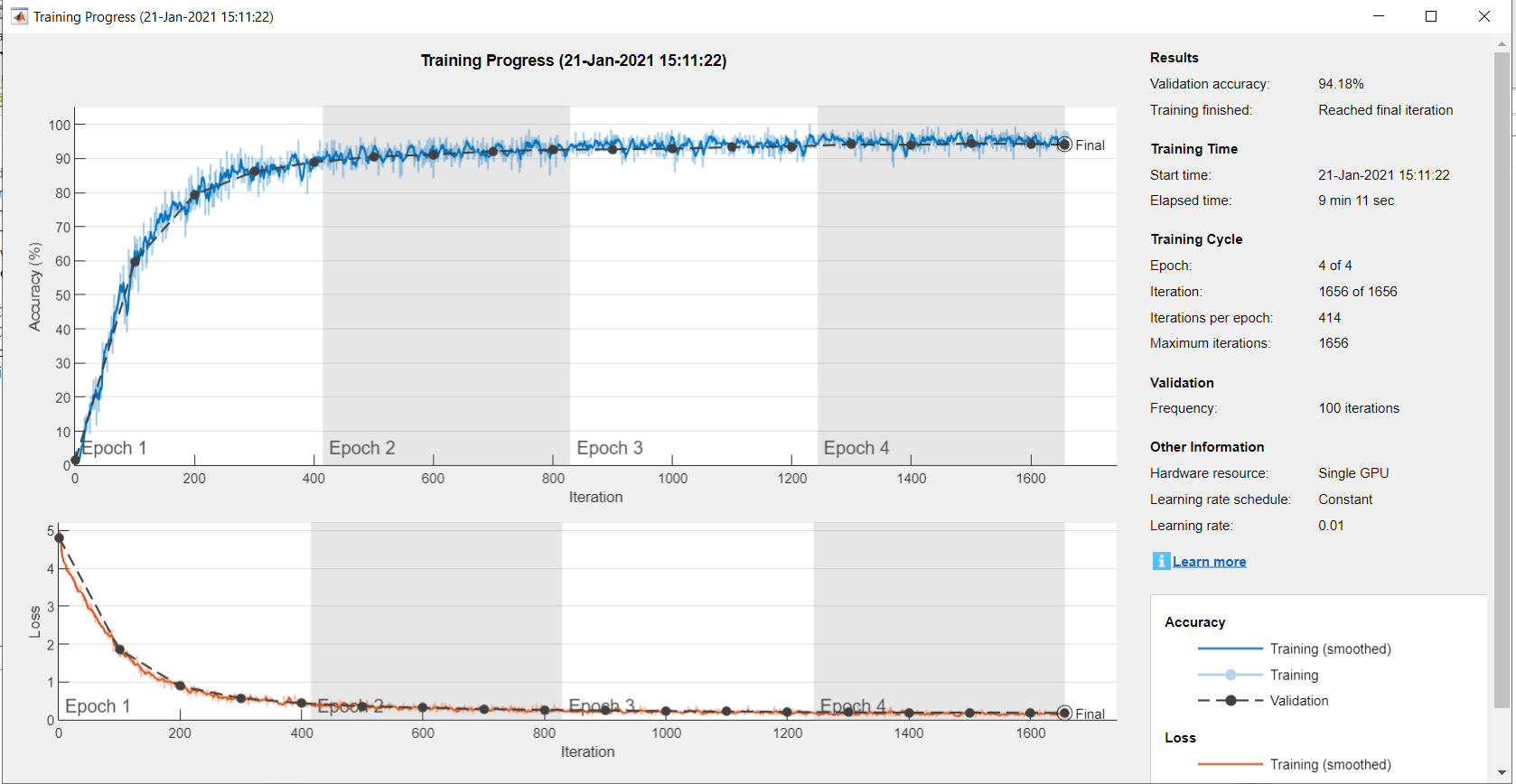
**Fig 3: Network architecture used**

**Experimental results and analysis**

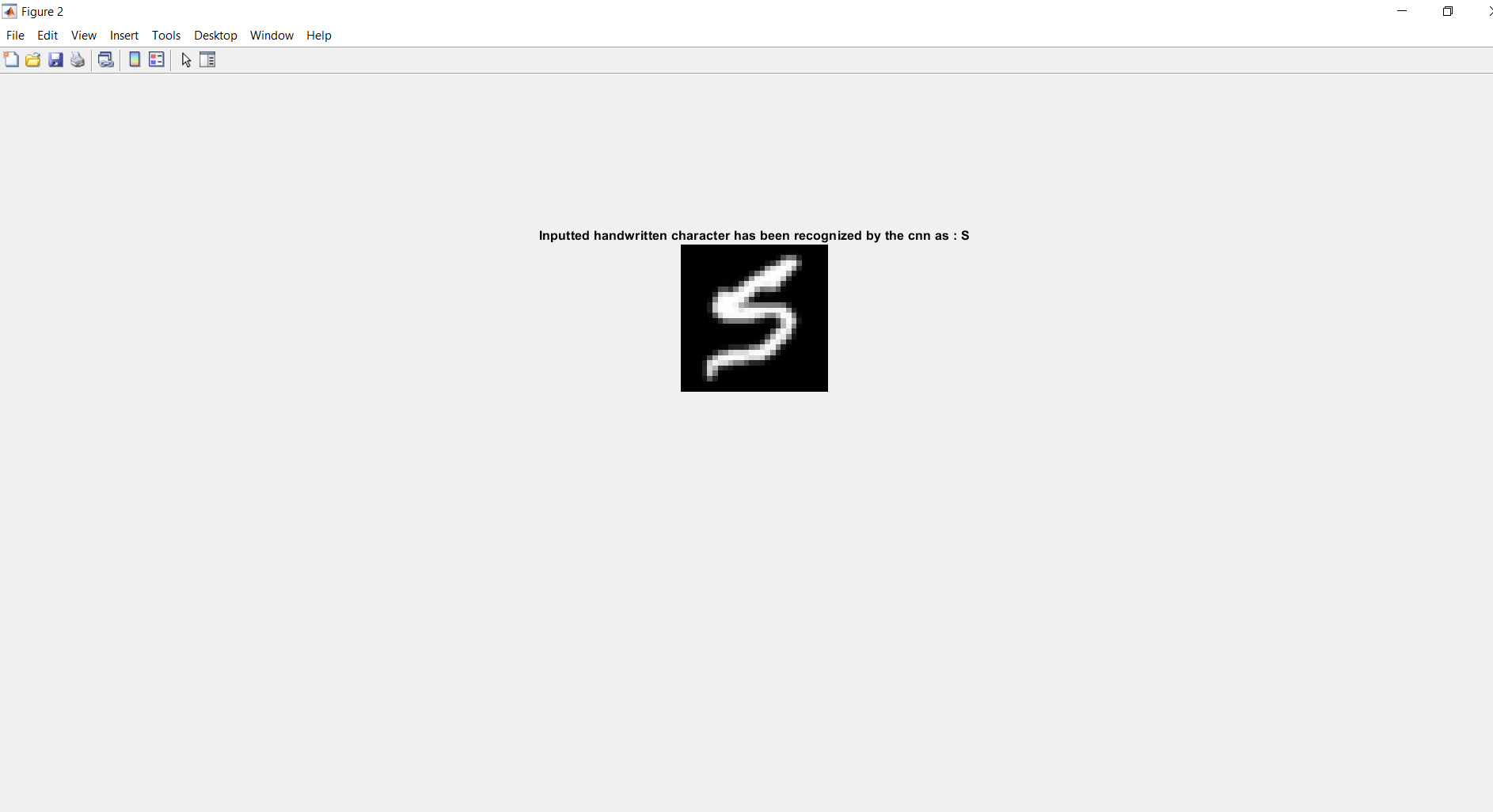
After evaluating the handwritten character classifier on the test images that the Model had not seen before, we got a satisfactory result with an accuracy of 94.18% which is competitive to state of the art results.

All the experiments were performed in MATLAB R2020b on a personal computer with Windows 10, Intel(R) Core(TM) i5-6200U CPU @ 2.30GHz, 12.00 GB memory and NVIDIA 1650 GTX GPU. The dataset involved in training had around 52500 pictures and the dataset involved in testing the model had around 17500 images.

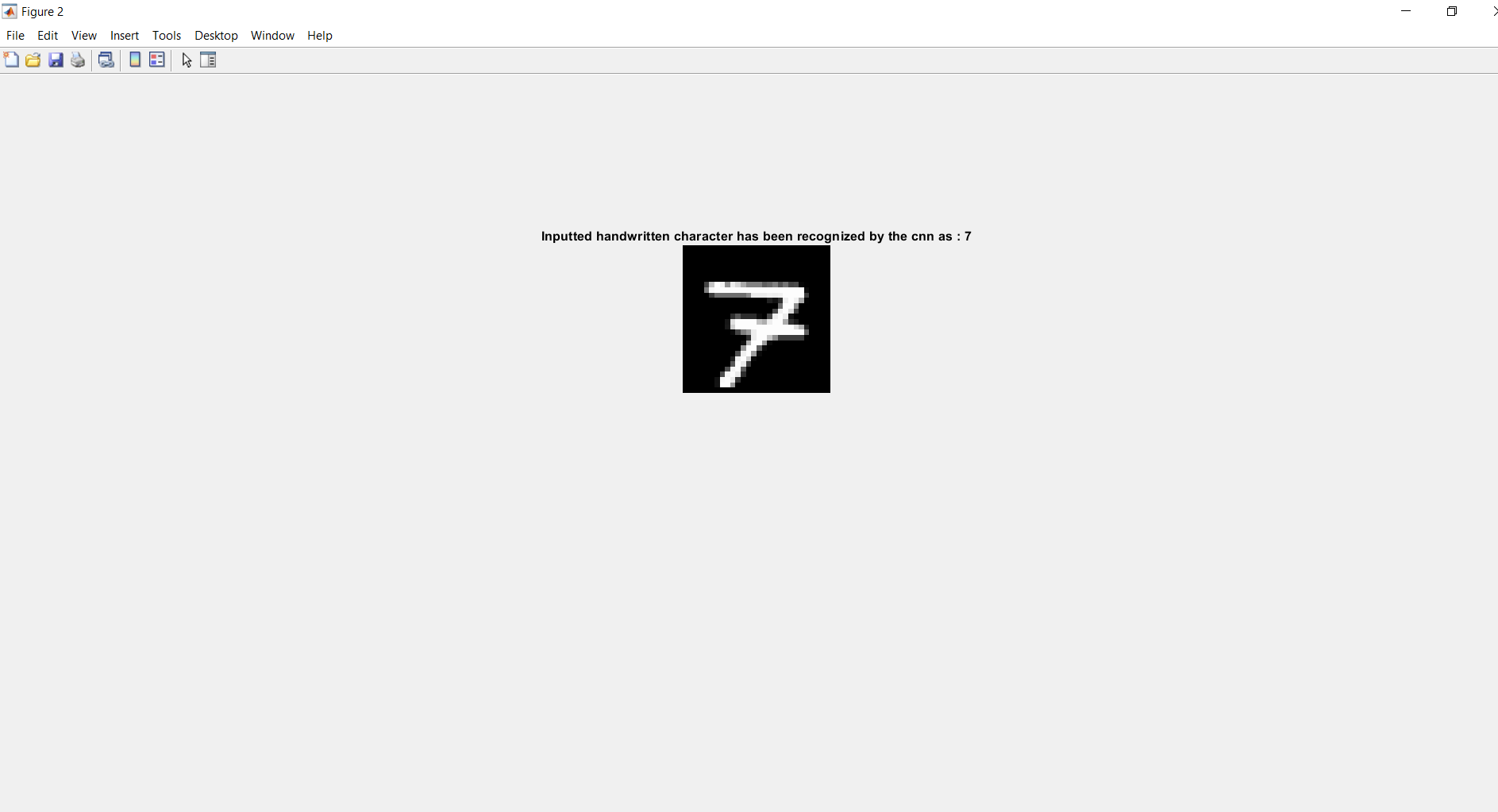




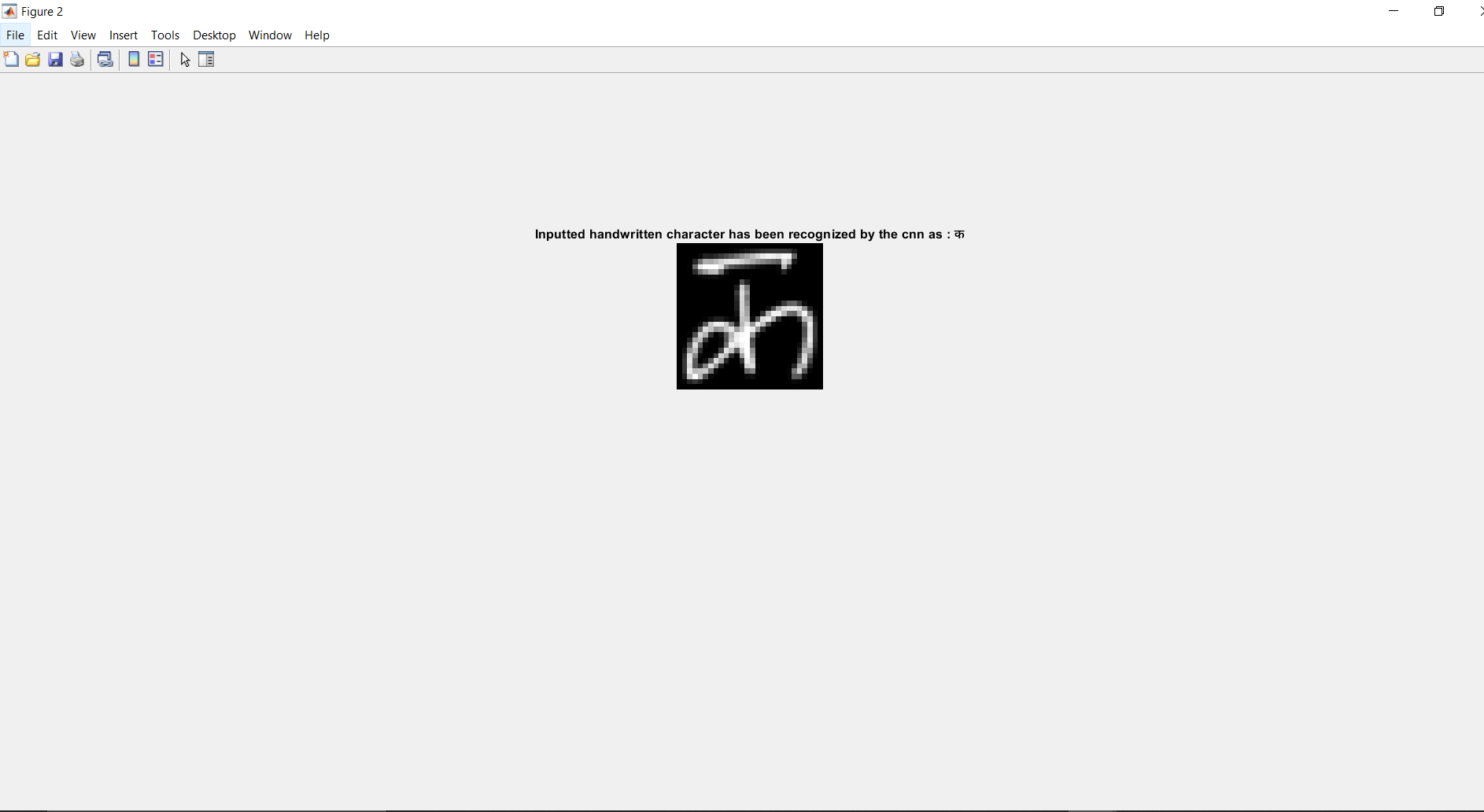
**Fig 4: Analysis of Deep Learning Training Process**



**Fig 5: Screenshot of output screen classifying character ‘S’**

****

**Fig 6: Screenshot of output screen classifying numeral ‘7’**

****

# Fig 7: Screenshot of output screen classifying character ‘क’

**Comparative Study**

Table 3 gives a comparison of Handwritten Character Recognition for different scripts. A brief sketch of pre-processing, feature extraction, classification methods and accuracy for several works are summarized below.

|  |  |  |  |
| --- | --- | --- | --- |
| **Authors** | **Feature Extraction Method** | **Classifier** | **Accuracy** |
| **Sukhpreet Singh[13]** | Gabor filter | SVM | 94.29% |
| **Alvaro Gonzalez[14]** | Gradient feature | KNN | 85.8% |
| **Hassiba Nemmour[15]** | Ridgelet transform | SVM | 84% |
| **Mamatha H R [16]** | Curvelet Transform | KNN | 90.5% |
| **Ashutosh Aggrawal [17]** | Gradient | SVM | 94% |

**Table 3: Comparative study for handwritten character recognition of different scripts.**

A classification system has been proposed by Mr. Sukhpreet[13] Singh for isolated handwritten characters of Gurumukhi script. Gabor Filter based methods have been used for feature extraction and SVM(support vector machine) has been used for classification. An accuracy of 94.29% has been achieved.

A fast and easy method for classification of handwritten characters has been suggested by Mr. Alvaro Gonzalez[14]. This method uses gradient teachnique for feature extraction and KNN for classification. The accuracy achieved is 85.8%.

Analytic and holistic approaches for handwritten Arabic word recognition have been explained by Hassiba Nemmour[15]. Ridgelet transform has been used for feature extraction and SVM(support vector machine) has been used for classification. Recognition efficiency of 84% has been achieved.

Mamatha H R[16] has presented a method for the classification of numerals of Kannada language which is derived from the southern Bramhi script and one of the challenges faced in the classification of the numerals in Bramhi script is the distinction of several similar shaped components in the script. Curvelet transform has been used for feature extraction and KNN(k-nearest neighbors) has been used for classification purpose. A recognition efficiency of 90.5% has been achieved through the proposed method.

Ashutosh Aggarwal[17] has suggested a model for handwritten Devanagari Character classification. Gradient methods have been used for extracting the features. The obtained features are then sent to SVM(support vector machine) for recognition. Recognition efficiency of 94% has been obtained.

Figure 8 can be looked at for comparing our proposed CNN model with other approaches for handwritten character recognition of various scripts. It can be observed that our CNN model performs better than the most other models mentioned in the table in terms of accuracy. Being neural networks with many stacked layers, one atop the other, cnn learn to extract increasingly more and more abstract high-level features. However, it is important to view different algorithms as complimentary to each other.

**Fig 8: Comparison of the accuracy levels of related works on various scripts.**

**Conclusion**

In this experiment we have presented a method for classification of handwritten characters of Roman and Devnagri scripts using convolutional neural network, which can be used to interpret the manually written characters from various sources like messages, bank cheques, papers, pictures, identifying number plates of vehicles, and so forth.

Performance of CNN is dependent upon the choice of hyper-parameters like, number of epochs, kernel size, learning rate, hidden units, hidden layers, etc. These are very important as they control the way an algorithm learns from the data.

So, we have specifically trained a deep convolutional neural network for the classification task and according to the results got, we can suggest that the model performs satisfactory and can be effectively used for recognition purpose.

**References**

1. Choudhary, A.; Rishi, R.; Ahlawat, S. Handwritten numeral recognition using modified BP ANN structure. In Proceedings of the Communication in Computer and Information Sciences (CCIS-133), Advanced Computing, CCSIT 2011, Royal Orchid Central, Bangalore, India, 2–4 January 2011; Springer: Berling/Heildelberg, Germany, 2011; pp. 56–65.
2. Choudhary, A.; Rishi, R.; Ahlawat, S. Handwritten numeral recognition using modified BP ANN structure. In Proceedings of the Communication in Computer and Information Sciences (CCIS-133), Advanced Computing, CCSIT 2011, Royal Orchid Central, Bangalore, India, 2–4 January 2011; Springer: Berling/Heildelberg, Germany, 2011; pp. 56–65.
3. Choudhary, A.; Rishi, R. Improving the character recognition efficiency of feed forward bp neural network. Int. J. Comput. Sci. Inf. Technol. 2011, 3, 85–96. [CrossRef]
4. Badrinarayanan, V.; Kendall, A.; Cipolla, R. SegNet: A Deep convolutional encoder-decoder architecture for image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. 2017, 39, 2481–2495. [CrossRef]
5. Sueiras, J.; Ruiz, V.; Sanchez, A.; Velez, J.F. Offline continuous handwriting recognition using sequence to sequence neural networks. Neurocomputing. 2018, 289, 119–128. [CrossRef]
6. Liang, T.; Xu, X.; Xiao, P. A new image classification method based on modified condensed nearest neighbor and convolutional neural networks. Pattern Recognit. Lett. 2017, 94, 105–111. [CrossRef]
7. Boufenar, C.; Kerboua, A.; Batouche, M. Investigation on deep learning for off-line handwritten Arabic character recognition. Cogn. Syst. Res. 2018, 50, 180–195. [CrossRef]
8. Kavitha, B.; Srimathi, C. Benchmarking on offline Handwritten Tamil Character Recognition using convolutional neural networks. J. King Saud Univ. Comput. Inf. Sci. 2019. [CrossRef]
9. Ahmed, S.; Naz, S.; Swati, S.; Razzak, M.I. Handwritten Urdu character recognition using one-dimensional BLSTM classifier. Neural Comput. Appl. 2019, 31, 1143–1151. [CrossRef]
10. Liu, C.; Yin, F.; Wang, D.; Wang, Q.-F. Online and offline handwritten Chinese character recognition: Benchmarking on new databases. Pattern Recognit. 2013, 46, 155–162. [CrossRef]
11. Cohen, G., Afshar, S., Tapson, J., & van Schaik, A. (2017). EMNIST: an extension of MNIST to handwritten letters.
12. Acharya S., Pant A.K., Gyawali P.K.,”Deep Learning based large scale handwritten Devanagari Character Recognition” Proceeding of 9th International Conference on Software, Knowledge, Information Management & Applications. In Press.