International Journal of Advanced Science and Technology
Vol. 29, No. 6, (2020), pp. 58-66

Hindi Handwritten Character Recognition using CNN

B. Baranidharan[a], Apoorva Kandpal[b], Adhiraj Chakravorty[c]
[a]*Associate Professor,*
[a,b,c]*Department of CSE,*
SRM Institute of Science and Technology, Chennai

Abstract

This paper focuses on the task of recognizing handwritten Hindi characters using a Convolutional Neural Network (CNN) based Deep Learning model. The recognized characters can then be stored digitally in the computer or used for other purposes. The dataset used is obtained from the UC Irvine Machine Learning Repository which contains 92,000 images divided into training (85%) and test set (15%). It contains different forms of handwritten Devanagari characters written by different individuals which can be used to train and test handwritten text recognizers and perform writer identification and verification experiments. The model is implemented using Keras libraries on top of a TensorFlow backend. It contains four CNN layers followed by three fully connected layers for recognition. Grayscale handwritten character images are used as input. Filters are applied on the images to extract different features at each layer. This is done by the Convolution operation. The two other main operations involved are Pooling and Flattening. The output of the CNN layers are fed to the fully connected layers. Finally, the chance or probability score of each character is determined and the character with the highest probability score is shown as the output. A recognition accuracy of 98.94% is obtained. Similar models exist for the purpose, but the proposed model achieved a better performance and accuracy than some of the earlier models.

Keywords: *CNN, handwritten Hindi characters, TensorFlow, Keras, Devanagari*

Introduction

A. Overview

Handwritten text recognition technology is quite helpful and needed in today's world. Storing and accessing physical data is quite difficult and have chances of getting damaged or lost. Also, the physical data formation is prone to errors. However, if the text is present in a digital format, then error scanning mechanisms and autocorrect tools can help in storing data correctly and efficiently. In the pattern recognition area, this is one of the most challenging tasks. One of the most important advantages of storing text digitally is that it can be accessed from any place. You don't have to be at the place where the data is stored. This technology has made storing, accessing and analyzing data much easier.

There are many important applications of this system. This includes bank check analysis, reading addresses, text restoration, and preserving very old documents by reading and storing them etc.

One of the most popular examples of such a system is the Google lens. This project aims to develop such a system that takes in input in the form of image and shows what is written in that image as output. The key mechanism involved is the use of a Convolutional Neural Network (CNN).

B. Deep Learning

Deep learning is the model which aims to mimic the functioning of the human brain. As easy as it sounds, it is difficult to achieve this feat as the human brain has a very complex structure. Here, the concept of neural networks comes into the picture. It is a type of network which is like the way neurons are interconnected in the human body. Each neuron is a node devised in a network which performs a certain function and passes signals.

Artificial Neural Network is an interconnection of various layers. These layers are classified into three types:

Copyright ⓒ 2020 SERSC

1. Input Layer

2. Hidden Layer(s)

3. Output layer

The input layer provides the input to the neural network, as is clear from the name.

The hidden layer consists of multiple nodes that take in inputs from the previous layer along with weights (parameters). Then to these, the bias value is added and these nodes with the help of specific activation functions provide an output.

Finally, the output layer shows the output.

To train a neural network, the following steps are required:

1. First, obtain an output using forward propagation (going from the input layer to output layer).

2. Compare the obtained result with actual result, i.e., calculate the cost function.

3. Using Backpropagation, update the weights (using learning rate) to minimize the cost function.

4. Finally, obtain the required result/model.

Images or videos are the main input format given to this type of model for analysis. It searches for the important features and these features are used for classification. The prime application of CNNs is in image recognition and classification, natural language processing, recommender system etc.

There are certain steps involved in the CNN that helps in visual input analysis, which include:

1. *Convolution Operation*: This process is used to obtain the feature map. The input image is taken in the form of a matrix with pixel values. Here, a feature detector or filter is used that helps in reducing the size of the image and in providing the important features. After applying the convolution operation, what we are left with is called the feature map. Different filters are used to obtain different feature maps. To increase non-linearity in the image, Rectifier function is applied to the feature map.

2. *Pooling*: Pooling helps maintain spatial invariance or deal with other kind of distortions in the process. This means that, if we are trying to perform image recognition, like checking if the image contains a dog, there is a possibility that the image might not be straight (maybe tilted), or there is a difference in texture, or the object size in the image is small etc. These occurrences should not let the model provide incorrect output. Pooling helps in preserving the important features. There can be different types of pooling like min pooling or max pooling. After applying max pooling on the feature map, the maximum values from the clusters are used. What we obtain is called a pooled feature map. Here the size is reduced, features are preserved and distortions are dealt with.

3. *Flattening*: In this step, the matrix (pooled feature map) is converted into a vector. This vector is given to the fully connected layer for further processing.

Copyright ⓒ 2020 SERSC

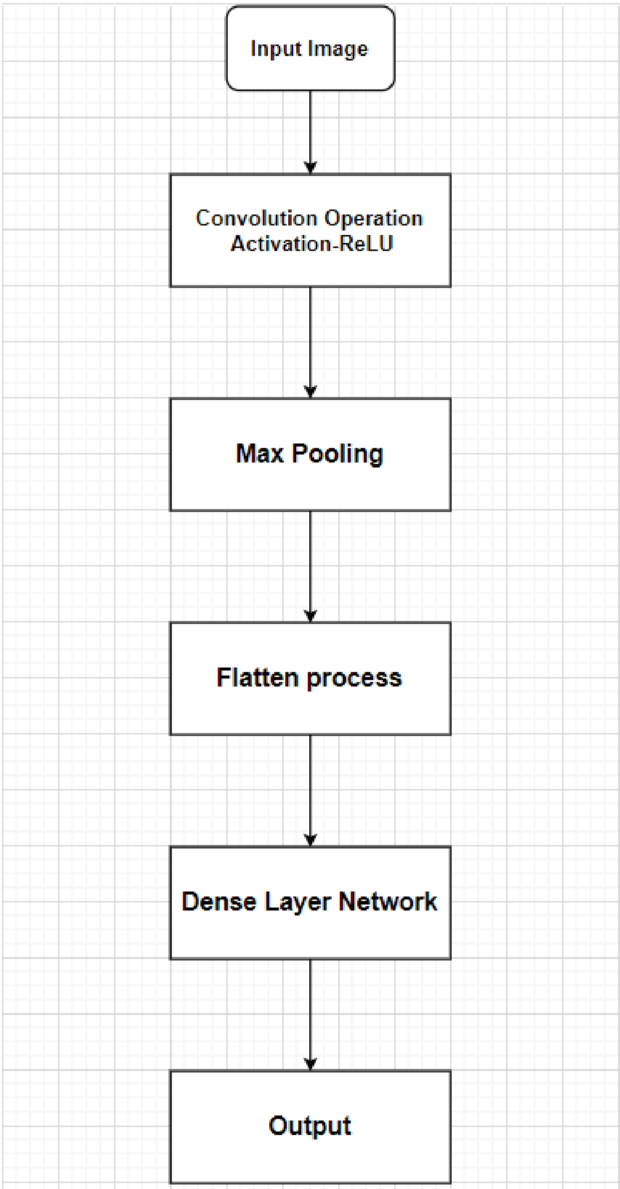


Figure 1: CNN architecture

C. Fully Connected Network

After the input passes through the CNN layers, the flattened feature map, i.e., the vector obtained is given as input to the fully connected layer. The pooled output of the convolution layers is flattened into a vector so that it can be processed by the fully connected layer. Fig. 1 depicts the sequence flow of a general CNN model.

Literature Survey

Ripal, Palak, et al.: In this paper, the main objective is to compare different neural network models for hand written character recognition. The paper also talks about the suitable hyperparameter selection that includes learning rate, number of epochs, batch sizes etc. Finally, the accuracies of these different models are evaluated and compared.

Gunjan, Sushma, et al.: This paper talks about the challenges in the field of pattern and image classification. This includes problems due to cursive writing and curves in characters, different orientation of objects, clarity in images etc. An offline handwritten Hindi character recognition system using a neural network is presented. The basic flow model includes scanning, image pre-processing, feature extraction and finally recognition. Preprocessing involves dimension processing, size and contrast retuning etc. Feature extraction involves obtaining the essential features that will help in the recognition process. Pixel evaluation is done for features. Backpropagation is used as the training algorithm. The proposed model achieves 93% accuracy.

Abhishek, Vikram, et al. This paper explains about the challenges involved in character recognition mainly due to the curves in cursive text. The process of how OCR recognizes text is elaborated. The paper talks about different approaches for text recognition and how different models achieves their better accuracy. Models like neural networks, SVM, random forest etc. have been explored here.

Mahesh, Sumit, et al. The paper first talks about the applications of handwritten text recognition such as text readers for the blind, robotics framework etc. The proposed model in this paper makes use of deep Convolutional Neural Networks for recognition of handwritten Devanagari characters. ISIDCHAR database has been used here provided by (Information Sharing Index) ISI, Kolkata and V2DMDCHAR database with six different architectures of DCNN to evaluate the performance and also investigate the use of six recently developed adaptive gradient methods. To get higher accuracy, layer wise DCNN structure is involved. Layer-wise DCNN is better than standard CNN.

Sujatha, Lalitha, et al. The paper talks about the variations that are encountered in handwritten text when different people write characters of the same language in different styles. This variation can be dealt with by converting text into a digital format. Convolutional Neural Networks is used for efficient pattern and text recognition. Less number of native language models are present. Languages like Hindi and Telugu are difficult to process. Here, the proposed model focuses on increasing the accuracy of the trained model for better recognition.

Sheng Wang, et al. This paper talks about object detection in computer vision research. Some of the most popular methods for pattern and image classification in computer vision have been reviewed here. Highly researched models are analyzed for their performance. Feature extraction methods are categorized into gradient-based and edge-based methods, depending on the low-level features they use. Extended definitions of gradient and edge-based learning are provided here.

Proposed Methodology

Data set

The dataset for the model is obtained from the UC Irvine Machine Learning Repository. In this dataset there are 46 classes of characters. This includes Devanagari alphabets and digits. For each class, around 2000 images are present. Training set is 85% of the data and test set is 15%.

The data type is Grayscale image. The images are in .png format of resolution 32x32. The actual characters are centered within 28x28, and a padding of 2 pixels is added on all four sides.

Architecture

The main architecture of this model involves the use of Convolutional Neural Network (CNN) layers for image processing and feature extraction. Fully connected layers are added for recognition purpose. Fig.2 depicts the general architecture of the proposed CNN model.

The following phases is involved in the system:

A. Image Preprocessing

The input data i.e. the input image is given. It is first resized to 32x32 so that it is easier for the model to predict. Data augmentation is used to increase the size of the training set by performing slight alterations in the training images. Only grayscale images are used to train the model.

Copyright ⓒ 2020 SERSC

B. CNN layers

After the image is preprocessed, it is fed to the CNN layers. As described earlier, CNN layers are trained in such a way that they are capable of extracting the essential features in the image. Four layers of CNN are used. In each layer, three major operations are performed. The first is the application of the filter kernel. For all the layers, the filter kernel is of size 3x3. The first two layers involve 32 kernels each and the third and fourth layers involve 64 kernels each. The second operation is the application of ReLU activation function. This helps achieve non-linearity. At the end, pooling is applied which reduces the size of the feature maps.

C. Fully Connected Layers

After the image is processed by the CNN layers and the essential features are extracted, the flattened output from the final CNN layer is given to three fully connected layers. The first layer consists of 128 units, with all having Rectified Linear Unit (ReLU) as the activation function. After that, batch normalization is applied. The second layer consists of 64 units with ReLU followed by batch normalization. The final layer is the output layer which consists of 46 units that represent each class of output. The SoftMax activation function is used in the final layer.

The chance or probability scores of all the characters are produced and the one with the maximum score is provided as the output.

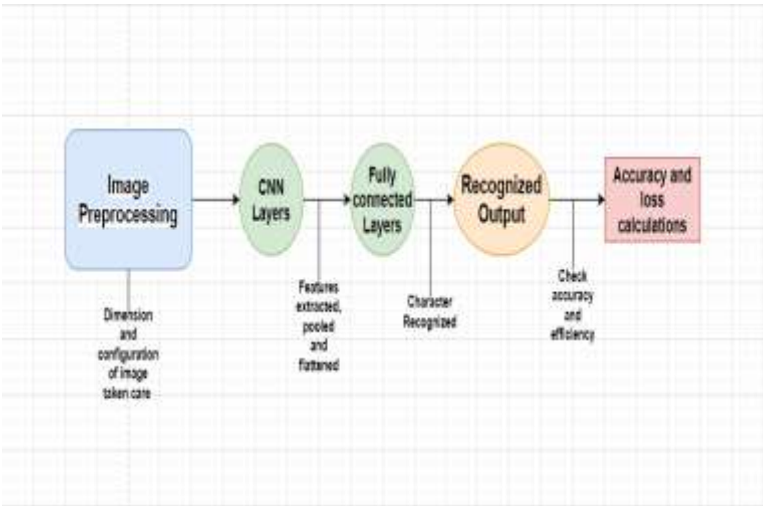


Figure 2: Architecture Diagram

Loss function used in the model is Categorical Cross Entropy. The optimizer involved is the Adam optimizer.

In the final output layer, SoftMax activation function is used to provide the final output.

Copyright ⓒ 2020 SERSC

Results and Discussion

The model summary is as follows-

```
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_12 (Conv2D)           (None, 30, 30, 32)        320
_________________________________________________________________
batch_normalization_8 (Batch (None, 30, 30, 32)        128
_________________________________________________________________
max_pooling2d_8 (MaxPooling2 (None, 15, 15, 32)        0
_________________________________________________________________
conv2d_13 (Conv2D)           (None, 13, 13, 32)        9248
_________________________________________________________________
batch_normalization_9 (Batch (None, 13, 13, 32)        128
_________________________________________________________________
max_pooling2d_9 (MaxPooling2 (None, 7, 7, 32)          0
_________________________________________________________________
conv2d_14 (Conv2D)           (None, 5, 5, 64)          18496
_________________________________________________________________
batch_normalization_10 (Batc (None, 5, 5, 64)          256
_________________________________________________________________
max_pooling2d_10 (MaxPooling (None, 3, 3, 64)          0
_________________________________________________________________
conv2d_15 (Conv2D)           (None, 1, 1, 64)          36928
_________________________________________________________________
batch_normalization_11 (Batc (None, 1, 1, 64)          256
_________________________________________________________________
max_pooling2d_11 (MaxPooling (None, 1, 1, 64)          0
_________________________________________________________________
flatten (Flatten)            (None, 64)                0
_________________________________________________________________
dense (Dense)                (None, 128)               8320
_________________________________________________________________
batch_normalization_12 (Batc (None, 128)               512
_________________________________________________________________
dense_1 (Dense)              (None, 64)                8256
_________________________________________________________________
batch_normalization_13 (Batc (None, 64)                256
_________________________________________________________________
dense_2 (Dense)              (None, 46)                2990
=================================================================
Total params: 86,094
Trainable params: 85,326
Non-trainable params: 768
```

Figure 3: Proposed Model Summary

Testing

Model training is followed by testing. Testing can be done by providing images from different datasets.

There are certain techniques that can be used to achieve better efficiency such as applying data augmentation to increase the size of the dataset, increasing the input size or the number of epochs, or involving much more complex layered structure based on hardware available etc.

In our dataset, 15% of the images are kept as the test set. The proposed model provides an accuracy of 97.64% on the training set and 98.94% on the test set. The accuracy trend while training is depicted by the graph below-

Copyright ⓒ 2020 SERSC

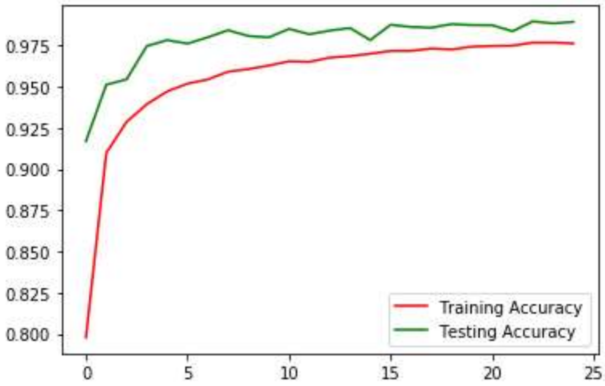


Figure 4: Accuracy Analysis

The loss parameter is also calculated. The following graph depicts the loss analysis-

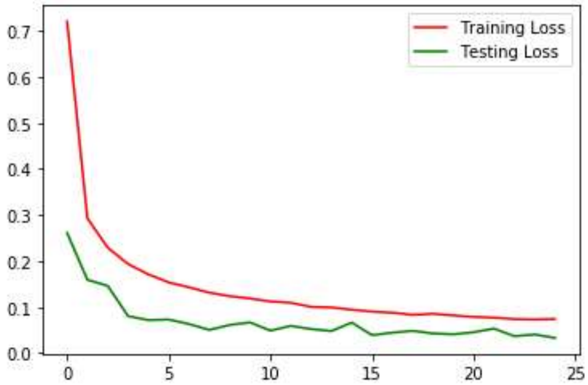


Figure 5: Loss Analysis

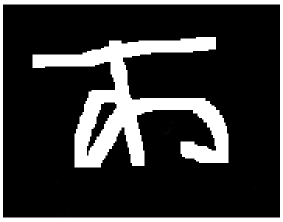
Table 1: Comparison with other CNN models

Model	**Valid Accuracy (in 1st epoch)**	**Best Accuracy (in 15 epochs)**	**Best Accuracy achieved in # Epochs**	**Total Time (15 epochs)**	**Average Training Time per Epoch**
Implemented model	94.78	98.66	14	11m53s	47s
AlexNet	95	98	3	33m 8s	2.2m
DenseNet 121	73	89	7	80m 3s	5.3m
DenseNet 201	74	90	6	113m 22s	7.6m
Vgg 11	97	99	8	86m 6s	5.7m
Vgg 16	97	98	3	132m 12s	8.8m
Vgg 19	96	98	3	148m 57s	9.9m
Inception V3	99	99	1	244m 36s	16.3m

Output

Copyright ⓒ 2020 SERSC

If the following image is given as the input-



Output provided by the model

(a)

(b)

Figure 6 (a) Input character (b) Output character

Conclusion

The handwritten Hindi character recognition model, as presented in the paper, uses Convolutional Neural Networks (CNN) as the recognition network. Four layers of CNN are implemented, followed by three fully connected layers for classification. The model is trained on a dataset containing 78200 training set images and 13200 test set images that belonged to 46 classes of characters. An accuracy of 98.94% is achieved for 25 epochs using the Adam optimizer. Input images of size 32 x 32 are used. Accuracy comparison with similar models shows similar performance as AlexNet and better performance than DenseNet.

In the future, we plan on extending the model to recognize Hindi characters in a block of text, rather than individual characters.

References

1. B. K. Verma, "Handwritten Hindi Character Recognition Using Multilayer Perceptron And Radial Basis Function Neural Networks," Proceedings Of Icnn'95 - *International Conference On Neural Networks, Perth, Wa, Australia,* 1995, Pp. 2111-2115 Vol.4. Doi: 10.1109/Icnn.1995.489003
2. D. Singh, J. P. Saini And D. S. Chauhan, "Hindi Character Recognition Using Rbf Neural Network And Directional Group Feature Extraction Technique," 2015 *International Conference On Cognitive Computing And Information Processing(Ccip), Noida,* 2015, Pp. 1-4. Doi: 10.1109/Ccip.2015.7100726
3. S. Acharya, A. K. Pant And P. K. Gyawali, "Deep Learning Based Large Scale Handwritten Devanagari Character Recognition," 2015 *9th International Conference On Software, Knowledge, Information Management And Applications (Skima), Kathmandu, 2015*,Pp.1-6. Doi: 10.1109/Skima.2015.7400041
4. Poznanski, Arik, And Lior Wolf. "Cnn-N-Gram For Handwriting Word Recognition." In *Proceedings Of The Ieee Conference On Computer Vision And Pattern Recognition*, Pp. 2305-2314. 2016.
5. Lecun, Yann A., Léon Bottou, Genevieve B. Orr, And Klaus-Robert Müller. "Efficient Backprop." In *Neural Networks: Tricks Of The Trade*, Pp. 9-48. Springer, Berlin, Heidelberg, 2012.

Copyright ⓒ 2020 SERSC

6. Hoshank Jitendra Mali, Prof. D.K. Chitre, Sakshi Satish Gadhari, Sarang Sunil Shrikhande, Shruti Shrikant Kharkar, "Handwriting Recognition Using Neural Network", 2017
7. Wu, Jianxin. "Introduction To Convolutional Neural Networks." *National Key Lab For Novel Software Technology. Nanjing University. China* 5 (2017): 23.
8. Monika Jain, Rahul Saxena, Saransh Gupta, Sayam Dhingra, Shivam Solanki, Shubham Sanjay Mor, "Handwritten Text Recognition: With Deep Learning And Android", *International Journal Of Engineering And Advanced Technology (Ijeat)* Issn: 2249 – 8958, Volume-8, Issue-3s, February 2019
9. Priyanka Wankar, Sonali B. Maind, "Research Paper On Basic Of Artificial Neural Network", *International Journal On Recent And Innovation Trends In Computing And Communication, 2014*, Issn: 2321-8169 Volume: 2 Issue: 1
10. Palak Patel, Chirag I Patel, Ripal Patel, "Handwritten Character Recognition Using Neural Network", 2017.
11. Arora, S.; Bhattacharjee, D.; Nasipuri, M.; Basu, D.K.; Kundu, M. Recognition of non-Compound Handwritten Devanagari Characters Using A Combination Of Mlp And Minimum Edit Distance. Arxiv, 2010.
12. T. S. Gunawan, A. F. R. M. Noor, And M. Kartiwi, "Development Of English Handwritten Recognition Using Deep Neural Network," 2018.
13. C. Wu, W. Fan, Y. He, J. Sun, And S. Naoi, "Handwritten Character Recognition By Alternately Trained Relaxation Convolutional Neural Network," 2014 *14th International Conference On Frontiers In Handwriting Recognition*, Pp. 291–296, 2014.
14. Mansi Shah And Gordhan B Jethava "A Literature Review On Handwritten Character Recognition" *Indian Streams Research Journal*, Vol -3, Issue –2, March. 2013, Issn:- 2230-7850.
15. Ankit Sharma, Dipti R Chaudhary "Character Recognition Using Neural Network" International Journal of Engineering Trends and Technology, Volume4, Issue4- April 2013.
16. Prerna Kakkar, Umesh Dutta, "A Novel Approach To Recognition Of English Characters Using Artificial Neural Network" *International Journal Of Advanced Research In Electrical, Electronics And Instrumentation Engineering* (An Iso 3297: 2007 Certified Organization)Vol. 3, Issue 6, June 2014
17. Maharrey, Brandon. *A Neural Network Implementation Of Optical Character Recognition*. Technical Report Number Csse10-05 Comp 6600–Artificial Intelligence Spring 2009 [8], 2009.
18. Anjali Chandavale, Suruchi Dedgaonkar, Dr Ashok Sapkal "An Approach For Character Recognition Using Pattern Matching With Ann *"International Journal Of Scientific & Engineering Research,* Volume 3, Issue 10, October-2012 1 Issn 2229-5518
19. A. Graves And J. Schmidhuber, "Offline Handwriting Recognition With Multidimensional Recurrent Neural Networks,*" In Advances In Neural Information Processing Systems*, 2009, Pp. 545–552.
20. Manoj Sonkusare And Narendra Sahu "A Survey On Handwritten Character Recognition (Hcr) Techniques For English Alphabets*" Advances In Vision Computing: An International Journal (Avc)* Vol.3, No.1, March 2016

Copyright ⓒ 2020 SERSC