Handwritten Devanagari character recognition using convolutional neural networks

Savitha Shetty
Department of Computer Science & Engineering
NMAMIT, Karkala, Karnataka, India
Email- shettysavi1@nitte.edu.in

Saritha Shetty
Department of Master of Computer Applications
NMAMIT, Karkala, Karnataka, India
Email- shettysaritha1@nitte.edu.in

Abstract- Technique of recognizing Handwritten Devanagari characters using our Convolutional Neural Networks is mentioned in our paper. The goal is to create an efficient neural network for Devanagari character recognition which can yield high accuracy rate. The features of the dataset images are extracted by underlying convolution layers by applying 3×3 filters across images to convert them into feature maps. feature maps are then down sampled using the max pooling layer by applying a kernel of dimension 2×2. Then the output is being sent to the fully connected layer where they are computed and a frequency distribution list which will add up to 1 is received. And the index the largest number is compared with the labels index to get the prediction of the model. This approach of training the model with Dropouts prevents the model from over fitting and the trained model can be served as a base, from which other functionalities for Natural Language Processing can be built on top of.

Keywords – Natural Language Processing, Convolutional layer, Devanagari characters

I. INTRODUCTION

Devanagari is the oldest Indian script which is used to write languages like Sanskrit, Marathi, Hindi and several others languages. And handwritten digit recognition of Devanagari characters opens a wide range of possibilities in Natural Language processing field where handwritten characters are recognized and saved electronically which is faster than having to type them all again. In order to make that happen, an efficient neural network has to be constructed which can recognize characters accurately and serve as a base point from which other functionalities can be built on top of. This method of an efficient neural network architecture is mentioned in our paper.

II. RELATED WORK

Pradeep ,S. Himavathi and J.E. Srinivasan, , in their work have done the recognition of characters in offline mode using diagonal feature extraction method. It is based on new ANN model. They have used 2 approaches which uses 54 and 69 features to build our recognition system using neural networs[1]. Yoshimasa Kimura have proposed a method in which features are selected for Character Recognition using Genetic Algorithm. For character recognition novel method is used using genetic algorithms(GA)[2].

Bhatia Neetu had implemented a technique in which computer can recognize the symbols and characters which are written in hand by people in natural handwriting which is called recognition of handwriting[3].

Venu Govindaraju & Liana M. Lorigo proposed that recognizing handwriting by writing using stylus pen and touch pad. System of recognition of characterss is classified as segmentation free (global) and segmentation based (analytic) categories[4].

Francisco Z. M, Salvador España-Boquera, Jorge G. M. and Maria J. C. B., proposed that hybrid HMM model is used for recognition of offline handwritten texts. Here structural part of optical model has been modelled using Markov chains, and probability of emission is estimated using Multilayer Perceptrons [5].

F. Kimura ,U. Pal and T. Wakabayashi have implemented a quadratic_classifier_scheme for recognition of offline handwritten numbers for 6 Indian scripts[6].

S. Chaudhury, Lipika Dey and Reena Bajaj have used three kinds of features like descriptive component moment features, features and density features for classification of Numbera in Devanagari. Multi_classifier was proposed for increasing the reliability and they obtained 89.6% accuracy for numbers in handwritten-Devanagari [7].

Sandhya_Arora have used 4 techniques for extraction of features like intersection, straight line fitting feature and shadow_features. Shadow features are globally computed for character_image while other features are ccalculated by dividing the image of character into segments[8].

Ahmed M. AlKhawaldeh , Khedher Mohammed Z. and Abandah Gheith A. have Recognized characters using features. Arabic characters are used for training and testing the system [9].

Cleber Zanchettin, Neves Renata F. P., Filho Lopes, Mello Carlos A.B., Alberto N. G. described how to recognize handwritten digits using SVM.Performance of SVM is better a s compared to Multilayer perceptron classifier. Experiment was done by considering NIST SD19 dataset. One of the advantage of MLP is it segment classes which are non-linearly separable [10].

Fatos T. Yarman-Vural and Nafiz Arica, developed a method which removes most of the available preprocessing operations, as a result there is loss of important information. They have developed powerful segmentation algorithm which decreases the over-segmentation [11].

Anup Kumar Panda and Sushree Sangita Patnaik have proposed 2 algorithms which used for the optimal harmonic_compensation and it minimizes undesirable losses which occurs inside the APF [12].

III.PROPOSED SYSTEM

The proposed system includes three convolutional layer networks with max pooling layers. This allows our system to work on Aggressive convolutions followed by lightweight densely connected layers. So the feature mapping is done by extracting smallest significant features of the image and only those features can be then moved to the fully connected neural networks in the forthcoming layers. This allows the system to be robust and intelligent and also efficient in recognizing handwritten digits.

IV. METHODOLOGY

The whole process of training the model and recognizing the characters is divided into 3 parts.

A. Cleaning the data
B. Training the data
C. Predicting the data

Cleaning the data

In this phase the data converted from the image matrix values to numpy array of pixel value and image label. The image is represented as numpy array since tensorflow framework which is used to train the model, expects tensors or numpy arrays as training instance. The image that is used to train the data, each contains 32x32 pixels having 3 color channels (R,G,B). Since extra detail of the color channels is not significantly necessary for training, three color channels are reduced to one color channel by gray scaling. After applying grayscale filter to the image, it represents values of image pixel in ranges between 0 to 255, here 255 is white pixel and 0 is black pixel. Here values between 0 to 255 is normalized to range between 0 and 1 since that is the optimal range for image processing in machine learning. The same operation is recursively applied on all images present in the dataset and

at the end a 2D numpy array of normalized pixel value and their corresponding labels is obtained, ready for training. Figure 1 shows the conversion from image matrix to numpy matrix.

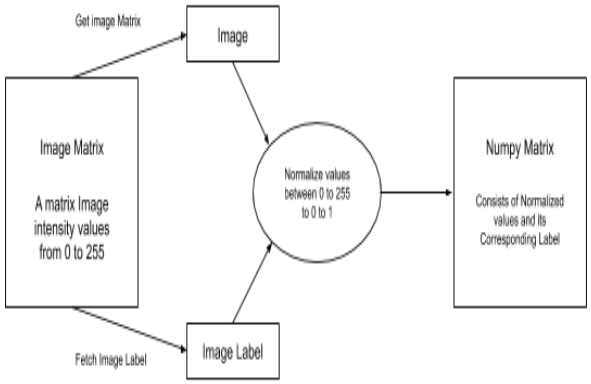

Figure 1:Conversion from image matrix to numpy matrix

Training the Data: Training the data is the crucial part of the entire process where the transformed images matrix which numpy arrays are used to train the model. For training the Sequential neural networks are used where the output of previous layer is input to next layer. For our specific problem, the following layers are stacked.

a. 2D convolutional layer with 3x3 kernel (convolution)
b. 2D max-pooling layer with 2x2 kernel (downsampling)
c. 2D convolutional layer with 3x3 kernel (convolution)
d. 2D max-pooling layer with 2x2 kernel (downsampling)
e. 2D convolutional layer with 3x3 kernel (convolution)
f. Fully connected layer with 64 units (feed forward)
g. Fully connected layer with 46 units (feed forward)

- Convolutional Layer. Convolutional Neural Network is applied while analyzing the visual_imagery. multilayer perceptrons which are regularized are CNNs. Because of "fully-connectedness" nature there is problem of over fitting data. So solution is to regularize it by adding weights to the loss function.
- Pooling Layer. We are adding a pooling layer in between the convolution layers. It reduces the amount of computation and number of parameters and thus controls overfitting. It uses MAX operation on every slice of input to resize it. Generally used pooling layer is filter with 2X2 having a stride of 2 which downsamples input by 2 along height and weight which discards 75% of activations.
- Fully Connected Layer. Here the Neurons are connected to all activations in our previous layer. The activations are calculated by doing bias offset and matrix multiplication. Table 1 shows the model architecture and functionality.

Table - 1 Model architecture and functionality

Layer (type_)	**Shape of Output**	**Par ams**
Conv_2d (Conv_2D)	(None., 30, 30, 32)	320
maxpooling2d (MaxPooling_2D)	(None_., 15, 15, 32)	0
_conv2d_1 (Conv2D_)	(None_., 13, 13, 64)	18496
maxpooling2d_1 (Max_Pooling2)	(None_., 6, 6, 64)	0
conv2d2 (Conv_2D)	(None_., 4, 4, 64)	36928
Flatten_ (Flatten_)	(None_., 1024)	0
Dense_ (Dense_)	(None_., 64)	65600
Drop out_ (Drop out_)	(None_., 64)	0
dense_ 1 (Dense_)	(None_., 46)	2990

2D Conv and 2D max-pooling (First pass). This is called as input layer. In the first convolutiona layer, 32 units were used with 3x3 kernels which produces a 3x3 convoluted image of 32 sets each. This produces the output feature maps of size 30x30x32 which is 32 sets of 30x30 convoluted images. Which produces a total of 320 parameters for the next layer. Layer here contains Relu activation which is used to convert negative value to 0. After max pooling with 2x2 kernels, 32 sets of 15x15 images are generated. Then the 15x15x32 feature maps are sent to the next layer.

2D Conv and 2D max-pooling (Second pass). In the second convolution layer, further convolution is applied to the feature maps that are generated in the previous layers. After applying the convolution of 3x3 for all the feature maps, an output shape of 13x13x64 is generated where 64 sets of images with 13x13 dimension is obtained. Layer here contains Relu activation which is used to convert negative value to 0. After max pooling with 2x2 kernels, 64 sets of 13x13 images are reduced to 64 sets of 6x6 images. Then these feature maps are sent to the next layer in the network.

2D Conv (Third pass). In the third convolution layer, further convolution is applied to 64 sets of images with dimension 6x6 to obtain 64 sets of feature maps with dimensions 4x4. These feature maps are being sent to the next layer in the network. Layer here contains Relu activation which converts negative value to 0.

Flatten Layer (Fourth pass). In this layer, the 2D feature maps are converted to 1D vector by flattening the 2D matrix. This is necessary since the dense layer expects a flat array as an input so this layer acts as the transformer for transforming data feeding into the dense layer. This layer takes the input 4x4x64 and converts it into 1024x1.

Fully Connected/Dense Layer (Fifth pass). The fully connected layer is a simple neural network where simple feed forward mechanism happens. Layer here contains 64 units with Relu activation which is used to convert negative value to 0 which is handy for image processing.

Dropout Layer. Dropout layer drops certain neurons at the training stage. During training stages, nodes are dropped from the net with probability of 1-p , kept in the network having probability p, here network which is reduced is left; outgoing , incoming edges to dropped-out nodes are removed.

Dense Layer/ Fully Connected (Sixth pass). This dense layer is the output layer containing 46 units which is same as the number of classes/labels. This layer contains a softmax activation which distributes the probability across 46 units so that their probabilities can add up to 1. And the index of the largest of those probability distribution is the prediction given by the model.

Predicting the data : When the training process ends, model is generated after training which can be used to predict any handwritten character which is sent to the model containing 32x32 dimensions. The model can be loaded by using tools like Tensorflow. Any image in a large space has to be applied character segmentation and resized to 28x28. And the resized image has to be applied padding of 4 pixels on four sides and should be sent for the prediction into the model. While predicting, the model gives the probability distribution of which the largest index of the largest value in the list is the prediction of the model.

V. RESULTS AND DISCUSSIONS

After training the model with 92000 public dataset instances containing, 78200 training instances and 13800 testing instances with 46 classes for 10 epochs each, the Accuracy of the model is 0.9746377015113831 and the loss is 0.14186847770127697. Figure 2 represents the minimizing loss and figure 3 represents the raising accuracy of the model.

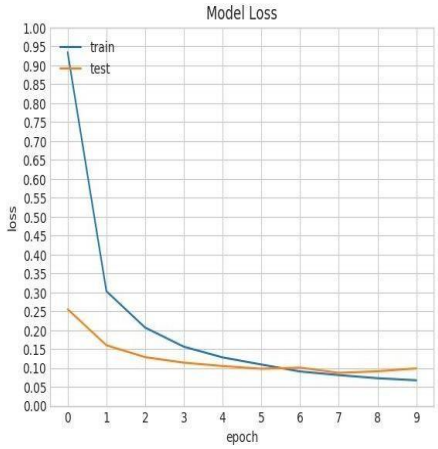

Figure 2: Raising accuracy of the model

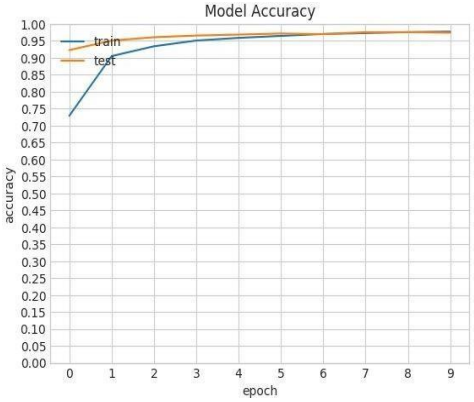

Figure 3: Raising accuracy of the model

In the above graphs, the 10 epoch values ranges from 0 to 9 in the x axis and accuracy of the model ranging from 0 to 1 in the y axis is plotted. The blue line represents the accuracy and loss of the model plotted in the course of the training per epoch. And the brown line is the accuracy and loss of the model plotted after testing the model after each In the "Model Accuracy" graph because of epoch. In the "Model Accuracy" graph because of dropouts, the model keeps losing inactive neurons in each epoch.

In the "Model Accuracy" graph because of dropouts, the model keeps losing inactive neurons in each epoch, and then the model has to put extra efforts in learning new patterns, which is a good sign since it doesn't allow the model to over-fit and remain flexible even after the training is done. So the model remains flexible for 92,000 records. The more instances the model is trained with, more efficient it would be. So the model will gain accuracy as the number of training/testing instances are raised.

VI. CONCLUSION

Efficient neural network for Devanagari character recognition which can yield high accuracy rate is created. The features of the dataset images are extracted by underlying convolution layers by applying 3×3 filters across images to convert them into feature maps. feature maps are then down sampled using the max pooling layer by applying a kernel of dimension 2×2. This approach of training the model with Dropouts prevents the model from over fitting and the trained model can be served as a base, from which other functionalities for Natural Language Processing can be built on top of.

REFERENCES

[1] G J. Pradeep, , S. Himavathi, Srinivasan E. "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.

[2] Kimura Yoshimasa, "Feature Selection for Character Recognition Using Genetic Algorithm", "Fourth International Conference on Innovative Computing, Information and Control"

978-0-7695-3873-0/09© 2009 IEEE, 2009.

[3] Bhatia Neetu, "Optical Character Recognition Techniques", "International Journal of Advanced Research in Computer Science and Software Engineering", Volume 4, Issue 5, May 2014.

[4] Liana M. Lorigo and Venu Govindaraju, "Offline Arabic Handwriting Recognition: A Survey", "IEEE Transactions on Pattern Analysis and Machine Intelligence", Volume 28 Issue 5, May 2006.

[5] Salvador España-Boquera, Maria J. C. B., Jorge G. M. and Francisco Z. M., "Improving Offline Handwritten Text Recognition with Hybrid HMM/ANN Models", "IEEE Transactions on Pattern Analysis and Machine Intelligence", Vol. 33, No. 4, April2011.

[6] U. Pal, T. Wakabayashi and F. Kimura, "Handwritten numeral recognition of six popular scripts," "Ninth International conference on Document Analysis and Recognition ICDAR 07", Vol.2, pp.749-753, 2007.

[7] Reena Bajaj, Lipika Dey, and S. Chaudhury, "Devanagari numeral recognition by combining decision of multiple connectionist classifiers", Sadhana, Vol.27, part. 1, pp.-59-72, 2002.

[8] Sandhya Arora, "Combining Multiple Feature Extraction Techniques for Handwritten Devnagari Character Recognition", "IEEE Region 10 Colloquium and the Third ICIIS", Kharagpur, INDIA, December 2008.

[9] Mohammed Z. Khedher, Gheith A. Abandah, and Ahmed M. AlKhawaldeh, "Optimizing Feature Selection for Recognizing Handwritten Arabic Characters", "proceedings of World Academy of Science Engineering and Technology", vol. 4, February 2005 ISSN 1307-6884.

[10] Renata F. P. Neves, Alberto N. G. Lopes Filho, Carlos A.B.Mello, CleberZanchettin, "A SVM Based Off-Line Handwritten Digit Recognizer", "International conference on Systems, Man and Cybernetics, IEEE Xplore", pp. 510-515, 9-12 Oct, 2011, Brazil.

[11] Nafiz Arica, and Fatos T. Yarman-Vural, "Optical Character Recognition for Cursive Handwriting,"[J]," IEEE Transactions on Pattern Analysis and Machine Intelligence", vol.24, no.6, pp. 801-113, June 2002.

[12] Sushree Sangita Patnaik and Anup Kumar Panda, "Particle Swarm Optimization and Bacterial Foraging Optimization Techniques for Optimal Current Harmonic Mitigation by Employing Active Power Filter Applied Computational Intelligence and Soft Computing", Volume 2012, Article ID 897127.