Machine Learning and Deep Learning Approaches for Sanskrit Character Recognition

Jadeja Pratikshaba S.
PG Scholar,SVIT
Vasad,Gujarat

Dr. Nehal G. Chitaliya
Head of E&C Department,SVIT
Vasad,Gujarat

Abstract :- ***The main objective of any character recognition system is to achieve modification or conversion of any form of text such as printed or scanned text images. Most of the greatest literature works in ancient India were written in Sanskrit. Hence it is necessary to preserve these highly prestigious documents by digitizing them. For digitization of these documents it is necessary that these characters get accurately recognized by the machine for which it need an accurate classifier. In this paper shows comparison between various classifiers used for letter classification which are designed using machine learning and deep Learning algorithms***

Keywords: ***Thresholding, Classifier, SVM, Deep learning, CNN, Discrete wavelet transform***

I. INTRODUCTION

Sanskrit is considered as the mother as of all the modern languages. Ancient Indian manuscripts were written in Sanskrit language but are in poor condition though many of the documents are reprinted but still are poorly maintained so before these documents get completely damaged their digitization becomes more crucial. But most of the existence OCR have only worked on classification and identification of single Sanskrit characters hence it becomes necessary that a better OCR is developed which can classify not only single characters but also words and sentences. The classifier used can be designed using both machine learning and deep learning approaches. Similarly the OCR systems developed in other languages are mostly designed using machine learning algorithms like A.Chaudhary [3] have developed an OCR system for hindi language where the fuzzy Hough transform is used for feature extraction. An efficient Devanagari Character Classification using Support Vector Machine (CC-SVM) [8] method is proposed by Shalini Puri and Satya Prakash Singh The Block Diagram shows various steps of the character recognition using classifier.

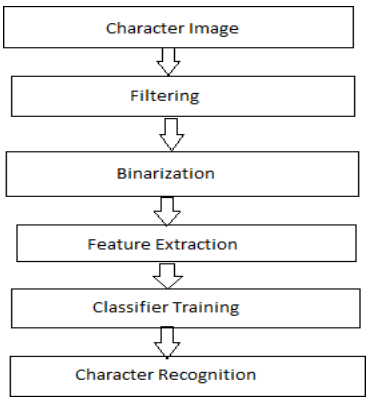

Figure 1: Block Diagram

The main difference between machine learning and deep learning algorithms is that in machine learning algorithms features are to be extracted by the designer and then fed to the classifier while in deep learning, based on input and what feature need to be extract only the kernel value need to be provided. Based on the required output various deep layers are designed while feature is extracted by neural networks. For machine learning based classifier design discrete wavelet Transform, moment feature extraction and combination of both the methods for feature extraction. Two different Classifier one SVM and one CNN are compared. Two different SVM models are used as classifier one of it is trained with feature vector as input which are generated using wavelet features and other model trained using feature vector formed by combination of Wavelet and moment features and both the classifiers are compared with CNN classifier. The OCR system designed till now have only classified single Sanskrit characters and not worked on line and word segmentation and segmentation and classification of faded and distorted characters. The test data is shown below. Many of the other OCR designed have used that test data which were without faded lines and distorted characters.

Figure 2: Cropped image of Sanskrit characters used as test data [1]

II. METHODOLOGY

The steps involved for character recognition starts with filtering of image which removes noise, binarization of image before that if image is coloured then convertion of RGB to grey scale image, feature extraction, and training of classifier and then testing process which is defined successful if it recognizes the sanskrit character even if has different font style than that used for training the classifier.

A) Datasets:

The Dataset used for training the classifier consists of total 12,912 images of Sanskrit characters out of which 2880 images are of numbers, 2652 images of vowels, 7380 images of consonants of Sanskrit characters which are downloaded from the following link : https://www.kaggle.com/rishianand/devanagari-character-set and would be used to train the classifier. Few of the sample images are shown below.

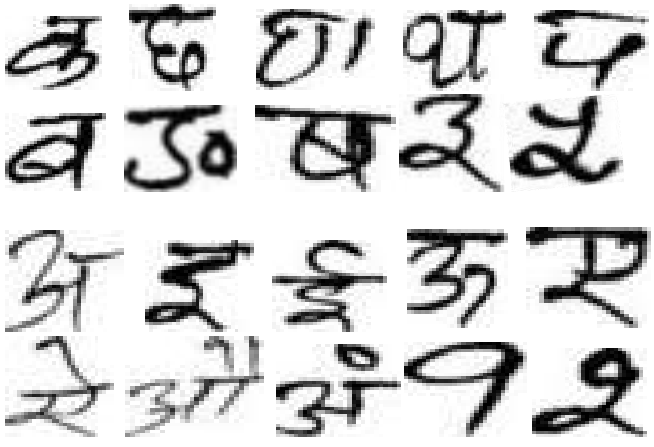

Figure 3: few Sample of character images used for training classifier

B) Filtering:

Before designing of classifier it is necessary that images are filtered for removal of unwanted noise. The Dataset is divided into 58 different classes each class consisting of images of same letter but of different font style. Before filtering process each image is resized form 28x28x3 pixels to 32x32x1 pixels.

To find out which filter to be applied before segmentation a part of cropped document is used for testing of various filters

The images shown below helps us to compare various filters

i) Noisy input image:

A noisy image from Sanskrit document is used for testing the filters. The matrix size of image is 109x588 which consists of salt&pepper type of noise

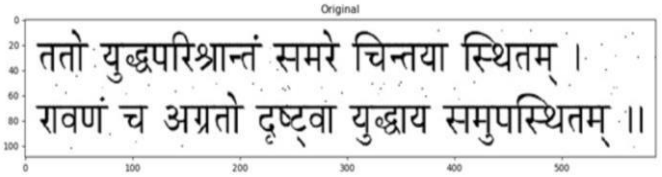

Figure 4: Cropped part of Sanskrit document.

ii) Median filter:

A filtering window of 3x3 is applied on the image where the central pixel value is replaced with the median value of the corresponding pixels which helps to remove noisy pixel of irrelevant value.

252	255	252
252	2	255
255	255	253

Figure 5: a part of matrix of 109x588 matrix

Neighbourhood value:

2,252,252,252,**253**,255,255,255,255

Median value: 253

Here the value of 2 is replaced with the value 253 which removes noise and fills the faded portion in the image. After applying the median filter to the original cropped image the output obtained is shown below.

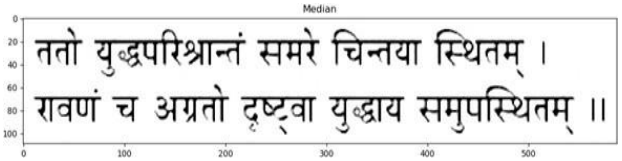

Figure 6: output after applying median filter

iii) Gaussian filter:

Gaussian filter is a smoothing filter which reduces noise by blurring. The 2D Gaussian smoothing filter is given by the

equation

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Where σ is the variance of the mask whose value chosen is 1

Figure 7: Output after applying Gaussian filter

iv) Bilateral filter:

It is a smoothing filter which reduces noise by preserving edges where each pixel intensity is replaced with weighted average of intensity values from nearby pixels

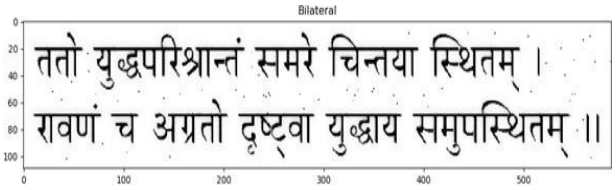

Figure 8: Output after applying Bilateral filter

v) Average filter:

Average filtering is a smoothing images by reducing amount of variation of nearby pixels. It works by replacing each value with the average value of neighbouring pixels, including itself. Smoothing through average filter is done by performing convolution of image with the matrix shown below.

1	1	1
1	1	1
1	1	1

Filter kernel is sided across the original image and the output obtained is shown below.

Figure 9: output after applying Average filter

From the above images it is clear that median filter gives better output compared to other filters hence before providing input to the classifier images first passed through median filter for noise removal.

C) Feature Extraction:

An image is a matrix consisting of certain pixel values. Feature extraction process helps to reduce this matrix size and forms a feature vector which contains information of image like its texture, area, shape, colour, object, edges, etc. The feature extraction used here are discrete wavelet transform (DWT) and moment features extraction

Discrete Wavelet Transform:

Using Discrete Wavelet Transform information can be extracted in time domain and frequency domain as well. Hence we can say that Wavelet has a multi-goals properties by deteriorating the sign in various scales.

Mathematically DWT is calculated using the following equation:

$$f(x) = \sum_k c_{j_0}(k)\phi_{j_0,k}(x) + \sum_{j=j_0}^{\infty}\sum_k d_j(k)\psi_{j,k}(x)$$

where j_0 is an arbitrary starting scale

$$c_{j_0}(k) = \left\langle f(x), \tilde{\phi}_{j_0,k}(x)\right\rangle = \int f(x)\tilde{\phi}_{j_0,k}(x)dx$$

called the approximation or scaling coefficients

$$d_j(k) = \left\langle f(x), \tilde{\psi}_{j,k}(x)\right\rangle = \int f(x)\tilde{\psi}_{j,k}(x)dx$$

called the detail or wavelet coefficients

In image processing the block diagram shown below shows that how DWT works for extracting features from image.

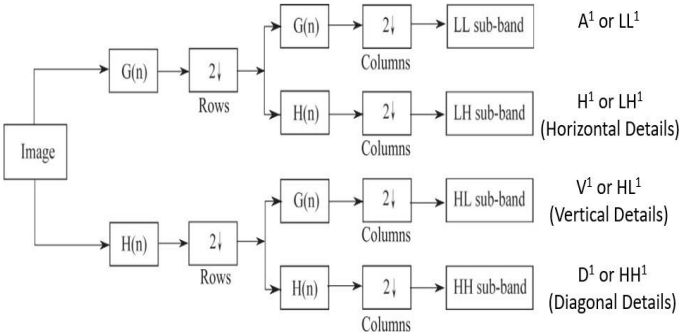

Figure 10: Block diagram of DWT

Using this Wavelet feature extraction technique vertical and diagonal features of the image are derived in the form of feature vector which is a row vector which is described as CA and CD respectively as vertical and diagonal feature vectors.

Moment Feature:

To find features like area, perimeter, centroid contour features are used. To find this various features image moment is supposed to be calculated this moment is used to calculate various datas like area, perimeter centroid etc where centroid is given by relation

Cx = M10/M00 and Cy = M01/M00.

Contour Area: It is defined as total number of pixels in the image.

Contour Perimeter: It gives total number of pixels in the boundary. It is also called arc length. argument specify whether shape is a closed contour (if passed True), or just a curve.

D) Classifiers:

SVM

Support vector machine (SVM) is one of the machine learning approach which is used for classification problem. The algorithm used here is simple it creates a line or hyperplane which separates the data into classes.

Support vector machine (SVM) used in this experiment is a linear SVM. Since SVM is a machine learning approach method to design a classifier feature vectors are to be extracted by the designer and then given as input to the SVM model for training. First SVM is trained with wavelet feature matrix and then combination of wavelet and moment feature is done The figure shown below shows that how SVM divides data into two separate classes.

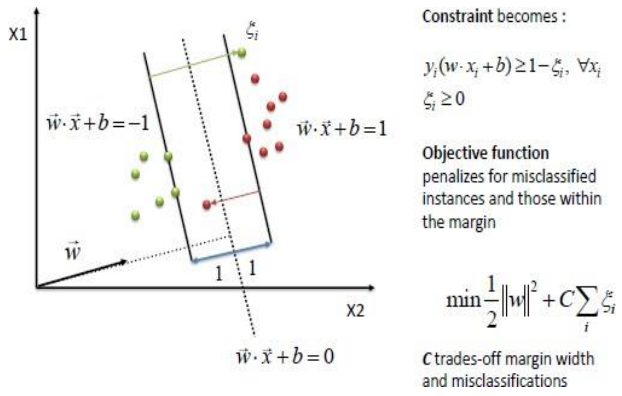

Figure 11: SVM model for two classes.

As per our model SVM divides the data into 58 different classes. The kernel selected for our model is linear and the value of C = 1.

Convolution Neural Network

Convolution neural network is a deep learning approach to design a classifier. A CNN is combination of different layers.

The figure shown below shows how various CNN layers work.

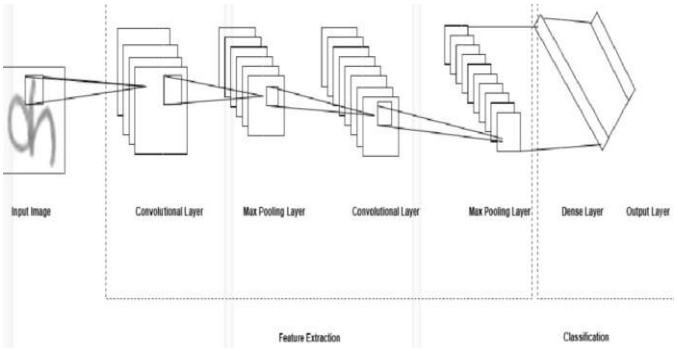

Figure 12: CNN Layers

Convolutional layer — It generates a feature Vector of each image each by applying a filter that scans the whole image, few pixels at a time.

Pooling layer (down sampling) — Scales down the measure of data the convolutional layer created for each component and keeps up the most fundamental

Data (the procedure of the convolutional and pooling layers for the most part rehashes a few times).

Fully connected layer(input layer) — After down sampling is performed by the pooling layer the final output obtained is a row and column matrix which is converted into row matrix by using ―Flatten‖ function.

Fully connected network — It is the central layer which connects input layer with output layer and hence it forms a dense layer.

Fully connected layer(output Layer) — creates the last probabilities to decide a class for the picture.

The main advantage of deep learning method is that we just have to provide image as input to the network and the network itself extracts features and learns by itself.

III. PROPOSED CLASSIFIER MODEL

The proposed classifier model is designed in Anaconda Navigator in spider 3.3.6 version of software environment. The variable explorer of this software helps us to view value matrics of various variable parameter and Python console window helps us to view the all layers of the CNN which we have designed.

In the proposed classifier model a CNN classifier which possess two deep layers have been designed. Here of 12911 image 80% images are used for training the neural network and rest 20% for testing of the data. Before providing input to the CNN images are first resized from 28x28x3 to 32x32x1 and converted from RGB to grey and certain morphological operations are performed which includes

filtering for which median filter is used and segmentation of letters from background which is done by performing thresholding operation on the image. The dataset we have used is divided into 58 different classes each class of single letter and numeric value consisting of more than 200 images which will be used as input in the neural network.

Table-I: CNN Layers

```
Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_5 (Conv2D)            (None, 30, 30, 28)        280
_________________________________________________________________
conv2d_6 (Conv2D)            (None, 28, 28, 64)        16192
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 14, 14, 64)        0
_________________________________________________________________
conv2d_7 (Conv2D)            (None, 12, 12, 64)        36928
_________________________________________________________________
conv2d_8 (Conv2D)            (None, 10, 10, 64)        36928
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 5, 5, 64)          0
_________________________________________________________________
dropout_2 (Dropout)          (None, 5, 5, 64)          0
_________________________________________________________________
flatten_4 (Flatten)          (None, 1600)              0
_________________________________________________________________
dense_4 (Dense)              (None, 128)               204928
_________________________________________________________________
dense_5 (Dense)              (None, 64)                8256
_________________________________________________________________
dense_6 (Dense)              (None, 58)                3770
=================================================================
Total params: 307,282
Trainable params: 307,282
Non-trainable params: 0
_________________________________________________________________
None
Train on 10329 samples, validate on 2583 samples
```

In the designed layer Activation function used is relu and kernel of 3x3 is used on 32x32 image and convolution operation is performed between the image matrix and the kernel matrix. 'Relu' function is used to provide non-linearity .After that maxpooling operation performed by 2x2 matrix reduces the image matrix by half i.e. matrices of 28x28 is reduced to 14x14 and considers only essential information. The same operation is performed by the second layer of the CNN. Dropout layer used after that would deactivate around 20% of neurons that give same output to avoid the problem of overfitting. After that the feature matrices obtained is flatten which converts feature map to a single column and passed to fully connected layer which is formed by two dense layers. Here dense layers connects each output layer with input layer thus forming a fully connected network each layer with 128 and 64 neurons.

IV. RESULT

MODEL ACCURACY and MODEL LOSS GRAPHS

The graph below shows model accuracy and model loss graph of train test data of CNN classifier with 20 epoch.

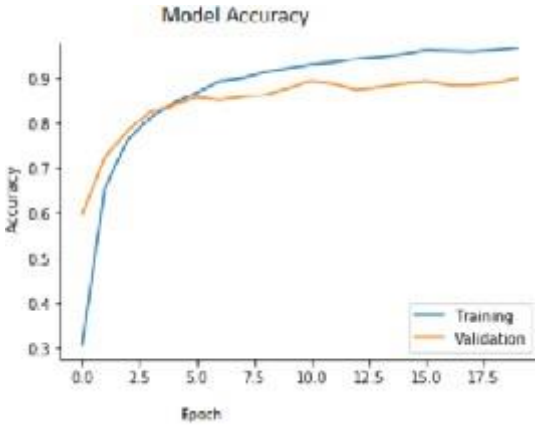

Figure 13: model accuracy

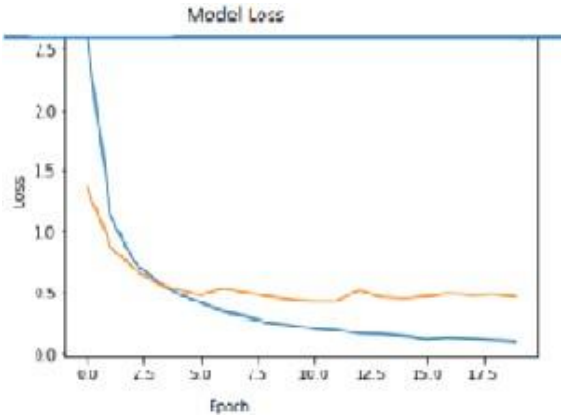

Figure 14: model loss graph

Table: Time required to train and test various classifiers

Classifier	Time Training	Accuracy
SVM		
Wavelet features	6 min 23 sec	72.164%
Wavelet+Moment features	6 min 46 sec	70.368%
CNN		
Epoch:20	23min 04sec	89.74%
Epoch:100	2 hrs 11min	89.956%

The table shown above shows that how much time is required to train both the classifier, and how much accuracy each classifier provides, one is designed using machine

learning algorithm i.e. SVM classifier where first classifier is trained using only wavelet feature which is combination of diagonal and vertical feature matrix and combination of wavelet and moment . While for deep learning Convolution Neural Network based classifier is designed which takes more time to train feature. The classifiers are trained without GPU on CPU with 64 bit i3 processor and shows that CNN provides more accuracy compared to SVM classifier.

V. CONCLUSION

As per the results obtained of various classifiers it is clear that Convolution neural network based classifier provides better accuracy compared to SVM classifier hence it can be said that Deep Learning provides better results compared to Machine Learning though training time for CNN is more compared to SVM but CNN will give approx 89.95% accuracy. Hence if it becomes necessary to get faster output use of GPU becomes necessary to reduce the training time otherwise can be avoided to reduce the cost of the project.

REFERENCES

[1] Avadesh, M., & Goyal, N. (2018). Optical character recognition for sanskrit using convolution neural networks. Proceedings - 13th IAPR International Workshop on Document Analysis Systems, DAS 2018, 447–452. https://doi.org/10.1109/DAS.2018.50

[2] Ding, H., Chen, K., & Huo, Q. (2019). Compressing CNN-DBLSTM models for OCR with teacher-student learning and Tucker decomposition. Pattern Recognition, 96, 106957. https://doi.org/10.1016/j.patcog.2019.07.002

[3] Phangtriastu, M. R., Harefa, J., & Tanoto, D. F. (2017). Comparison between Neural Network and Support Vector Machine in Optical Character Recognition. Procedia Computer Science, 116, 351–357. https://doi.org/10.1016/j.procs.2017.10.061

[4] Abdul Robby, G., Tandra, A., Susanto, I., Harefa, J., & Chowanda, A. (2019). Implementation of optical character recognition using tesseract with the javanese script target in android application. Procedia Computer Science, 157, 499–505. https://doi.org/10.1016/j.procs.2019.09.006

[5] Farhat, A., Hommos, O., Al-Zawqari, A., Al-Qahtani, A., Bensaali, F., Amira, A., & Zhai, X. (2018). Optical character recognition on heterogeneous SoC for HD automatic number plate recognition system. Eurasip Journal on Image and Video Processing, 2018(1). https://doi.org/10.1186/s13640-018-0298-2

[6] Jayadevan, R., Kolhe, S. R., Patil, P. M., & Pal, U. (2011). Offline recognition of Devanagari script: A survey. IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews, 41(6), 782–796. https://doi.org/10.1109/TSMCC.2010.2095841

[7] Wei, T. C., Sheikh, U., & Rahman, A. A. H. A. (2018). Improved optical character recognition with deep neural network. Proceedings - 2018 IEEE 14th International Colloquium on Signal Processing and Its Application, CSPA 2018, (March), 245–249. https://doi.org/10.1109/CSPA.2018.8368720

[8] Puri, S., & Singh, S. P. (2019). An efficient Devanagari character classification in printed and handwritten documents using SVM. Procedia Computer Science, 152, 111–121. https://doi.org/10.1016/j.procs.2019.05.033

[9] Thakral, B., & Kumar, M. (2015). Devanagari handwritten text segmentation for overlapping and conjunct characters- A proficient technique. Proceedings - 2014 3rd International Conference on Reliability, Infocom Technologies and Optimization: Trends and Future Directions, ICRITO 2014. https://doi.org/10.1109/ICRITO.2014.7014746

[10] Joshi, D. S., & Risodkar, Y. R. (2018). Deep Learning Based Gujarati Handwritten Character Recognition. 2018 International Conference On Advances in Communication and Computing Technology, ICACCT 2018, 563–566. https://doi.org/10.1109/ICACCT.2018.8529410

[11] Bathla, A. K., Gupta, S. K., & Jindal, M. K. (2016). Challenges in Segmentation of Handwritten text for recognition of Devanagari Scripts: A Review. In the Proceedings of 3rd International Conference on Computing for Sustainable Global Development (INDIACom), 2711–2715.

[12] Adiga, D., Saluja, R., Agrawal, V., Ramakrishnan, G., Chaudhuri, P., Ramasubramanian, K., & Kulkarni, M. (n.d.). Improving the learnability of classifiers for Sanskrit OCR corrections.

[13] Chaudhuri, A., Mandaviya, K., Badelia, P., & K Ghosh, S. (2017). Optical Character Recognition Systems for Different Languages with Soft Computing. Optical Character Recognition Systems, 352(January 2018), 9–42. https://doi.org/10.1007/978-3-319-50252-6

[14] Yin, Y., Zhang, W., Hong, S., Yang, J., Xiong, J., & Gui, G. (2019). Deep Learning-Aided OCR Techniques for Chinese Uppercase Characters in the Application of Internet of Things. IEEE Access, 7(c), 47043–47049. https://doi.org/10.1109/ACCESS.2019.2909401

[15] Nurhayati, Risda, B. C., & Masruroh, S. U. (2014). Optical character recognition feature implementation in cooking recipe application using tesseract Google project. 2014 International Conference on Cyber and IT Service Management, CITSM 2014, 44–47. https://doi.org/10.1109/CITSM.2014.7042173