Implementing HOG features to Recognize Multilingual Characters in Machine Learning

Kethineni Venkateswarlu[1], Nellore Sudha[2], Dr. P. Pavankumar[3]

[1]Assistant professor, Dept of CSE, KMIT, Narayanguda, Telangana, venkat0534@gmail.com

[2]Assistant professor, Dept of IT, KMIT, Narayanguda, Telangana, sdkr.nlr@gmail.com

[3]Assistant professor, Dept of CSE, KMIT, Narayanguda, Telangana, pavankumar.palla@gmail.com

***Abstract*- Text recognition has been increasing its attention in computer vision community in recent years, particularly to assist blind personnel. This paper focuses on the problem of text recognition from scene images. Proposed a novel methodology to recognize the characters from scene images. Extracting Histogram of Oriented Gradients (HOG) features are done firstly. Subsequently the extracted HOG features are supplied to train Support Vector Machine (SVMs). Trained SVM model recognizes the new text characters. For comparison purpose Neural Networks (NN) are used to recognize the characters present in scene images. The SVM and NN Experimental results prove that SVM performs better than NNs. Experiments is done with English, Telugu and Hindi languages to recognize respective characters and found that the algorithm works very well.**

Key words: HOG, SVM, NN, TEXT.

1. INTRODUCTION

With the rapid growth of Technology there are so many applications are available for capturing the Images in mobile devices and many other portable devices. The main Problem with those images are whenever the text is in read mode and needed to present in that image it gains so much of attention, and it is the main problem in the computer vision community in recent years. In some conditions like navigation of automobiles through cameras facing in streets plays a vital role for reading scene text for assisting a blind man to navigate in certain indoor environments, and its plays an important role in language translation [1]

In this paper, it is focused on special case of the scene text problem where given a list of small letters, capital letters, in English and also focused on two different Indian languages Hindi and Telugu to be detected and read. Work is focused mainly in the four fields

(i) Optical Character Recognition (OCR) is used to recognize the text presented in the images.
(ii) Histograms of Oriented Gradients (HOG) its extracts the features of every image presented in the Datasets.
(iii) Support Vector Machines (SVM) the HOG features give to the SVM it is able to classify an object into one of two classes. Finally used the
(iv) Neural Networks (NN) for comparison purpose.

1.1 Optical Character Recognition (OCR)

The OCR is the one of the best method for identifying the text presented in the images. It is mainly classified into two categories: traditional Optical Character Recognition (OCR) based and object recognition based. For traditional OCR based method various binarization methods have been proposed to get the binary image which is directly fed into the off-the-shelf OCR engine. On the other hand, object recognition based methods assume that scene character recognition is quite similar to object recognition with a high degree of intraclass variation. For scene character recognition, these methods directly extract features from original images and use various classifiers to recognize the character [2].

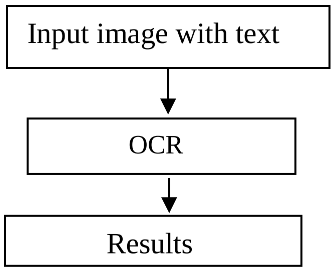


Figure1.1: Optical Character Recognition

2. Histogram of Oriented Gradients

The Histogram of Oriented Gradients (HOG) it extracts the features of every image. HOG gives the good performance on many computer vision tasks. The HOG descriptor is that local object appearance and shape within a picture are often described by the distribution of intensity gradients or edge directions. The descriptors are implemented by dividing the image into cell which are small connected regions and for every cell compiling a histogram of gradient directions or edge orientations for the pixels within the cell. The combination of these histograms then represents the descriptor [3].

2.1 GRADIENT COMPUTATION

It calculates the many features detectors in image preprocessing are to ensure normalized color and gamma values. Image preprocessing thus provides little impact on performance. The most common method is to apply the 1-D centered, point discrete derivative mask in both of the horizontal and vertical directions [4]. It Specifically, this method requires filtering the colour intensity data of the image with the subsequent filter kernels: [-1, 0, 1] and $[-1, 0, 1]^T$.

2.2 ORIENTATION BINING

It involves creating the cell histograms. Each pixel within the cell casts a weighted vote for an orientation based histogram channel supported the values found within the gradient computation. The shape of the cells can be rectangular or radial in shape such that the histogram channels are evenly cover 0 to 180 degrees or 0 to 360 degrees, counting on whether the gradient is "unsigned" or "signed". The unsigned gradients used in conjunction with 9 histogram channels performed best the detection [5].

2.3 BLOCK NORMALIZATION

The block normalization has the four different methods. Let *v* be the non-normalized vector containing all histograms in a given block, $||v||k$ be its k-norm for k=1, 2 and e be some small constant [6]. The major factor of normalization can be one of the following:

L2-norm: $f = \frac{v}{\sqrt{||V||22+e2}}$ ------------(1)

L2-hys: renormalizing and clipping follows L2-norm

L1-norm: $f = \frac{v}{(||v||1+e)}$ ---========-- (2)

L1-sqrt: $f = \sqrt{v}(||v||1 + e)$–======== ---(3)

L2 Hys can be computed by first taking the L2-norm, clipping the result and then renormalizing.

3. SUPPORT VECTOR MACHINES

Here mainly focused on the identifying individual characters for that SVM is used due to its good performance. The SVM produces a binary classifier, which means that it is able to classify an object into one of two classes. In order to use a binary SVM for digits totally have 10 digits, so 10 such a classifiers are required, each one trained for specific digit.

This is a common technique used to solve multi-class classification problems with binary classifiers. Similarly required 26 classifiers for the both the small letters and capital letters. SVMs are set of related supervised learning methods used for classification and regression. They belong to a family of generalized linear classification. A special property of SVM is, SVM simultaneously minimize the empirical classification error and maximum margin classifiers. SVM based on Structural risk Minimization (SRM). SVM map input vector to higher dimensional space where a maximal separating hyperplane is constructed. Two parallel hyperplanes are constructed on each side of the hyperplane that separate the data [7].

Considered the data set points of the form

$$\{(X1, Y1), (X2, Y2), (X3, Y3) \dots \dots . (Xn, Yn)\}$$

Where $Y_n = 1/-1$, a constant denoting the class to which that point X_n belongs. n=number of sample. Each X_n is P-dimensional real vector. The scaling is important to guard against variable with larger variance. Training data can be viewed by means of the dividing hyperplane, which takes

$$w.x + b = 0 - - - - - - - - - - - - - - - (4)$$

Where b is scalar and w is p-dimensional vector.

Parallel hyperplanes can be described by equation

$$w.x + b = 1$$

$$w.x + b = -1$$

if the training data are linearly separable then select these hyperplanes so that there are no points between them and then try to maximize their distance. By geometry found the distance between the hyperplane is 2/|w|. It is needed to minimize |w| to excite data points, Need to ensure that for all I either

$$w.x_i - b \geq 1 \; or \; w.x_i - b \leq -1$$

This can be written as

$$y_i(w.x_i - b) \geq 1\,, 1 \leq i \leq n - - - - - - -(5)$$

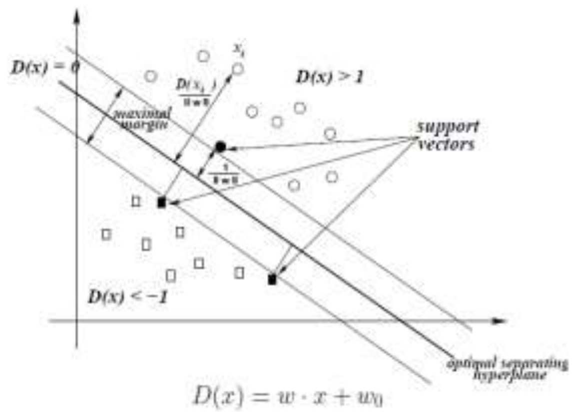


Figure 2.1: Support Vector Machine

3.1 KERNEL SELECTION

Training vectors x_i are mapped into higher dimensional space by the function Φ. Then SVM finds a linear separating hyperplane with maximal margin this higher dimensional space. C>0 is the penalty parameter of the error term [8].

Furthermore, $K(X_i, X_j) = \emptyset(X_i)^T \emptyset(X_j)$ is called the kernel function. There are many kernel functions in SVM, so how to select a good kernel function is also research issue. However, for general purposes, there are some popular kernel functions:

Linear kernel:

$$K(X_i, X_j) = {X_i}^T X_j - - - (6)$$

Polynomial kernel:

$$K(X_i, X_j) = (\gamma {X_i}^T X_j + r)^d, \gamma > 0 - - - -(7)$$

RBF kernel:

$$K(X_i, X_j) = \exp(-\gamma \parallel X_i - X_j \parallel 2)\,, \gamma > 0 - - - (8)$$

Sigmoid kernel:

$$K(X_i, X_j) = \tanh(\gamma {X_i}^T X_j + r) - - - - - (9)$$

Here, γ, r and d are kernel parameters.

in these popular kernel functions, RBF is the main kernel function because of following reasons:

- The RBF kernel nonlinearly maps samples into a higher dimensional space unlike to linear kernel.
- The RBF kernel has less hyper parameters than the polynomial kernel.
- The RBF kernel has less numerical difficulties.

3.1.1 SVM EXECUTION PROCESS:

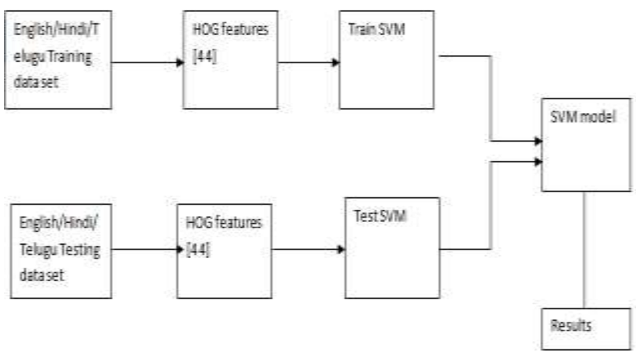


Figure3.1: overview of SVM process

3.2 NEURAL NETWORKS

Neural networks with their remarkable ability to derive meaning from complicated or imprecise data, can be used to extract patterns and detect trends that are too complex to notice by either humans or other computer technique. A trained neural network can be thought of as an "expert" in the category of information it has been given to analyse. The neural networks consist the three layers, input layer, hidden layer, output layer [9].

A unit in the output layers its activity by the fallowing three steps.

1. First it computes the total weighted input x_j, using the formula, $x_{j}=\sum_i y_{i} w_{ij}$

2. The unit calculates the activity y_j using some function of the total weighted input. $y_{i}=\frac{1}{1+e^{-x_j}}$

3. Once the activities of all output units have determined, the network computes the error E. $E=\frac{1}{2}\sum_i (y_i-d_i)^2$. [10][11]

3.2.1 NN EXECUTION PROCESS:

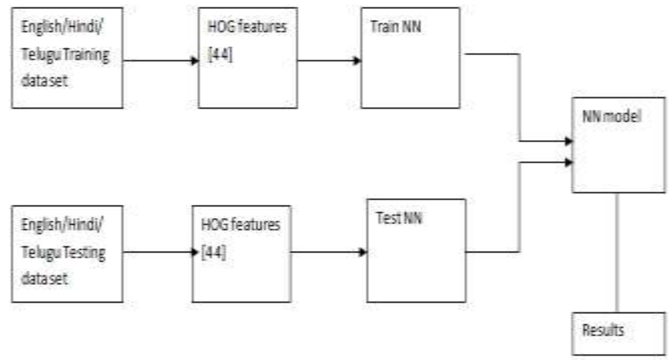


Figure3.2: overview of NN process

4. EXPERIMENTAL RESULTS

Here in this section experiment results are discussed. Testing of experiments is done on all the letters from small a to z, same as the capital letters in English, Hindi, and Telugu languages here some of the results are given

4.1 Results for HOG:

The HOG gives the features of the text presented in the image with some of the results. Internally the HOG features can be represents like as the figure 4.1, based on the pixel value it concatenate all the pixels, for these experiment cellsize [4 4] is better one.

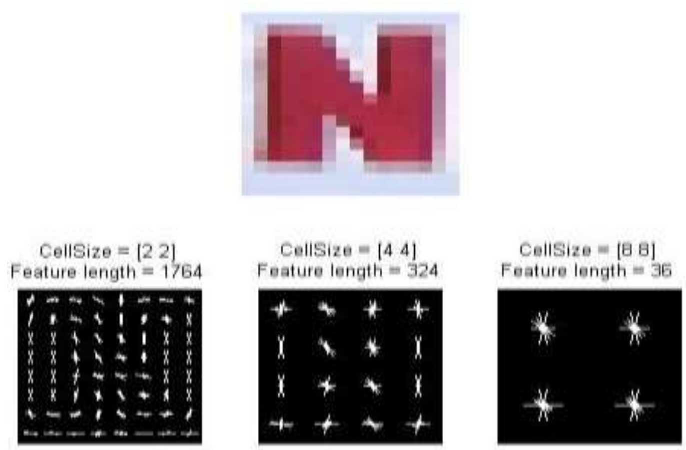


Figure 4.1: HOG features for 'N'

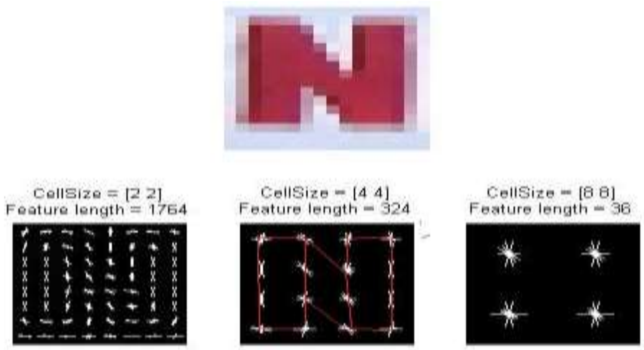


Figure 4.2: internal representation of ‘N’

In these results there are three cell sizes 2*2, 4*4, and 8*8. It selects the best hog features, similarly features are extracted for the all the small, capital letters and for the digits, some of the features shown as

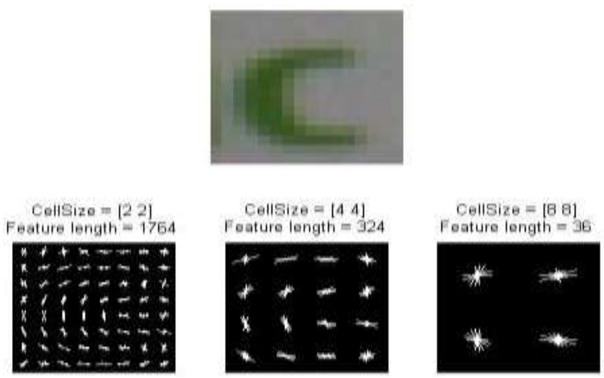


Figure 4.3: HOG features for small letter ‘c’

Similarly the HOG features for the small letter ‘c’ observe the red line it combine all the same pixel values.

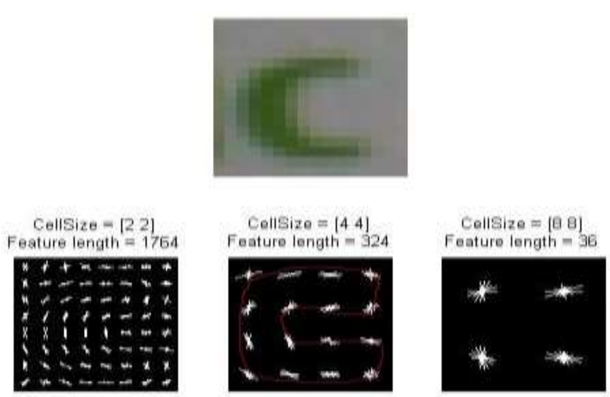


Figure 4.4: internal representation of small ‘c’

Telugu letters:

Telugu is one of the Indian regional language, it consist so many letters, consider 49 letters for the work the following figure is one of the example.

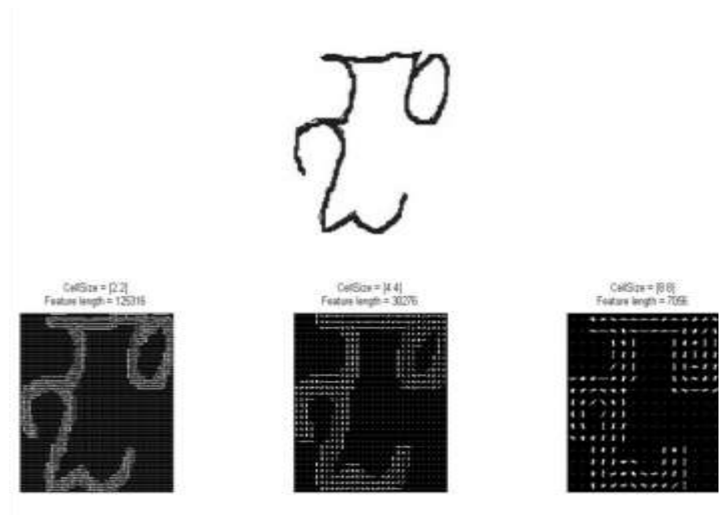


Figure 4.5: Hog features for Telugu letter

Three types of hog results are obtained for the cell sizes [2 2], [4 4], [8 8] respectively, then choose the [4 4] cell size due to its better appearance.

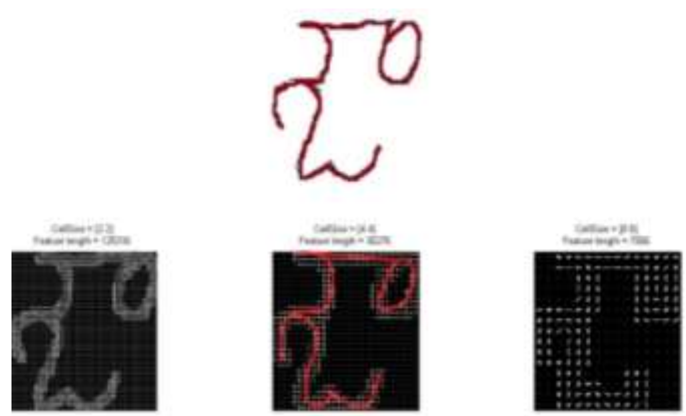


Figure 4.6: internal representation for the HOG features

Hindi Letters:

Hindi is the mostly used language in India it consists different letters which has 47 letters. This work can be useful in the language translation process.

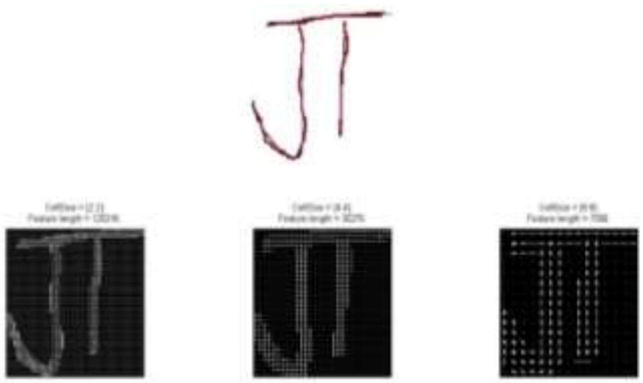


Figure 4.7: Hog features for Hindi letter

The HOG features are extracted in the image it shows three types of cell sizes, so choose the [4 4] cell size for the better appearance

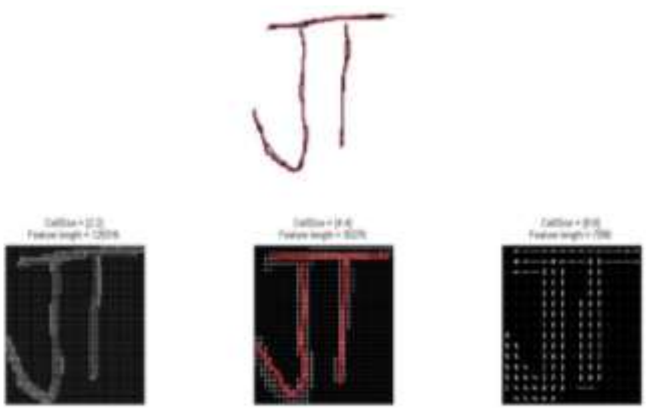


Figure 4.8: internal representation for the HOG features

Results for Small letters using SVM:

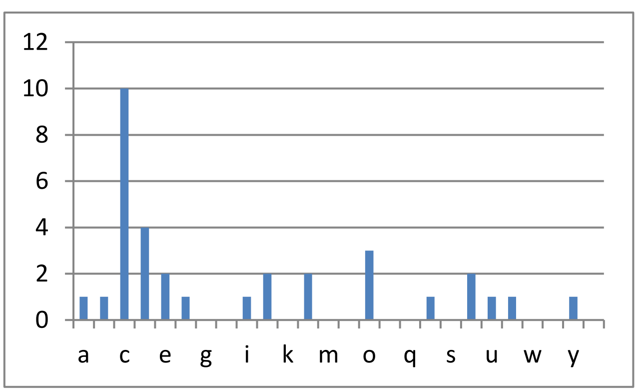
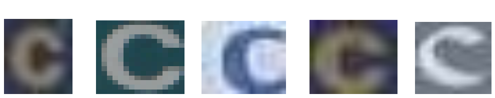


Figure 4.9: results for small letter ‘c’



The figure 4.9 shown the results for letter ‘c’ it gives the best results as observed in the chart, which have 10 testing images and 100 training images ‘c’ gives exact results in maximum cases.

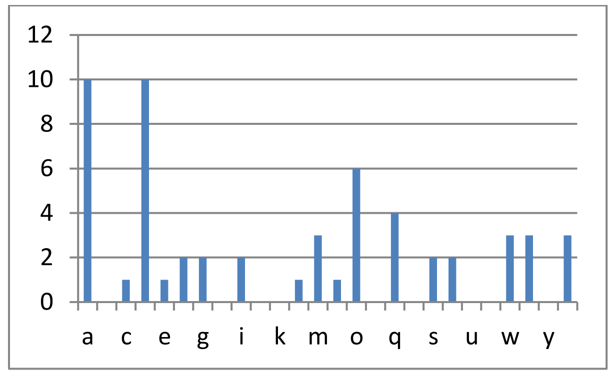


Figure 4.10: results for letter‘d’

Those images are examples in the data set which has the nearest representation. In the figure 4.10 the results for‘d’ is appeared as ‘a’ in most situation because of they have same appearance in that case it gives the worst results.

Results for capital letters using SVM:

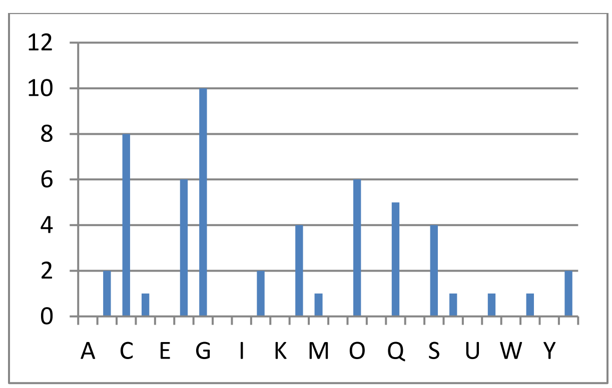


Figure 4.11: results for capital letter ‘G’

The letter ‘G’ has appeared like ‘C’ in some cases but it gives the good results in the experiment.

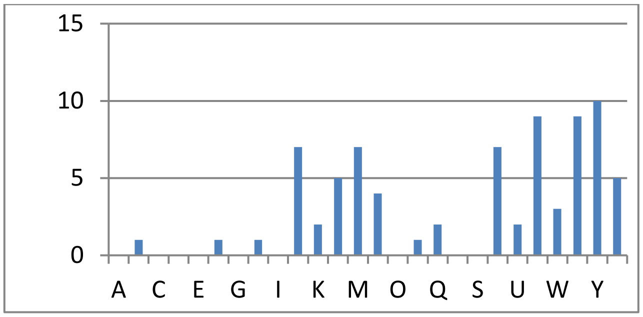
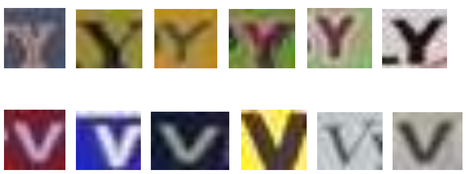
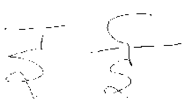


Figure 4.12: results for capital letter ‘Y’



Here, the ‘Y’ gives the worst results in some cases it appears like ‘V’ and ‘X’.

SVM Results for Hindi language:



This two are the two different letters in the Hindi language but the representation of those letters is very closer. All the difficulties are identified in all letters the results for these two letters are shown in the figure.

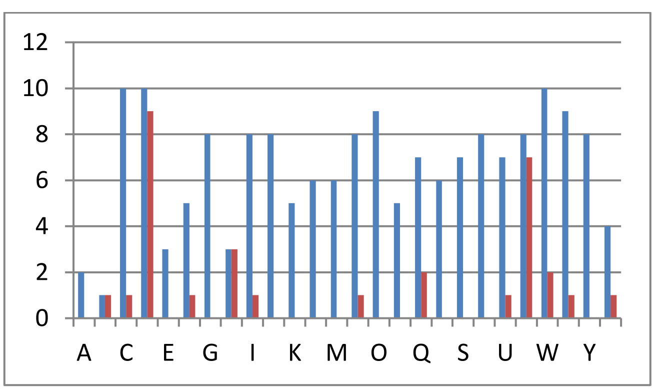
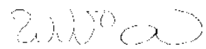


Figure 4.13: SVM results for Hindi letters

SVM Results for Telugu language:



The same way worked on the Telugu letters and took two letters the two are very different from each other it gives some best results as shown in the figure, this process is helpful in the language translation.

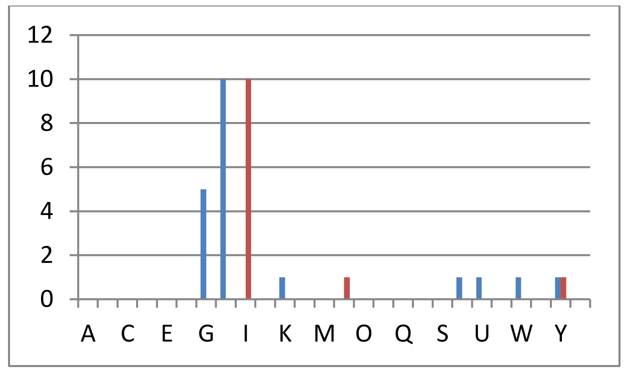


Figure 4.14: SVM results for Telugu letters

NN Results for English letters:

Here comparison is done between the SVM and NN results to know the better results. In the NN used fitting app with the help of Scaled Conjugate Gradient method because it gives the good performance on the large data sets. The results for these methods are shown here.

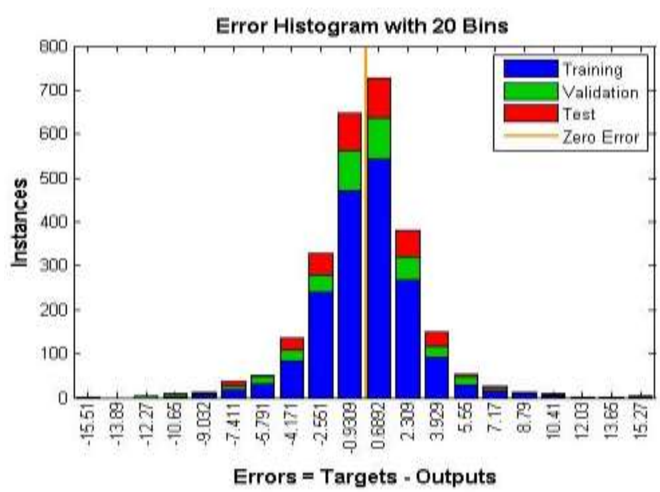


Figure 4.15: NN results for the capital letters

In the figure it has three types of data training, validate and test data. It selects the data randomly as 70% of the dataset consider as training and 15% respectively for the test and validate data. it gives the zero error at the 0.6892, in the NN consider the 10 hidden layers, Similarly done for the both small letters and digits.

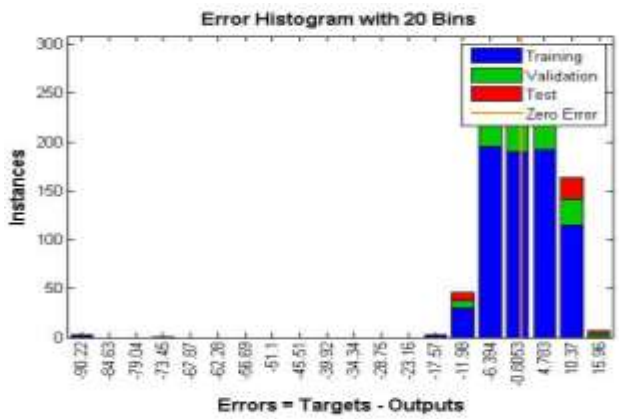


Figure 4.16: NN results for the small letters

The results for small letters shown in the Figure 4.16, and got the zero error at the -08053.

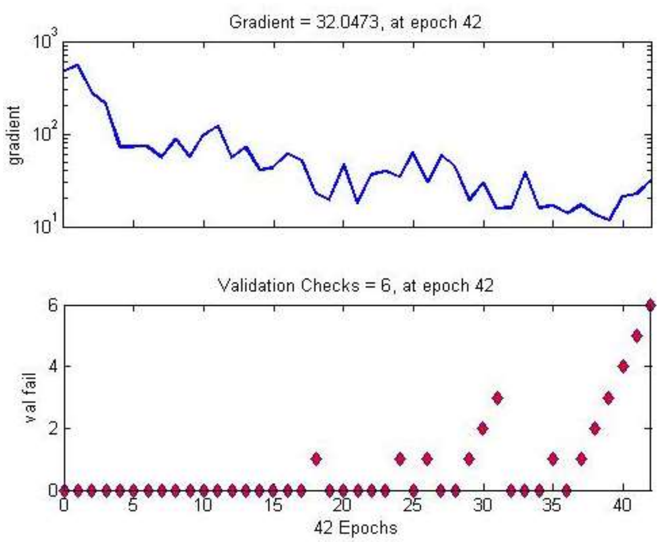


Figure 4.17: validation checks for capital letters

Totally obtained the 2600 images for all the capital letters, for these the experiment gives 6 validation checks, with the 42 Epochs, and 32.043 gradient.

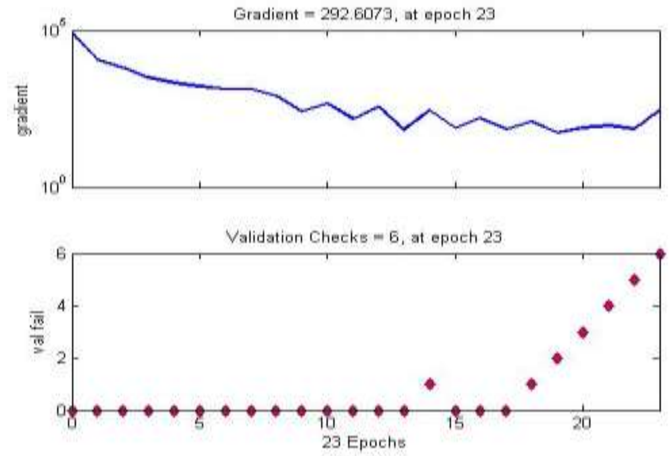


Figure 4.18: validation checks for small letters

Similarly obtained 1024 images for small letters and got the 6 validation checks, 23 epochs and 292.6073 gradient.

NN Results for Telugu letters:

Applying same procedure to Telugu letters the total data set divided into three types training, testing and validation data with the amount of 70%, 15% and 15% respectively. The dataset is given to the NN fitting app for checking the results. The zero error rate comes at the 0.5318.

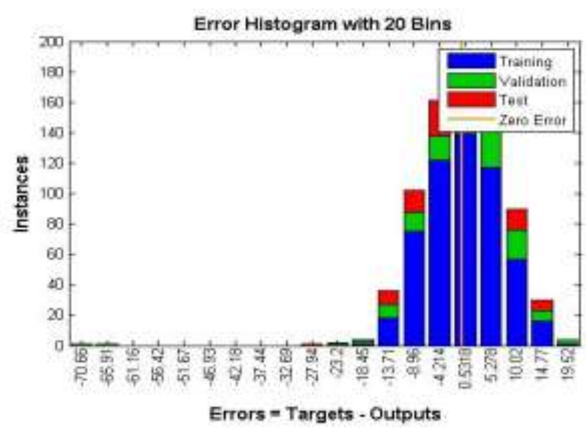


Figure 4.19: NN results for Telugu letters

The dataset contains the totally 49 letters each letter contain 30 images, for those all images got the 6 validation checks with the gradient 779.6633 at epoch 53

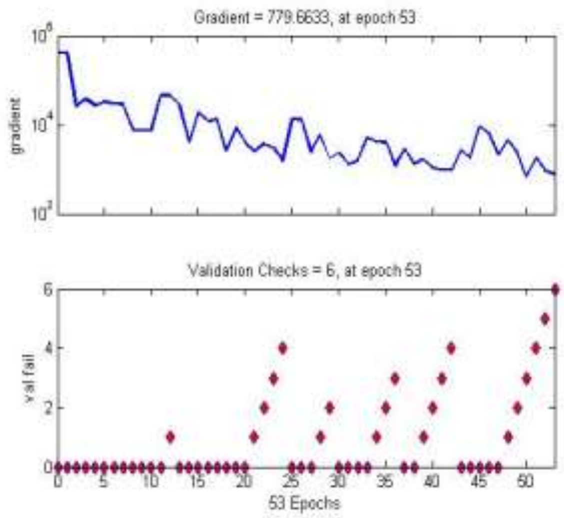


Figure 4.20: validation checks for Telugu letters

NN Results for Hindi letters:

Similarly the Hindi data set contains 47 letters and 30 images each one the appropriate results shown in the figures, the error rate comes at the position -1.12 and the gradient at 350.6147 with six validation checks at epoch 20.

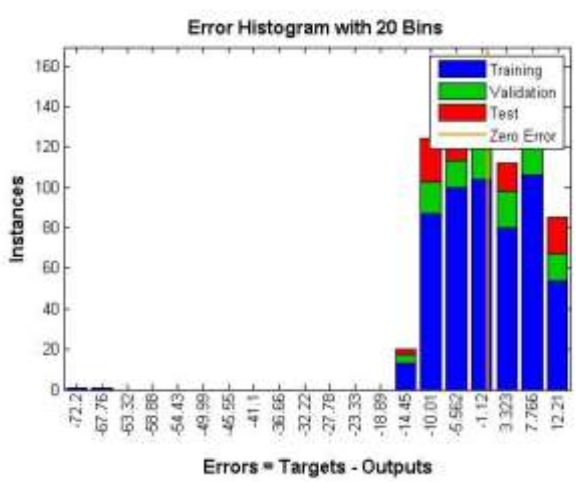


Figure 4.21: NN results for Hindi letters

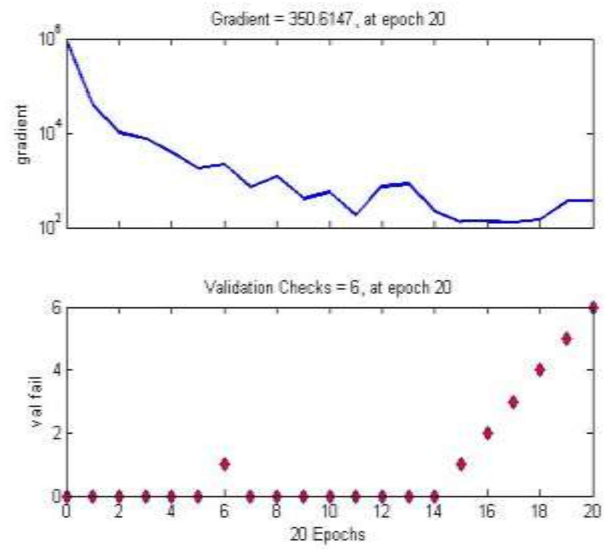


Figure 4.22: validation checks for Hindi letters

COMPARISSION OF HOG AND NN

After getting the both results compared them to identify which gives the better results to this work.

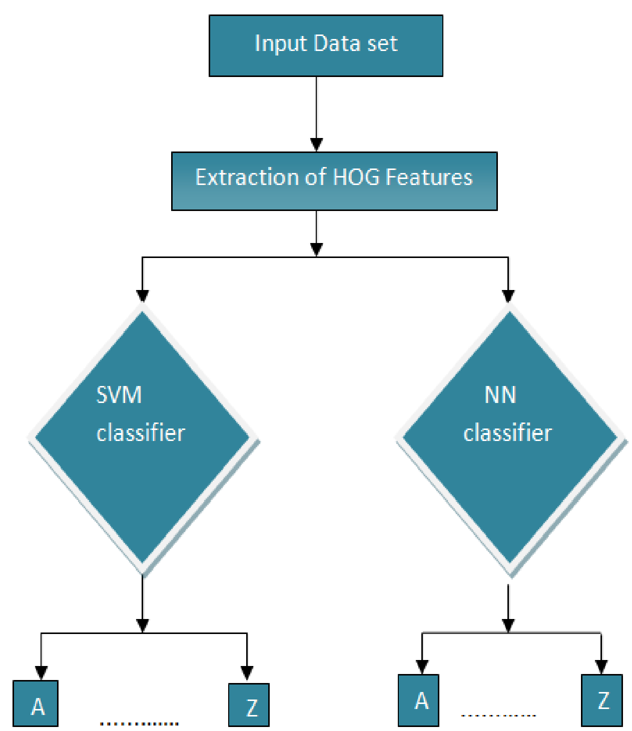


Figure 4.23: Overview of text detection and recognition

The overall results shown in the figure series1 indicates SVM results and series2 indicates NN results, from these results known that SVM works well to identify the letters in different languages.

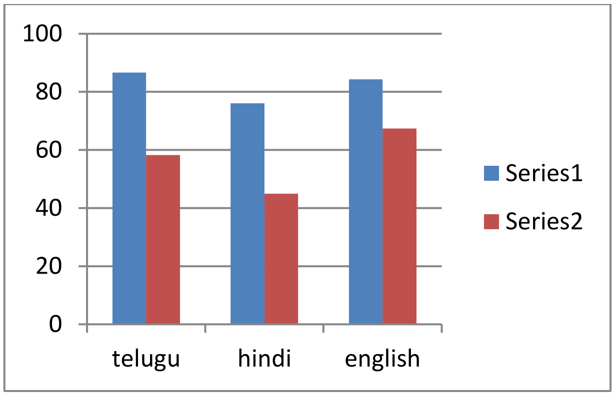


Figure 4.24: Results comparison for all languages

5. CONCLUSION

Proposed a novel methodology to recognize characters from scene images using HOG features and machine learning techniques. Firstly, the algorithm extracts HOG features, then using these features SVM and NN models are trained. Subsequently compared the results of SVM and NN and found that SVM performs better for character recognition. The algorithm is applied for English Hindi and Telugu. As a future work, this work can be extended to test various languages.

REFERENCES

[1] Neumann, Lukas, and Jiri Matas. "Real-time scene text localization and recognition." 2012 IEEE Conference on Computer Vision and Pattern Recognition. IEEE, 2012.

[2] Bartz, Christian, Haojin Yang, and Christoph Meinel. "STN-OCR: A single neural network for text detection and text recognition." arXiv preprint arXiv:1707.08831 (2017).

[3] Satpathy, Amit, Xudong Jiang, and How-Lung Eng. "Human detection by quadratic classification on subspace of extended histogram of gradients." IEEE Transactions on Image Processing 23.1 (2013): 287-297.

[4] Newell, Andrew J., and Lewis D. Griffin. "Multiscale histogram of oriented gradient descriptors for robust character recognition." 2011 International conference on document analysis and recognition. IEEE, 2011.

[5] Roe, Edward, and Carlos AB Mello. "Binarization of color historical document images using local image equalization and XDoG." 2013 12th International Conference on Document Analysis and Recognition. IEEE, 2013.

[6] Dlagnekov, Louka, X. Chen, and A. L. Yuille. "Detecting and Reading Text in Natural Scenes." (2004): 1-22.

[7] Gao, Lianru, et al. "Subspace-based support vector machines for hyperspectral image classification." IEEE Geoscience and Remote Sensing Letters 12.2 (2014): 349-353.

[8] Al-Omari, Saleh AK, et al. "Digital recognition using neural network." Journal of computer science 5.6 (2009): 427.

[9] Ciregan, Dan, Ueli Meier, and Jürgen Schmidhuber. "Multi-column deep neural networks for image classification." 2012 IEEE conference on computer vision and pattern recognition. IEEE, 2012.

[10] De Campos, Teófilo Emídio, Bodla Rakesh Babu, and Manik Varma. "Character recognition in natural images." VISAPP (2) 7 (2009).

[11] Wang, Xiaobing, et al. "End-to-end scene text recognition in videos based on multi frame tracking." 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR). Vol. 1. IEEE, 2017.