Comparative Study Of Preprocessing And Classification Methods In Character Recognition Of Natural Scene Images

Yash Sinha, Prateek Jain, and Nirant Kasliwal

Birla Institute of Technology and Science
Department of Computer Science and Information Systems
Pilani, India, 333031.

Abstract. This paper presents an approach to character recognition in natural scene images. Recognizing such text is a challenging problem in the field of Computer Vision, more than the recognition of scanned documents due to several reasons.
We propose a classification technique for classifying characters based on a pipeline of image processing operations and ensemble machine learning techniques. This pipeline tackles problems where Optical Character Recognition (OCR) fails. We present a framework that comprises of a sequence of operations such as resizing, gray scaling, thresholding, morphological opening and median filtering on the images to handle background clutter, noise, multi-sized and multi-oriented characters and variance in illumination. We used image pixels and HOG (Histogram of Oriented Gradients) as features to train three different models based on Nearest Neighbour, Random Forest and Extra Tree classifiers.
When the input images were pre-processed, HOG features were extracted and fed into Extra Tree Classifier, the model classified the characters with maximum accuracy, among the other models that we tested. The proposed steps have been experimentally proven to yield better accuracy than the present state of the art classification techniques on the Chars74k dataset.
In addition, the paper includes a comparative study elaborating on various image processing operations, feature extraction methods and classification techniques.

Key words: Camera based Character Recognition, Histogram of Oriented Gradients, Feature Extraction, Scene text Recognition, Ensemble classifiers

1 Introduction

We focus on recognition of individual characters from natural scene images. It is a field of active research with high utility and potentially high impact applications. Unlike scanned grayscale images, natural scene images have characters with additional colour information and noise.

The challenge arises due to variation in fonts, styling, formatting, background clutter, size and orientation of characters in addition to background noise and occlusion. Additionally, low resolution, uneven illumination, glare, gloss and artistic format make the classification error prone. These are the reasons that commercial OCR engines have dismal performance on natural scene images.

Our proposed methodology tackles these challenges by binarization, morphological opening and feature extraction. The extracted features are used as input for the classification models. The 62 classes into which the images can be categorized are 0-9, a-z and A-Z.

The performance and accuracy achieved is far better than optical character recognition systems. This yielded better accuracy than the present state of the art technique. [1]

2 Related Work

Recent papers on character recognition in natural scenes have proposed multiple techniques of image processing, feature extraction and classification models. The problem lies on the common intersection of computer vision and machine learning. Campos et al. [2] introduced Chars74k dataset of characters collected from natural scene images. The work highlighted that commercial OCR engines do not provide satisfactory results for natural scene character images. Their experiment was based on methods of Multiple Kernel Learning, Nearest Neighbour Classification and Support Vector Machines. Sheshadri et al. [3] proposed the exemplar based multi layered classification approach to recognize characters. They proposed a methodology using SVM to classify images on features from Histogram of Oriented Gradients (HOG) [4] on resized and warped images. Zhang et al. [1], although used a mixed dataset of natural scene images, handwritten characters and digital text images, achieved an accuracy of 72.68%. Fraz et al. [5] have recently proposed a pipeline of image processing operations for identification of characters. They explored efficacy of colour information and variation for the identification of characters from an image to give accuracy of 72% for 49 classes. Kumar et al. [6] applied Discrete Cosine Transform (DCT) for character recognition after binarization to remove noise for English and Kannada character recognition. kNN was used as the classification model. Neumann et al. [7] achieved a word recognition rate of 72% using graph based approaches in Maximally Stable Extremal Regions (MSER). Machine learning model used was a SVM with RBF. Proposed methods focus on individual character recognition with techniques for image processing, feature extraction and machine learning techniques on the images.

3 Dataset

The data is taken from the Chars74K dataset [8], which consists of images of characters selected from Google Street View images *(EngImg)*, as shown in Figure 1 . Street view images consist of English characters taken in natural contexts

such as street signs, house names and numbers, shop signboards and so on. The data set differentiates between digits, uppercase and lowercase characters to give a total of 62 classes. Individual characters were manually segmented from these images. Our dataset has 12503 characters, of which 6283 (50.25%) are randomly sampled for training while rest 6220 (49.75%) are used as test data. There is significant variation in thickness, size, alignment, orientation and position of characters both within the class and across the classes. Handwritten and synthetic images were not used.

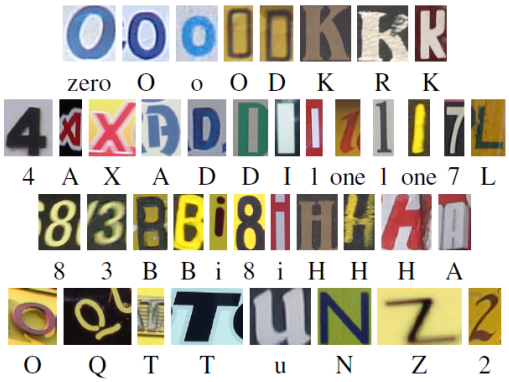

Fig. 1. Examples of images from dataset

4 Experiments And Results

Proposed Methodology: The proposed methodology consists of four steps:

1. Image processing – *tackles background clutter, fonts and illumination of images*: This step includes converting all images to a common standard size, binarization using Otsu's Thresholding followed by Morphological opening and median filter for removing salt and pepper noise.
2. Feature extraction – *addresses variation in character size and orientation*: Histogram of Oriented Gradients is used as a method of feature extraction. The value of used cell size is calibrated experimentally to an optimal value. It also reduces the number of features in comparison to using direct pixel values.
3. Training the classifier – *does the learning for classifier models and classifies test images into classes*: kNN, Random Forest and Extra Tree classifier are tested and compared. The number of estimators are determined experimentally to an optimal value. The input parameters HOG features and pixel values both are tested and compared.

4. Performance testing – *compares various classification and feature extraction techniques*: Cross-validation is performed and the classification model is tested on the test data set. Accuracy obtained from both the methods is mentioned.

MATLAB [9] and Python's scikit-learn library [10] is used for the above exploration and experimentation.

4.1 Image Processing

Resizing and Gray scaling To have a consistent feature set, all the images were gray scaled and resized to 75x75 pixels.

The work of Fraz demonstrated that processing colour information is an additional challenge. In order to tackle this, images were gray scaled by using a weighted sum of the R, G, and B components.

The images were enlarged using bi-cubic interpolation, where the output pixel value is a weighted average of pixels in the nearest 4-by-4 neighbourhood. Enlarging the image led to extraneous edge detection. Accuracy improved because more features increased variability in the data.

The size of images was reduced using antialiasing. Visual inspection showed that reducing the image size led to noise suppression in in ensuing edge detection.

Thresholding Otsu's method for thresholding [11] is an image processing technique for image segmentation. The algorithm calculates the optimum threshold separating the two classes (foreground/object and background) so that their combined spread is minimal. Separating the foreground and background pixels is necessary due to high variance and noise in the data set.

We tried basic thresholding, average thresholding and Otsu thresholding techniques (Figure 2). In comparison with average thresholding, we observed Otsu retains much more information about local changes in the pixel values.

Fig. 2. Comparison of thresholding techniques: No thresholding, Average thresholding, Otsu thresholding

Morphological Operations Morphological operations rely only on the relative ordering of pixel values and are especially suited to the processing of binary images.

The image was opened [12] using a disc of 1 pixel radius. In our particular case, this removed the salt and pepper noise in addition to smoothing the edge and filling the gaps within the character strokes (leaving the holes unfilled). (Figure 3)

Fig. 3. Comparison of morphological operations: Original image, Average thresholding, Morphological Opening

Noise Reduction Median filter showed better results than the averaging filter, as expected [13] for noise reduction. Median filter yields enhanced performance because it preserves edge details better than mean filter. It allows high spatial frequency detail to pass while remaining very effective at removing noise. [14]

4.2 Feature Extraction

Image Pixels We represented the 2D image matrix as single dimensional vector with pixels as features for classification. This gave 5625 features.

Histogram of Oriented Gradients HOG creates histograms of edge orientation to compute the gradients of the edges from patches of the image. HOG as feature extraction technique is it is independent of variation in orientation, hence is advantageous.

HOG is an iterative process, where a cell of nxn pixels is extracted from the image and algorithm is applied to calculate the histogram of oriented gradients. These gradient values are used as features of the image.

A series of values for n was applied. In HOG, with increasing value of n, there is decrease in the number of features in the image (Figure 4). To capture large-scale spatial information, we increased the cell size. However, for n>12, there was loss of small scale detail rather than extraction of useful features, due to submerging of features derived from larger cells. Smaller values of n make the classifier susceptible to noise. Experimentally, n = 12 stood out to be an optimal value to use.

Smaller block size helps to capture the significance of local pixels and suppress illumination changes of HOG features. Hence, a block of size 2x2 was used.

To ensure adequate contrast normalization, an overlap of at least half the block size: 1x1 was selected. Large overlap values can capture more information, but they produce larger feature vector size.

To achieve a trade-off between finer orientation details and size of the feature vector, number of orientation histogram bins were selected as 9. This gave 900 features.

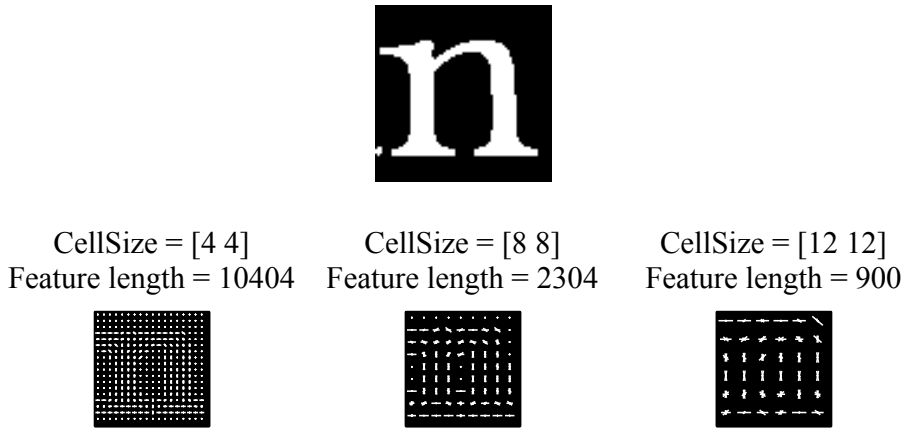

Fig. 4. Visualization of HOG features

4.3 Classification Techniques

K Nearest Neighbour (kNN) Taking the one-dimensional n bit vector image as a point in n dimensional hyperspace, we applied kNN [15] for classification using *neighbors* sub-module of Python's scikit-learn library [16]. Value of k was varied from 1 to 9 to select appropriate k value.

The kNN model used for both HOG and Image pixels, use Euclidean distance to calculate distance. The weight for all k obtained points in kNN had uniform weights.

Random Forest Classifier It is a meta-estimator [17] based on subsampling which controls over-fitting well. The *ensemble* sub-module of Python's scikit-learn library was used [18].

The model computed 1025 trees (estimators), using GINI criterion for splitting tree nodes and applied ensemble technique to make final prediction. The value for number of estimators was obtained using grid-search [19]. It was fed with both HOG and Image pixels and prediction accuracy was computed.

Extra Tree Classifier An estimator [20] that fits a number of randomized decision trees (a.k.a. extra-trees) on various sub-samples of the dataset and uses averaging to improve the accuracy and control over-fitting.

This classifier gave the best results over others. It is a part of the *ensemble* sub-module [21] of Python's scikit-learn library. The model used 1025 trees (estimators). The value for number of estimators was obtained using grid-search algorithm [19]. In extra tree classifier, a set of attributes were selected randomly and GINI criterion was used for splitting nodes. It was fed with both HOG and Image pixels and prediction accuracy was computed.

4.4 Performance Testing

The Chars74k dataset was partitioned into two sets. The model was trained over 50.25% of the dataset. The rest of the images were used to test the model and calculate the estimator performance. We performed 10 fold cross validation on training set to improve on training efficiency.

4.5 Results

We fed the features obtained from HOG (preceded by other processing) into Extra Tree classifier and tested our model using 10-fold cross validation and test data. Our proposed pipeline accurately classified 73.167% (Table 2)of the character images in the test data set, which beats the current state of the art technique [1] as shown in Table 1. On cross validation, 73.022% accuracy was obtained as shown in Table 3.

Table 1. Accuracy comparison with other methods

Method	Accuracy	Dataset	Classes
Proposed Method	**73.167%**	**Chars74K**	**62**
Zhang et al. [1]	72.68%	Chars74K(mixed)	62
Fraz et al. [5]	72%	Chars74K-15	49
Shi et al. [22]	69.9%	Chars74K-15	62
Newell et al. [23]	66.5%	Chars74K-15	62
Campos et al. [2]	55.26%	Chars74k	62

Table 2. Accuracy across different features and classifiers on test data

Features	kNN(k = 4)	Random Forest	Extra Tree
HOG Features	69.549%	71.575%	**73.167%**
Image Pixels	55.659%	62.636%	64.244%

Table 3. Cross Validation scores across different features and classifiers

Features	kNN(k = 4)	Random Forest	Extra Tree
HOG Features	69.220%	71.590%	**73.022%**
Image Pixels	55.120%	62.013%	63.570%

5 Observations And Trends

A Comparative Study:
A comparative study elaborating on various image processing operations, feature extraction methods and classification techniques is presented here. Certain trends of the exploration have been discussed.

5.1 Image Pixels as feature set vs. HOG features

Edge orientations of images give better features to the training model as compared to image pixels. HOG features helps reduce variations arising from orientation and size despite sensitivity to non-random noise. While using HOG features with Extra Tree classifier, the accuracy increased by 10% on 10 fold cross validation scores and 9% for test data. (Figure 5)

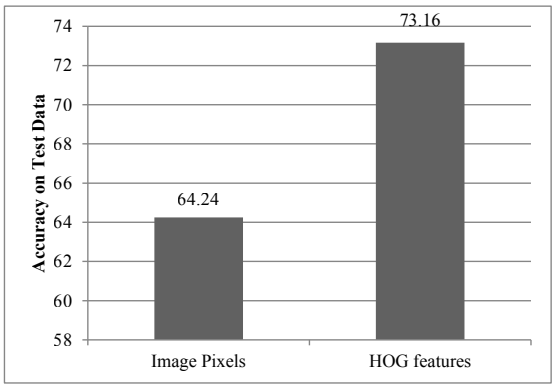

Fig. 5. Image Pixels as feature set vs. HOG features

5.2 Average Thresholding vs Otsu Thresholding

Different thresholding algorithms approach noise removal in images differently. Otsu's thresholding improved the accuracy in comparison to average thresholding by retaining more information in high noise environments. (Figure 6)

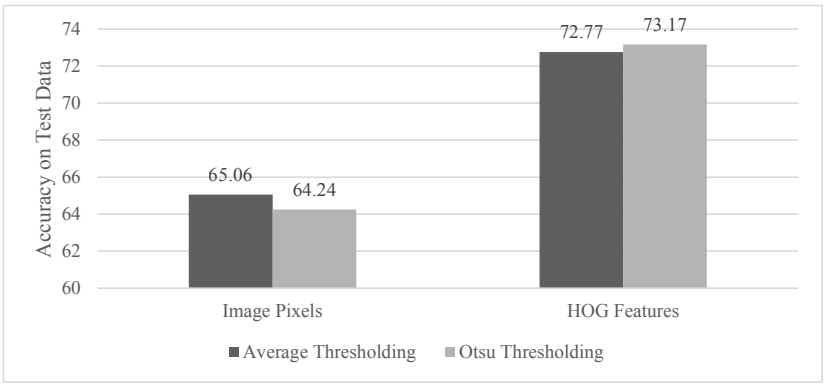

Fig. 6. Thresholding Trends on Image Pixels and HOG features as feature set

5.3 K Nearest Neighbours (1NN to 9NN)

The graph shows scores of k Nearest Neighbours classifier with different values of k on test data. Clearly 1NN and 4NN outperform other k values. (Figure 7 and 8)

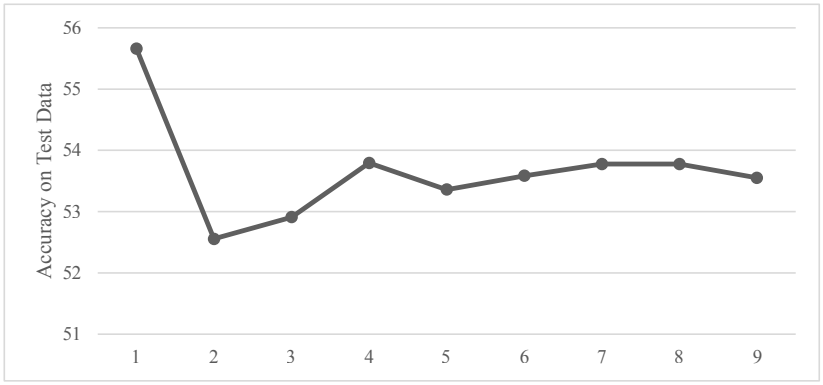

Fig. 7. K Nearest Neighbour with different values of k on Image Pixels as feature set

5.4 K Nearest Neighbours vs Random Forest vs Extra Tree

Ensemble methods outperform k Nearest Neighbours significantly. Extra tree classifiers usually have trees with greater height than random forest. This implies that Extra Tree can generalize better but is computationally more expensive in comparison. In our case extra tree classifier gives a noteworthy improvement over random forest classifier. (Figure 9)

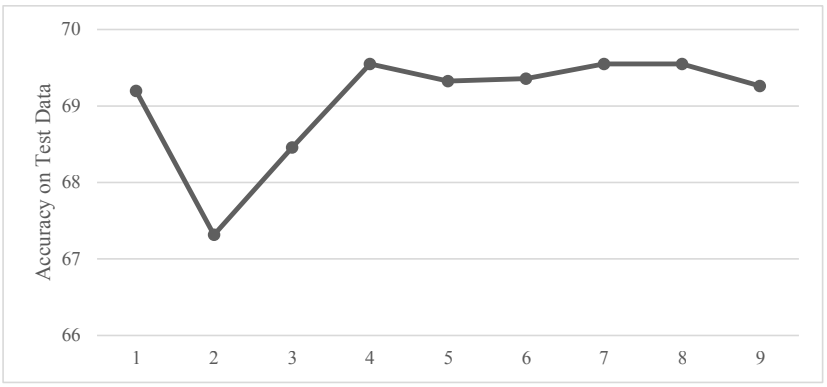

Fig. 8. K Nearest Neighbour with different values of k on HOG features

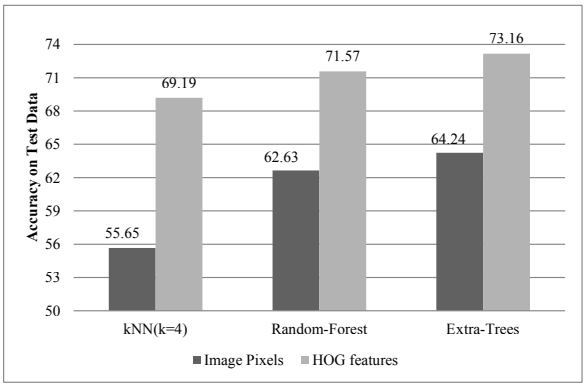

Fig. 9. Comparison of accuracies of different classification techniques (on both feature inputs)

6 Conclusion And Future Prospects

In summary, we have proposed an effective method to recognize scene characters. Factors like background clutter, noise, multi-sized and multi-oriented characters and variance in illumination make character recognition from scene images a challenging task. Our model incorporates operations in the pipeline in an order which is crucial to account for these factors.

Our results show that scene text characters can be recognized with a good accuracy in natural images, although there is scope for improvement.

Variation in thickness, illumination, geometric distortions arising from camera angles as well as writing variations, occlusion and non-random noise reduce effectiveness of feature extraction of proposed method. Kernel Regression for Image Processing and Reconstruction is a less investigated yet intriguing idea. Shape context can be possibly used for better feature extraction. In the machine learning pipeline, boosting and dimensionality reduction can be applied to improve accuracy. Convolutional neural networks is also a promising alternative.

This work can further help in building vision systems, which can solve higher level semantic tasks, such as word recognition and scene interpretation.

References

1. Zhang, Z., Sturgess, P., Sengupta, S., Crook, N., Torr, P.: Efficient discriminative learning of parametric nearest neighbor classifiers. (2012)
2. de Campos, T.E., Babu, B.R., Varma, M.: Character recognition in natural images. In: Proceedings of the International Conference on Computer Vision Theory and Applications, Lisbon, Portugal. (2009)
3. Sheshadri, K., Divvala, S.K.: Exemplar driven character recognition in the wild. In: BMVC. (2012) 1–10
4. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on. Volume 1., IEEE (2005) 886–893
5. Fraz, M., Sarfraz, M.S., Edirisinghe, E.A., Loughborough, U.: (Exploiting colour information for better scene text recognition)
6. Kumar, D., Ramakrishnan, A.: Recognition of kannada characters extracted from scene images. In: Proceeding of the workshop on Document Analysis and Recognition, ACM (2012) 15–21
7. Neumann, L., Matas, J.: A method for text localization and recognition in real-world images. In: Computer Vision–ACCV 2010. Springer (2011) 770–783
8. deCampos, T.: The Chars74k Dataset. http://www.ee.surrey.ac.uk/CVSSP/demos/chars74k/ (2009) [Online; accessed 5-December-2014].
9. 8.3, M., Release, C.V.S.T., 2014a, I.P.T.R.: version 8.3 (R2014a). The MathWorks Inc., Natick, Massachusetts (2014)
10. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al.: Scikit-learn: Machine learning in python. The Journal of Machine Learning Research **12** (2011) 2825–2830
11. Otsu, N.: A threshold selection method from gray-level histograms. Automatica **11** (1975) 23–27

12. Gonzalez, R.C., Woods, R.E.: Morphological image processing. In: Digital Image Processing. Addison-Wesley Longman Publishing Co.,Inc., Boston, MA, USA (2001) pp.635–639
13. Gonzalez, R.C., Woods, R.E.: Order-statistic (nonlinear) filters, example 3.14. In: Digital Image Processing. Pearson Education (2008) pp.157
14. Lim, J.S.: Two-dimensional signal and image processing. Englewood Cliffs, NJ, Prentice Hall, 1990, 710 p. **1** (1990)
15. Cover, T., Hart, P.: Nearest neighbor pattern classification. Information Theory, IEEE Transactions on **13** (1967) 21–27
16. Scikit-learn: sklearn.neighbors.KNeighborsClassifier Scikit-learn. (http://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html) [Online; accessed 5-November-2014].
17. Breiman, L.: Random forests. Machine learning **45** (2001) 5–32
18. Scikit-learn: sklearn.ensemble.RandomForestClassifier. (http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html) [Online; accessed 5-November-2014].
19. Scikit-learn: Grid Search: Searching for estimator parameters. (http://scikit-learn.org/stable/modules/grid_search.html) [Online; accessed 5-November-2014].
20. Geurts, P., Ernst, D., Wehenkel, L.: Extremely randomized trees. Machine learning **63** (2006) 3–42
21. Scikit-learn: sklearn.ensemble.ExtraTreesClassifier - Scikit-learn. (http://scikit-learn.org/stable/modules/generated/sklearn.tree.ExtraTreeClassifier.html) [Online; accessed 5-November-2014].
22. SHI, C.Z., WANG, C.H., XIAO, B.H., ZHANG, Y., GAO, S.: Multi-scale graph-matching based kernel for character recognition from natural scenes. Acta Automatica Sinica **40** (2014) 751–756
23. Newell, A.J., Griffin, L.D.: Natural image character recognition using oriented basic image features. In: Digital Image Computing Techniques and Applications (DICTA), 2011 International Conference on, IEEE (2011) 191–196