Volume 66, Issue 2, 2022
Journal of Scientific Research
of
The Banaras Hindu University

A deep learning framework for handwritten Ol Chiki character recognition

Debaditya Barman*[1], Tuheli Bhattacharya[2], and Nirmalya Chowdhury[3]

*[1]Department of Computer and System Sciences, Visva-Bharati, India, debadityabarman@gmail.com
[2]Department of Computer Science & Engineering, Jadavpur University, India, tuheli.bhattacharya@gmail.com
[3]Department of Computer Science & Engineering, Jadavpur University, India, nirmalya63@gmail.com

***Abstract*—Ol Chiki is an Austroasiatic-Santali language used by the Santhal tribe of India. Despite being one of the official languages of India, Ol Chiki language still remains marginalized. Although there exist significant amount of work on recognizing handwritten characters of several other *mainstream* Indian languages (e.g. Hindi, Bengali etc.), very less number of work have been carried out for this language. In this work, a robust Handwritten Ol Chiki Character Recognition (HOCCR) system based on Deep learning has been proposed. Efficacy of the Histogram of Oriented Gradient (HOG) feature descriptor for recognizing Ol Chiki characters has been shown. A detailed comparative study has been carried out using several proposed prediction models.**

***Index Terms*—Handwritten Character Recognition, Histogram of Oriented Gradient, Convolutional Neural Network, Support Vector Machine.**

I. Introduction

Most of the research works in the area of Handwritten Character Recognition (HCR) are based on the languages which have the most speakers. For instance, a number of research works can be found in the English (1; 2), Japanese (3), Arabic (4; 5), Chinese (6; 7), Bengali (8; 9) languages. The availability of a good quality dataset in these languages could be the motivation. The number of works in other Indian languages like Telegu (10), Meitei Mayek (11), Marathi (12), Odia (13), Kannada (14), Devnagari (15) are comparatively very less.

The Ol Chiki language is one of the 22 Scheduled languages of India [1]. In the year 1920, Pandit Raghunath Murmu first conceptualized the Ol Chiki script and gave it the final shape in the year 1940, providing appropriate writing symbols to the *Santhals* (16). The characters of the script are derived/ inspired from the nature including the five basic elements - fire, soil, water, air, and sky; physical environment and a number of them are inspired from various postures of everyday life. The script is of type alphabetic and is written in left to right direction. It consists of 30 alphabets - 6 vowels and 24 consonants. Fig. 1 presents the Ol Chiki characters. In the Unicode Standard 5.1 [3], the block for Ol Chiki script is from U+1C50 to U+1C7F. However, unlike other scheduled

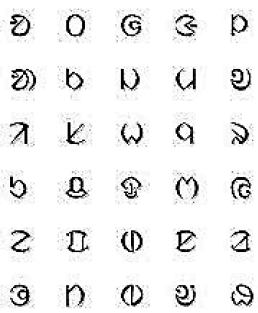


Fig. 1: The Ol Chiki Script.

languages of India, very few works have been done on this language using the techniques from Pattern Recognition and Artificial Intelligence. Daw and Mondal (17) used K Nearest Neighbor (KNN) and Support Vector Machine (SVM) to develop Ol Chiki character and digit recognizer. They achieved 87% accuracy rate. Basu et al. (18) worked on *Santali* speech data. They have considered only six vowels of this language in their speech corpus. Following the raising significance of the language, it was convenient to work on this language so that advancement can be done in the field of technology too. Feature extraction is an important step to develop any HCR system. It extracts important features from the images of the handwritten characters. These features are fed to the machine learning models for classification. Over the last few decades several powerful multipurpose feature descriptors have gained

[1]Distribution of the 22 Scheduled Languages, Census of India, Online Available: www.censusindia.gov.in

[3]The Unicode Standard, Chapter 13.0: Ol Chiki, Unicode Consortium, Online Available: https://unicode.org/charts/PDF/U1C50.pdf

*DOI:*10.37398/JSR.2022.660227

popularity such as HOG (19), "Scale-Invariant Feature Transform" (SIFT) (20), "Speeded-Up Robust Features" (SURF) (21), "Local Binary Patterns" (LBP) (22; 23), etc. Researchers have successfully applied these feature extraction techniques along with Deep learning for various Indian languages: Kannada (24), Bengali (25), Malayalam (26) etc.

A. Contribution

A deep learning based approach to develop an HOCCR system has been proposed here. Our contributions have been summarized below.

- A dataset of handwritten Ol Chiki characters has been built, cleaned, and pre-processed.
- Efficacy of HOG technique for extracting relevant features from the Ol Chiki characters has been established.
- Ideal cell-size of HOG feature descriptor has been determined.
- Four different character recognition systems have been developed. Robustness of these systems have been established using *writer dependent* and *writer independent* test samples.

Rest of the paper has been organized as follows: Section II presents the proposed methodology. In Section III, we have discussed the data collection and data preparation process. The experimental results have been analysed and presented in Section IV. The concluding remarks can be found in Section V.

II. Proposed Methodology

The objective of this paper is to develop a robust HOCCR system. In this work, four different character recognizer systems have been developed and their performance have been compared using two datasets prepared by us. In the first method, a six-layered CNN has been trained and used for both feature extraction followed by classification. Here, softmax layer is responsible for the classification task. In the second method, similar process has been followed. However, the softmax layer of the previous model has been replaced by SVM. Here, SVM has been trained using the CNN features. In the third method, HOG based feature selection technique has been used and SVM acted as the classifier. Instead of using the whole image to train the CNN, HOG features have been used in the fourth method. Sub-section II-A describes the procedure for recognizing handwritten Ol Chiki characters using CNN. The description of a Handwritten Character Recognition system using HOG feature descriptors and a Linear SVM classifier has been discussed in Sub-section II-C and the training process of the CNN with the HOG feature vectors have been discussed in Sub-section II-D.

A. Handwritten Character Recognition using CNN

The development of CNN is motivated by the visual cortex part of the human brain. It has very small regions of cells which are sensitive to specific regions and patterns. The idea of specialized components in-side a system for performing specific tasks (like the neuronal cells in the visual cortex) is the basic idea behind CNN. In technical terms, CNN takes an image as input then passes it through a series of layers to produce an output. In our first proposed method, we have used six different layers. These layers are: the Convolutional Layer, Rectified Linear Unit (ReLU), Max Pooling, Flatten Layer, Dropout Layer, and the Fully Connected Layer. Architecture of the first method is presented in Fig. 2. The prediction model obtained from our first method has been named as the **CNN Model**. The input layer of our proposed model takes an input

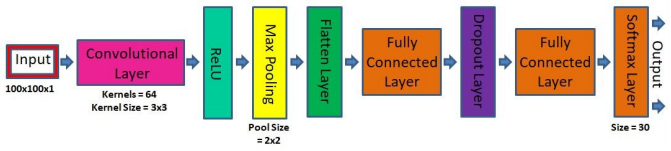


Fig. 2: The CNN Based Architecture.

sample of 1001001. The dimensions of the input layer are height, width, and channel size, respectively. Since we are dealing with gray-scale images, the channel size has been set to 1. We have used Convolution2D Layer for our model with a set of 64 filters. The filter size has been chosen to be 33 with the stride value as 1. These filters act as feature identifiers. ReLU layer introduces non-linearity into the system. It is more computationally efficient than other nonlinear functions like *tanh* or *sigmoid*. Using this layer, the network trains much faster without sacrificing the accuracy. It also solves the problem of vanishing gradient. The ReLU layer applies the Equation 1 to change all the negative values to zero.

$$f(x) = max(0, x) \tag{1}$$

The max pooling layer generally accepts a kernel and a stride of the same length. It convolves the filter around the input volume and produces the maximum number in every sub-region as output. In our proposed model, we have used a 2×2 sized kernel and a stride of size 2. Flatten Layer transforms the feature matrix into a vector which can be fed into a fully connected layer. The Dropout layer randomly drops out a set of activations by setting their values to zero. Thus in a way, it forces the network to be redundant, i.e. even if some of the weights are dropped out, the network still provides the right classification for a specific example. It actually reduces the problem of over-fitting. Our model performed best by setting the Drop-out layer to 0.8. Finally, the Fully Connected Layer takes the input from the preceding layer and combines the local information learned by the previous layers in order to identify a bigger pattern. We have used three fully connected layers in our proposed model at various positions including the last layer. Last layer is the *Softmax Layer*. It looks at which high level features strongly correlate with a particular class and has particular weights so that when the products are computed between the weights and the previous layer, the correct probabilities for the different classes are obtained. Thus for our model, the last fully connected layer produces a vector as output. Since appropriate number of classes for Ol Chiki script is 30, dimension of this vector is 30.

The layers of CNN can be grouped into two basic parts with separate goals: feature extraction and classification. The con-

volution layers followed by the max-pooling and the activation functions act as a feature extractors while the classifier usually consists of fully connected layers, specifically the *Softmax* layer. The output extracted from the layer preceding the final fully connected layer provides feature map of the input. This feature-map can be easily plugged into another classification algorithm like SVM, KNN, etc.

B. The CNN-SVM Based Architecture

Since, SVMs are fast and reliable means of discriminating features, in our second method we have replaced the *softmax* layer with a SVM classifier. Fig. 3 presents the architecture of the second proposed method. The model obtained using this method has been named as the **CNN-SVM** Model. We have used SVMs with different specializations such as Linear Support Vector Classifier (SVC) having L1 loss function and L2 regularization, Linear kernel, Polynomial kernel, Sigmoid kernel, and Radial Basis Function (RBF) kernel; and trained them using the features extracted by the CNN. Although Linear SVC and SVC with Linear kernel both uses linear kernel, their approaches are different for multiclass problems, resulting into different performance. For multiclass reduction, Linear SVC uses the One-vs-All, method while SVC with Linear Kernel uses the One-vs-One method. Also, for multi-class classification problem SVC fits $N \times (N-1)/2$ models where N is the number of classes. Linear SVC, however, simply fits N models. Linear SVC uses the estimator *liblinear* which penalize the intercept for mis-classification but SVC uses the estimator *libsvm* which do not penalize the intercept. Moreover, *liblinear* converges faster than *libsvm* for linear problems, which makes Linear SVC solve the problem in very less time.

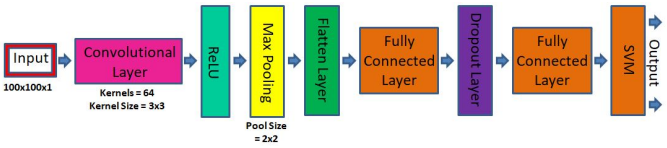
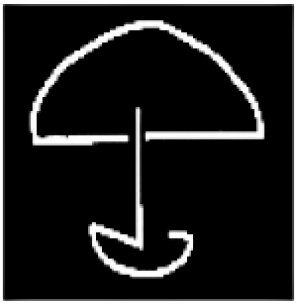


Fig. 3: The CNN-SVM Based Architecture.

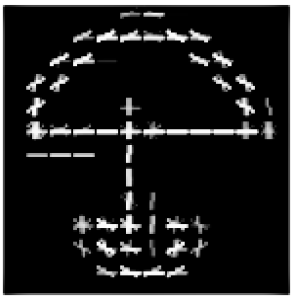
C. Handwritten Character Recognition using HOG feature descriptors

The histogram of oriented gradients (19), abbreviated as HOG, is a very frequently used feature descriptor. The key advantage of HOG descriptor over other descriptors is that it is invariant to geometric as well as photo-metric transformations except for object orientation as it operates on local cells. Such changes only appear for images having larger spatial regions. Fig. 4 and 5 show two different input images which have been fed into the HOG Feature Descriptor along with the corresponding outputs.

HOG has been used extensively to extract features from the images and has shown promising result on the tasks of handwritten character recognition (27; 28; 29). In our third method, we have used multiple cell-sized HOG feature descriptors to extract the features from our Training-Set samples

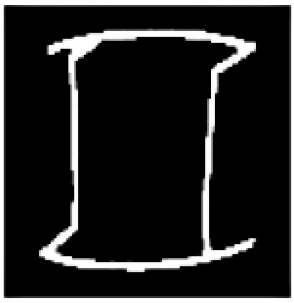


(a) Input Image of Letter “ud”

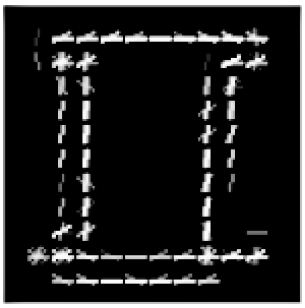


(b) Output Image of Letter “ud”

Fig. 4: First example for HOG Feature Descriptor.



(a) Input Image of Letter “ep”



(b) Output Image of Letter “ep”

Fig. 5: Second example for HOG Feature Descriptor.

having images of size 100×100. The extracted HOG features, returned as $1 \times N$ vector, have been used to train a SVM for the classification of the 30-class Ol Chiki datasets. For obtaining an efficient recognition rate, we have varied the cell sizes from 2×2 to 8×8 and compared the classification performance for each case. It is worth noting that large-scale spatial information can be captured by increasing the cell size. However, the suppression of changes in local illumination might get reduced as a result of averaging. Once the suitable

cell size has been determined, we have used the extracted HOG features to train multiple SVMs such as Linear SVC having L1 loss function and L2 regularization, SVC with Linear kernel, Polynomial, Sigmoid, and Radial Basis Function (RBF) kernel. Fig. 6 shows the architecture of the third proposed method called the **HOG-SVM** Model.

Fig. 6: The HOG-SVM Based Architecture.

D. Handwritten Character Recognition using combination of HOG and CNN

In this proposed method, we have combined the goodness of HOG and CNN together. Instead of training the CNN network directly with the whole image, we have trained it with the HOG feature descriptors. Fig. 7 shows the architecture of our fourth proposed method where we have used the HOG feature descriptors to extract the features from the samples in our Training-Set having size 100×100. The output of the HOG feature descriptor is fed as the input to a 1D CNN. It takes a two-dimensional input consists of pixels as well as color channels of an image to learn image features. The CNN architecture proposed by us consists of Convolutional layer followed by the layers: ReLU, Maxpooling, Flatten, Fully-connected or Dense, Dropout, and the Softmax. The model thus obtained has been referred as the **HOG-CNN** Model.

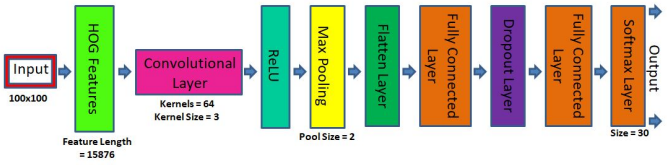


Fig. 7: The HOG-CNN Based Architecture.

III. Data Collection and Data Preparation

Since no standard dataset for this language is available yet, creation of a considerable large dataset was biggest challenge for us. The task has been divided into two phases - Data Collection and Preparation.

A. Data Collection

At first, several native as well as non-native writers have been asked to write few characters of the Ol Chiki script. Then two groups (i.e. Group 1 and Group 2) have been created in such a way that writers from both the categories were present in each group. From Group 1, we have collected 10,080 handwritten Ol Chiki characters, i.e. 336 samples for each character. From Group 2, we have collected 120 handwritten Ol Chiki characters, i.e. 4 samples for each character. Thus, in total 10,200 samples of handwritten Ol Chiki characters have been collected. Fig. 8 shows few samples from our collected data.

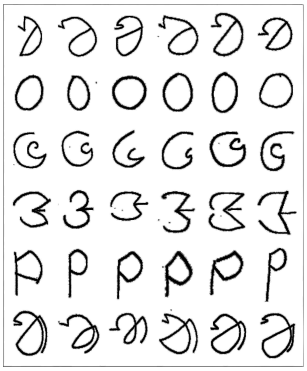


Fig. 8: Samples from collected Data.

B. Data Preparation

After data collection, all the 10,200 samples of Ol Chiki characters have been scanned, reshaped into a particular image size of 100×100 and converted into gray-scale. Two datasets have been created with these samples - Dataset 1 and Dataset 2 - each of them containing all the 10,200 images. A morphological closing operation has been applied on all samples belonging to Dataset 2 for reducing the noises present in the images. On the other hand, all the samples present in Dataset 1 have been kept as it is. Thus, the only difference in Dataset 1 and Dataset 2 is that - Dataset 1 contained noisy images while for Dataset 2 noises were reduced. For both of these datasets, all the 10,080 samples collected from the writers belonging to Group 1 have been partitioned into Training-Set and Test-Set 1 in $9:1$ ratio. Since, the samples in both Training-Set and Test-Set 1 have been written by the writers from Group 1, Test-Set 1 can be called as the *writer dependent* test set. Another test set, Test-Set 2 have been created for both the datasets consisting of 120 samples each, written by the writers from Group 2. Test-Set 2 can be called as the *writer independent* test set since, the handwriting samples of these writers have not been included in the Training-Set.

IV. Experimental Results

In this section, we have reported all the experimental results. All the models have been developed in *Google Colab* using *Keras* deep learning API. We have used sequential model along with *Categorical Cross Entropy* as a loss function and *Adam* as an optimizer. We got best results using epoch size of 100. We have compared performance of all the four models on Test-Set 1 (i.e. TS1) and Test-Set 2 (i.e. TS2) for both Dataset-1 and Dataset-2. Then suitable feature extractor and classifier to recognize handwritten Ol Chiki characters have been found out. It should be noted that while choosing the efficient HOCCR, we have given more priority to the accuracy obtained on TS2. Since the TS2 is *writer independent*, we believe it resembles the real-world environment. It may be noted that *Accuracy* has been used as a performance measurement metric here (30; 31).

TABLE I: Performance of the Models based on the CNN Architectures.

		Dataset 1		Dataset 2	
Model	Specialization	TS1	TS2	TS1	TS2
CNN	Not Applicable	84.62%	80.83%	85.12%	78.33%
CNN-SVM	Linear SVC	79.27%	70.00%	81.85%	66.67%
CNN-SVM	Linear Kernel	85.52%	74.17%	87.10%	71.67%
CNN-SVM	Polynomial Kernel	82.14%	75.00%	85.81%	71.67%
CNN-SVM	RBF Kernel	85.42%	75.00%	87.60%	71.67%
CNN-SVM	Sigmoid Kernel	85.22%	75.83%	87.60%	73.33%

TABLE II: Performance of HOG with Linear SVC on the basis of cell sizes.

		Dataset 1		Dataset 2	
Cell Size	Feature Length	TS1	TS2	TS1	TS2
2×2	4356	84.03%	70.83%	87.40%	68.33%
3×3	8100	87.60%	76.67%	89.58%	69.17%
4×4	11664	88.59%	78.33%	90.28%	73.33%
5×5	14400	90.77%	80.00%	91.37%	74.17%
6×6	15876	90.97%	82.50%	91.96%	77.50%
7×7	15876	91.67%	79.17%	92.16%	72.50%
8×8	14400	87.80%	80.00%	90.08%	72.50%

A. Experimental Results using CNN Based Architectures

Experimental results obtained using the handwritten Ol Chiki character recognition systems based on **CNN** and the **CNN-SVM** Models have been presented here. These models have been described in the Sub-section II-A. The results have been presented in Table I. Our proposed **CNN-SVM** Model has achieved the best result using the Sigmoid Kernel. The CNN-SVM Model using Linear SVC has performed most poorly. Although for Test-Set 1 of Dataset 1, the CNN-SVM Model with Linear Kernel has given the best result, but considering the overall performance of all the CNN-SVM models on all test cases, the **CNN-SVM** Model with Sigmoid Kernel has been chosen as the best.

B. Experimental Results using HOG Based Architectures

This section describes the experimental results obtained using **HOG-SVM** Model to recognise handwritten Ol Chiki characters. For obtaining the best result for the **HOG-SVM** Model, we have varied the cell sizes of the HOG feature descriptor from 2×2 to 8×8 described in Sub-section II-C. Since for our **CNN-SVM** Model the Linear SVC gave the worst performance, we have first checked the performance of the HOG features with Linear SVC. Table II presents the behavior of HOG-Linear SVC Model when cell size has been varied. The best performance has been achieved with cell size 7×7 for Test-Set 1 (i.e. TS1) and cell size 6×6 for Test-Set 2 (i.e. TS2). Since we have given more priority to TS2 than TS1, we have selected the cell size 6×6 for the HOG feature descriptor for both the datasets. We have extracted the HOG features with cell size 6×6 and trained it with different specializations of SVMs whose results have been shown in Table III. Unlike the **CNN-SVM** Model where the Sigmoid kernel gave the best results, in case of **HOG-SVM** Model the worst result has been obtained on all the test cases using the SVM with Sigmoid kernel. However, the best result has been obtained using the Polynomial kernel.

TABLE III: Performance of the HOG-SVM Models.

	Dataset 1		Dataset 2	
SVM Specialization	TS1	TS2	TS1	TS2
Linear SVC	90.97%	82.50%	91.96%	77.50%
Linear Kernel	91.17%	80.83%	92.66%	75.83%
Polynomial Kernel	93.85%	84.17%	94.35%	81.67%
RBF Kernel	92.66%	83.33%	92.76%	81.67%
Sigmoid Kernel	79.66%	72.50%	80.56%	69.17%

TABLE IV: Accuracy Table

	Dataset 1		**Dataset 2**	
Models	**TS1**	**TS2**	**TS1**	**TS2**
CNN	84.62%	80.83%	85.12%	78.33%
CNN-SVM	85.22%	75.83%	87.60%	73.33%
HOG-SVM	93.85%	84.17%	94.35%	81.67%
HOG-CNN	92.26%	92.50%	93.55%	89.17%

C. Discussion

Table IV presents the results obtained by all the four proposed models on Test-set 1 and Test-set 2 for both Dataset 1 and Dataset 2. Based on the overall obtained results the following observations have been made.

- For all the models, the accuracies obtained using writer-dependent test cases are better for the noiseless dataset while the accuracies of the writer-independent test cases are better for the noisy dataset. Thus, it establishes the fact that all the models are doing better generalization and fault tolerance when they have been trained with noisy handwritten dataset (32). The little distortions in the pixels of the sample images are adding more variations and in a way acting as a regularization technique and hence, preventing the models from overfitting the handwriting of the writers (33). Bishop (34) showed that presence of noise in training data may introduce a regularization effect (i.e. Tikhonov Regularization) in the neural network model. It actually increases robustness of the model. Given the noise variance parameter is small, the noises have a similar effect on the loss function. The random noises present in the samples force the network to memorize the training sample less because they are changing all the time. Finally, it results into a very robust neural network with low generalization error (35). These noises also act like new samples which have been created from the known samples. Thus, introduction of noise to input samples may also be treated as a form of data augmentation (36). This, in turn, carries out a smoothing effect in the structure of the input space. This smoothing technique helps to learn the mapping function more easily. So, this ensures better and faster learning.
- The results were improved when the CNN had been trained with the HOG features (i.e. HOG-CNN) instead of whole images. Thus, it shows that by training the CNN network with only relevant information of the image, instead of the entire image, has improved the performance significantly.
- Out of all the models, the **HOG-SVM** Model gave the best result on Test-Set 1, i.e. the writer-dependent test case, for both Dataset 1 and Dataset 2 which also confirms the fact that SVM performed better with HOG

features than with the CNN features. Moreover, the best result on Test-Set 1, i.e. 94.35%, has been obtained with this model on Dataset 2, i.e. the noiseless dataset. However, the accuracies obtained on Test-Set 2, i.e. the writer-independent test case, for both the datasets are very low in comparison to any HOG-CNN based models.

- For writer-independent test cases, i.e. Test-Set 2, of both the datasets, the best performance has been returned by the **HOG-CNN** Model. Moreover, the best result on Test-Set 2, i.e. 92.50%, has been obtained on the dataset having noisy images, i.e. Dataset 1 by the **HOG-CNN** Model. This is due to the primary property of the max-pooling layer which builds up tolerance to severe distortions. Pooling allows the features to move around, relative to each other. Since this pooling takes place at multiple levels, the low-level features are allowed to shift by small shifts while the high-level features can shift dramatically without affecting the confidence score much. This makes the CNNs very robust (37).
- Considering all the results and observations we can conclude that the performance of the **HOG-CNN** Model is much better than any other models as the learning made by this model is able to classify the Ol Chiki characters irrespective of any handwriting. For the **HOG-CNN** Model, we have obtained the percentage of confidence with which the class prediction is being made. For this, we have randomly selected one sample from each class from Test-Set 2 of Dataset 1. The complete output in form of a video can be found here [1]. For 25 classes, the confidence varied between 96%-100%. For the remaining classes, the confidence varied between 75% to 90%. The lower confidence in few of the characters have been occurred because of the presence of another character which is similar to it. For instance, the character "lu" is very similar to the character "aak" and hence the class prediction for "lu" is done with 88% confidence for character "lu" and 12% for confidence for character "aak". Similarly, the class prediction for "ov" is done with 76% confidence for character "ov" and 24% for confidence for character "aaw", which are again very similar to each other. These characters are presented in the Fig. 9.

Fig. 9: Character "lu", "aak", "ov", and "aaw"

V. Conclusion

A deep-learning based framework for recognizing handwritten Ol Chiki characters has been proposed here. Two handwritten Ol Chiki character datasets - Dataset 1 and Dataset 2 have been created by us, consisting of noisy and noiseless sample images respectively. We have proposed four different prediction models. These models have been trained and tested using both these data sets. Moreover, we have run all the models with various parameters to obtain the best results on each of them. The results obtained by us showed that training the model using the noisy dataset resulted into a more generalized performance than that using the noiseless dataset. For *writer-dependent* test cases, the best result has been obtained by the **HOG-SVM** model. However, its performance for the *writer-independent* test cases for both the datasets were comparatively poor, leading to the finding that performance of the **HOG-SVM** model is highly dependent on the handwriting of the writers the model has been trained with. For writer-independent test case, the best result has been obtained by the **HOG-CNN** Model. Also, similar performance have been achieved for the writer-dependent test case for both the datasets using the **HOG-CNN** Model. Moreover, our results indicate that the CNN performed better when trained with the HOG features instead of the whole image. Considering the fact that the Ol Chiki character recognizer should be robust, i.e. it should be able to recognize characters of any handwriting, the **HOG-CNN** Model has been chosen as the best one for classifying all the Ol Chiki Characters. As our present work has been limited to *offline* classification of the handwritten Ol Chiki characters, in the future this research can be extended for recognition of *online* Ol Chiki handwritten characters and the development of an Ol Chiki language model. Other recently proposed deep learning techniques (e.g. few-shot learning, liquid neural network etc.) can also be studied to improve the recognition rate.

Acknowledgements: Authors gratefully acknowledge financial support from DST-PURSE, Visva-Bharati.

[1] https://youtu.be/sYjGRvE-3hw

References

[1] A. Yuan, G. Bai, L. Jiao, and Y. Liu, "Offline handwritten english character recognition based on convolutional neural network," in *2012 10th IAPR International Workshop on Document Analysis Systems*. IEEE, 2012, pp. 125–129.

[2] P. Jain, K. Taneja, and H. Taneja, "Which ocr toolset is good and why: A comparative study," *Kuwait Journal of Science*, vol. 48, no. 2, 2021.

[3] N.-T. Ly, C.-T. Nguyen, K.-C. Nguyen, and M. Nakagawa, "Deep convolutional recurrent network for segmentation-free offline handwritten japanese text recognition," in *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, vol. 7. IEEE, 2017, pp. 5–9.

[4] T. S. El-Sheikh and S. El-Taweel, "Real-time arabic handwritten character recognition," *Pattern recognition*, vol. 23, no. 12, pp. 1323–1332, 1990.

[5] M. Rashad and N. A. Semary, "Isolated printed arabic character recognition using knn and random forest tree classifiers," in *International Conference on Advanced Machine Learning Technologies and Applications*. Springer, 2014, pp. 11–17.

[6] T. Wang, Z. Xie, Z. Li, L. Jin, and X. Chen, "Radical aggregation network for few-shot offline handwritten chinese character recognition," *Pattern Recognition Letters*, vol. 125, pp. 821–827, 2019.

[7] J. Gan, W. Wang, and K. Lu, "Compressing the cnn architecture for in-air handwritten chinese character recognition," *Pattern Recognition Letters*, vol. 129, pp. 190–197, 2020.

[8] A. Gupta, R. Sarkhel, N. Das, and M. Kundu, "Multiobjective optimization for recognition of isolated handwritten indic scripts," *Pattern Recognition Letters*, vol. 128, pp. 318–325, 2019.

[9] K. Dutta, M. Bal, A. Basak, S. Ghosh, N. Das, M. Kundu, and M. Nasipuri, "Multi scale mirror connection based encoder decoder network for text localization," *Pattern Recognition Letters*, 2020.

[10] A. K. Pujari, C. D. Naidu, M. S. Rao, and B. Jinaga, "An intelligent character recognizer for telugu scripts using multiresolution analysis and associative memory," *Image and Vision Computing*, vol. 22, no. 14, pp. 1221–1227, 2004.

[11] S. Inunganbi and P. Choudhary, "Recognition of handwritten meitei mayek script based on texture feature," *Int. J. Nat. Lang. Comput.(IJNLC)*, vol. 7, no. 5, pp. 99–108, 2018.

[12] P. M. Kamble and R. S. Hegadi, "Handwritten marathi character recognition using r-hog feature," *Procedia Computer Science*, vol. 45, no. 1, pp. 266–274, 2015.

[13] O. Surinta, M. F. Karaaba, T. K. Mishra, L. R. Schomaker, and M. A. Wiering, "Recognizing handwritten characters with local descriptors and bags of visual words," in *International Conference on Engineering Applications of Neural Networks*. Springer, 2015, pp. 255–264.

[14] B. Dhandra, M. Hangarge, and G. Mukarambi, "Spatial features for handwritten kannada and english character recognition," *IJCA, Special Issue on RTIPPR (3)*, pp. 146–151, 2010.

[15] U. Pal, N. Sharma, T. Wakabayashi, and F. Kimura, "Off-line handwritten character recognition of devnagari script," in *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*, vol. 1. IEEE, 2007, pp. 496–500.

[16] S. S. Mohapatra, "Formation of ol chiki script and process of its transmission," *Santhal Worldview*, p. 74, 2001.

[17] S. Daw and A. C. Mondal, "A hybrid recognition system of handwritten olchiki character and digit," *International Journal of Computer Sciences and Engineering*, vol. 7, no. 1, 2019.

[18] J. Basua, T. R. Hrangkhawlb, T. K. Basuc, and S. Majumderd, "Identification of two tribal languages of india: An experimental study," in *Artificial Intelligence and Speech Technology: Proceedings of the 2nd International Conference on Artificial Intelligence and Speech Technology,(AIST2020), 19-20 November, 2020, Delhi, India*. CRC Press, 2021, p. 221.

[19] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, vol. 1. IEEE, 2005, pp. 886–893.

[20] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proceedings of the seventh IEEE international conference on computer vision*, vol. 2. Ieee, 1999, pp. 1150–1157.

[21] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *European conference on computer vision*. Springer, 2006, pp. 404–417.

[22] T. Ojala, M. Pietikainen, and D. Harwood, "Performance evaluation of texture measures with classification based on kullback discrimination of distributions," in *Proceedings of 12th International Conference on Pattern Recognition*, vol. 1. IEEE, 1994, pp. 582–585.

[23] T. Ojala, M. Pietikäinen, and D. Harwood, "A comparative study of texture measures with classification based on featured distributions," *Pattern recognition*, vol. 29, no. 1, pp. 51–59, 1996.

[24] A. Ganesh, A. R. Jadhav, and K. C. Pragadeesh, "Deep learning approach for recognition of handwritten kannada numerals," in *International Conference on Soft Computing and Pattern Recognition*. Springer, 2016, pp. 294–303.

[25] S. Roy, N. Das, M. Kundu, and M. Nasipuri, "Handwritten isolated bangla compound character recognition: A new benchmark using a novel deep learning approach," *Pattern Recognition Letters*, vol. 90, pp. 15–21, 2017.

[26] P. P. Nair, A. James, and C. Saravanan, "Malayalam handwritten character recognition using convolutional neural network," in *2017 International conference on inventive communication and computational technologies (ICICCT)*. IEEE, 2017, pp. 278–281.

[27] N. A. Jebril, H. R. Al-Zoubi, and Q. A. Al-Haija, "Recognition of handwritten arabic characters using histograms of oriented gradient (hog)," *Pattern Recognition and Image Analysis*, vol. 28, no. 2, pp. 321–345, 2018.

[28] K. Nongmeikapam, K. Wahengbam, O. N. Meetei, and T. Tuithung, "Handwritten manipuri meetei-mayek classification using convolutional neural network," *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)*, vol. 18, no. 4, pp. 1–23, 2019.

[29] M. R. Phangtriastu, J. Harefa, and D. F. Tanoto, "Comparison between neural network and support vector machine in optical character recognition," *Procedia computer science*, vol. 116, pp. 351–357, 2017.

[30] A. Sharma and P. K. Mishra, "State-of-the-art in performance metrics and future directions for data science algorithms," *Journal of Scientific Research*, vol. 64, no. 2,

2020.

[31] ——, "Performance analysis of machine learning based optimized feature selection approaches for breast cancer diagnosis," *International Journal of Information Technology*, pp. 1–12, 2021.

[32] R. Reed and R. J. MarksII, *Neural smithing: supervised learning in feedforward artificial neural networks*. Mit Press, 1999.

[33] C. M. Bishop *et al.*, *Neural networks for pattern recognition*. Oxford university press, 1995.

[34] C. M. Bishop, "Training with noise is equivalent to tikhonov regularization," *Neural computation*, vol. 7, no. 1, pp. 108–116, 1995.

[35] G. An, "The effects of adding noise during backpropagation training on a generalization performance," *Neural computation*, vol. 8, no. 3, pp. 643–674, 1996.

[36] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[37] S. Sharma and R. Mehra, "Implications of pooling strategies in convolutional neural networks: A deep insight," *Foundations of Computing and Decision Sciences*, vol. 44, no. 3, pp. 303–330, 2019.