SVM with Inverse Fringe as feature for Improving Accuracy of Telugu OCR Systems

Amit Patel*[1], Burra Sukumar[2], Chakravarthy Bhagvati[2]

[1] Rajiv Gandhi University of Knowledge Technologies, IIIT Nuzvid
Nuzvid, Krishna, 521201, Andhra Pradesh, India
[2] School of Computer and Information Sciences, University of Hyderabad,
Hyderabad, 500046, Telangana, India
amtptl93@gmail.com, chakcs@uohyd.ernet.in

Abstract. Designing an OCR system with high accuracy is quite a tough task as the system performance gets affected by its component modules. The accuracy and quality of the OCR system depends on impact of each module. The overall system performance changes if there is an improvement in a module. In our work at present we have developed an OCR system for Telugu (Drishti System).We proposed in our paper SVM algorithm with inverse fringe as feature for Telugu OCR. The idea is to improve the performance of system by increasing recognition accuracy of the developed system. Support Vector Machines (SVM) was shown by several researchers to deliver high performance on Indic OCRs. SVMs have been applied to Telugu OCR and are tested with different features. In our experiments, we used fringe distance and its complementary version, the inverse fringe as a feature to the SVM. These two features have been used to develop the working model of Telugu OCR with an accuracy approaching 90%. It is shown that the performance is good over more than 300 classes. With inverse fringe as feature, the system with 325 classes is trained with 15543 labeled Telugu characters and tested over 75335 unlabeled Telugu characters; the accuracy of the system is found 99.50%. The SVM based classifier is tested on our scanned image document corpus of more than 4500 pages and about 5,000,000 symbols. Evaluation of end-end system performance is done in our experiments. From the results it has been depicted that SVM classifier is giving an improvement of approximately 1.24% over the developed Telugu OCR (Drishti System).

Keywords: Fringe map, Telugu script, Telugu OCR, System performance, Indian scripts.

1 Introduction

Designing an OCR system with high accuracy is a challenging task as the performance of the system will be affected by its component modules like pre-processing (binarization, noise removal, skew detection and correction etc.), line, word and character segmentation, feature extraction, classification, post processing etc. The overall quality of the system depends on the correctness of each module. If

Errors occur in any of the modules they may propagate through the successive modules and may turmoil the overall system performance. In order to get a more robust OCR system [15] there is an urgent need to handle these errors i.e. we need to handle the errors in any of the modules. By considering an OCR system, being developed for Telugu as a part of funded project by Govt. of India, in our present. From different kinds of books a collection of 4500 scanned documents (about 5,000,000 symbols) has been created along with their ground truth. As most of these documents have been taken from old books so they have got degraded. Other attempts to Telugu OCR [13] [14] in the literature have neither have reported an end-end performance evaluation nor they have shown results on such large data [15]. We observed from the experiments that the performance of the system is less due to recognition module and even due to improper segmentation. Here we tries to improve the recognition module by replacing it SVM instead of KNN (K Nearest Neighbor) classifier which is based on the fringe map feature and uses distance measure [4]. We trained SVM with inverse fringe distance map as feature. Finally as a result we obtained a major improvement in the overall accuracy of the end-to-end system.

Telugu Optical Character Recognizing system (Telugu OCR) [4] is to recognize Telugu character components by labeling them with appropriate labels. Telugu script contains large number of component classes set, which is including consonants, vowels, and combinations of consonant, vowel modifier, combinations of consonants and few other symbols. If we can separate the vowel and consonant modifier part by segmenting, all these components may results in lesser number of classes. As there exists no mechanism to do this perfectly, components to be dealt by the classifier are resulting in large number of classes. We have adopted *connected component strategy* to reduce the possible number of distinct components to be recognized by Telugu OCR, this resulted in approximately 400 classes.

An OCR system has mainly consists of two phases in it. First is feature extractor and second is classifier. Purpose of feature extractor is to develop features for recognizing classes using available information. Classifier functionality is to classify the input samples and assign the appropriate labels. In literature there are many classifiers such as Bayesian Networks, Decision Tree Classifiers, Support Vector Machines (SVM), and Nearest Neighbor Classifiers. Among these all no classifier available which is guaranteed to perform well, we choose SVM for our experiments and tried with different features such as fringe and inverse fringe as feature for the SVM and found that performance is good over even if we increased the number of classes more than 300. The performance of inverse fringe distance map (IFDM) as feature is better than fringe distance map (FDM) as feature.

The paper is structured as follows. Apart from introduction, there are five sections in this paper. Section 2 highlights the overview of the Telugu OCR system. Section 3 is about SVM and SVM library adopted for experiments. In section 4 we discussed about the feature extraction process and recognition accuracy with that feature. In section 5 we outlined about the experimental results and compared with the existing system results. We concluded with section 6.

2 Telugu OCR system

An OCR system is one which consists of several modules like pre-processing (binarization, noise removal, skew detection etc.), segmentation of line, word and character, feature extraction, classification, post processing. In the survey paper [8] author have discussed about different methods for binarization and found adaptive method shows best performance on our corpus. Based on projection profiles segmentation of line, word and character are performed [9]. From the experiments performed on our corpus it is found that classifiers such as Neural Networks, SVM [12] do not perform well because corpus contains large number of broken characters/classes with different features [4], [10], [11]. Fringe distance map as a feature [4] for K-NN classifier is used as it is found that K-NN shown the better performance than other classifier on our corpus.

3 Support Vector Machine

In the field of pattern recognition Support Vector Machine (SVM) [6] is a statistical method which has shown great success in many practical approach, such as text classification, face recognition, handwritten digit recognition etc. By projecting data into the feature space and then finding the optimal separate hydroplane [1] SVMs can transform a non-linear separable problem into linear separable problem with the help of different kernel functions. Initially this kind of method was used to solve two class classification problems. Few strategies were introduced later to extend this technique to solve multiple class classification problems. In [2] author has discussed a very good and comprehensive theory about SVM.

3.1 SVM Library Adopted in the Experiments

OpenCV SVM (LIBSVM) [7] has been used to build the SVM models in our experiments. OpenCV SVM is quite efficient, simple and easy to use software for the classification using SVM. For the experiments, The SVM classifier was trained to make the prediction using probabilities value assigned to each class. By looking the values of probabilities assigned to each class we can easily apply post processing for decision making whether to reject or accept the character. Kernel function used is Radial Basis Function (RBF). SVM type as NU_SVC function is used for training and prediction. 10-fold cross validation is done while training.

4 Feature Extraction for the SVM Model

To achieve a good performance from an SVM classifier, we used fringe distance map and inverse fringe distance map as a feature. There are 1024 variables in one feature vector, computed by fringe. These features are described below.

4.1 Fringe Distance Map

We used *fringe distance map* [3] as feature for the SVM, to calculate FDM, we took an image of a character of dimension of 32 x 32, so total number of pixel present in image is 1024 and we assumed every pixel value as a variable, so one feature vector contain 1024 variables. This is also assumed that the glyphs are in black color and background of glyph is of white color. Use L1 metric or city-block distance to compute the distances [4] (Fig. 1). Efficient computation of the fringe distances can be done if at each pixel position of the template, the distances of the nearest black pixel are pre-computed and stored [4].

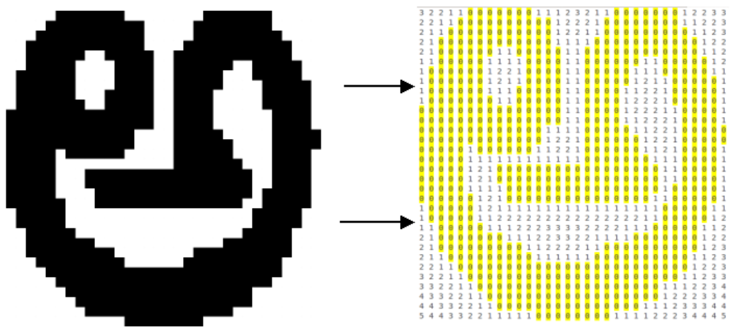


Fig. 1 Example of Fringe Distance Map: A character of Telugu is taken which is represented in over the white background; distances of black pixel are calculated by L1 metric or city block distance.

We used this calculated value as a feature vector.

```
"3,2,2,1,1,0,0,0,0,0,0,1,1,1,2,3,2,1,1,0,0,0,0,0,0,1,2,2,3,3,2,2,1,1,0,0,0,0,0,0,0,0,0,1,2,2,2,1,0,0,0,0,0,0,0,1,1,2,2
,3,2,1,1,0,0,0,0,0,0,0,0,0,0,1,2,2,1,1,0,0,0,0,0,0,0,0,0,1,1,2,3,2,1,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,1
,2,2,2,1,0,0,0,0,0,0,1,1,0,0,0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,1,1,2,1,1,0,0,0,0,0,0,1,1,1,1,0,0,0,0,1,1,0,0,0,0,0,0,1,1,0,0,0,0
,0,1,2,1,0,0,0,0,0,0,1,2,2,1,0,0,0,0,1,1,0,0,0,0,0,1,1,1,0,0,0,0,0,1,1,1,0,0,0,0,0,0,1,2,1,1,0,0,0,0,1,1,0,0,0,0,0,1,2,1,1,0,0
,0,0,0,1,1,0,0,0,0,0,0,1,1,1,0,0,0,0,0,1,1,0,0,0,0,1,1,2,2,1,0,0,0,0,0,1,1,0,0,0,0,0,0,0,1,1,0,0,0,0,0,1,1,0,0,0,0,1,2,2,2,1,0
,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,2,2,2,1,1,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,1,2,2,2
,1,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,0,0,0,0,0,1,1,2,2,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,2,2,1,0,0,0,0,0,0,1,2
,2,1,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,1,1,2,2,1,0,0,0,0,0,0,1,1,2,1,0,0,0,0,1,0,0,0,0,0,1,1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0
,1,1,1,0,0,0,0,1,0,0,0,0,0,1,2,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,0,0,0,1,2,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
,0,0,1,1,0,0,0,0,1,0,0,0,0,0,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,1,2,1,0,0,0,0,0,0,0,0,0,0,0,0,0
,0,0,1,1,0,0,0,0,0,1,1,0,0,0,0,0,1,2,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,1,1,1,0,0,0,0,0,1,1,2,2,2,2,2,2,2,2,2,2,2,2,2
,2,2,1,1,0,0,0,0,0,1,2,1,1,0,0,0,0,0,1,1,1,2,2,2,3,3,3,3,2,2,2,2,1,1,1,0,0,0,0,0,1,1,2,2,1,0,0,0,0,0,0,0,1,1,1,2,2,3,3,2,2,1,1
,1,1,0,0,0,0,0,0,0,1,2,2,2,1,0,0,0,0,0,0,0,0,0,1,1,2,2,2,2,1,1,0,0,0,0,0,0,0,0,0,0,0,1,2,3,2,1,1,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,0
,0,0,0,0,0,0,0,0,0,0,1,1,2,3,2,2,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,2,2,3,3,2,2,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0,0,0,1,1,2,3,3,3,3,2,2,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,2,2,3,4,4,3,3,2,2,1,1,0,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0,1,2,2,2,3,3,4,4,4,3,3,2,2,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,2,3,3,3,4,4,5,4,4,3,3,2,2,1,1,1,1,1,0,0,0,0
,0,0,0,0,1,1,1,1,2,2,2,3,4,4,4,5"
```

4.2 Inverse Fringe Distance Map

We used *inverse fringe distance map* [3] as feature for the SVM, to calculate IFDM, we took an image of a character of dimension of 32 x 32, so total number of pixel present in image is 1024 and we assumed every pixel value as a variable, so one feature vector contain 1024 variables. This is also assumed that the glyphs are in white color and background of glyph is of black color. Use L1 metric or city-block distance to compute the distances [4] (Fig. 2). Efficient computation of the fringe distances can be done if at each pixel position of the template, the distances of the nearest black pixel are pre-computed and stored [4].

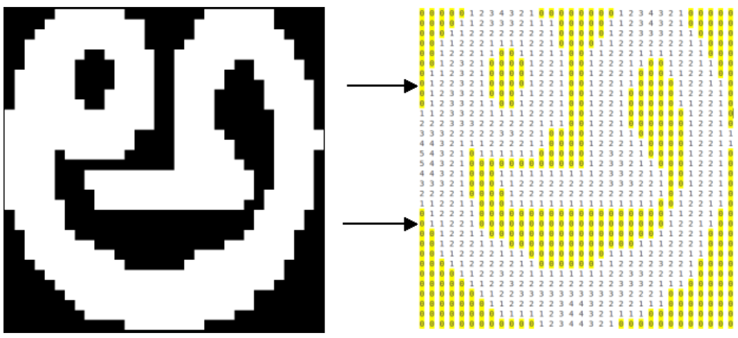


Fig. 2 Example of Inverse Fringe Distance Map: A character of Telugu is taken which is represented in over the black background; distances of black pixel are calculated by L1 metric or city block distance.

We used this calculated value as a feature vector.

"0,0,0,0,0,1,2,3,4,3,2,1,0,0,0,0,0,0,0,0,1,2,3,4,3,2,1,0,0,0,0,0,0,0,0,0,1,1,2,3,3,3,2,1,1,1,0,0,0,0,0,1,1,2,3,4,3,2,1,0,0,0,0
,0,0,0,0,1,1,2,2,2,2,2,2,2,2,1,0,0,0,0,0,1,2,2,3,3,3,2,1,1,0,0,0,0,0,0,1,1,2,2,2,1,1,1,1,2,2,1,0,0,0,0,1,1,2,2,2,2,2,2,2,2,1,1,0
,0,0,0,0,1,2,2,2,1,1,0,0,1,1,2,1,1,0,0,1,1,2,2,2,1,1,1,1,2,2,1,0,0,0,0,0,1,2,3,2,1,0,0,0,0,1,2,2,1,0,0,1,2,2,2,1,1,0,0,1,2,2,1
,1,0,0,0,1,1,2,3,2,1,0,0,0,0,1,2,2,1,0,0,1,2,2,2,1,0,0,0,1,1,2,2,1,0,0,0,1,2,2,3,2,1,0,0,0,0,1,2,2,1,0,0,1,2,2,1,1,0,0,0,0,1,2
,2,1,1,0,0,1,2,3,3,2,1,0,0,0,1,1,2,2,1,0,0,1,2,2,1,0,0,0,0,0,1,2,2,2,1,0,0,1,2,3,3,2,1,1,0,0,1,2,2,2,1,0,0,1,2,2,1,0,0,0,0,0,1
,1,2,2,1,0,1,1,2,3,3,2,2,1,1,1,1,2,2,2,1,0,0,1,2,2,1,0,0,0,0,0,0,1,2,2,1,0,2,2,2,3,3,3,2,2,2,2,2,2,2,1,1,1,0,0,1,2,2,1,0,0,0,0,0
,0,1,2,2,1,0,3,3,3,2,2,2,2,2,3,3,2,2,1,0,0,0,0,1,2,2,1,1,0,0,0,0,0,1,2,2,1,1,4,4,3,2,1,1,1,2,2,2,2,1,1,0,0,0,0,1,2,2,2,1,1,0,0
,0,0,1,2,2,1,1,5,4,3,2,1,0,1,1,1,1,1,1,0,0,0,0,0,1,2,3,2,2,1,0,0,0,0,1,2,2,1,0,5,4,3,2,1,0,0,0,0,0,0,0,0,0,0,0,0,1,2,3,3,2,1,1
,0,0,0,1,2,2,1,0,4,4,3,2,1,0,0,0,1,1,1,1,1,1,1,1,1,1,2,3,3,2,2,1,1,0,0,1,2,2,1,0,3,3,3,2,1,0,0,0,1,1,2,2,2,2,2,2,2,2,2,2,3,3,3,2
,2,1,0,0,1,2,2,1,0,2,2,2,2,1,0,0,0,0,1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1,1,0,1,1,2,2,1,0,1,1,2,2,1,1,0,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1
,1,1,0,0,1,2,2,1,1,0,0,1,2,2,2,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,2,2,1,0,0,0,1,1,2,2,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,1,2,2,1,1,0,0,0,0,1,2,2,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,2,2,1,0,0,0,0,0,1,2,2,2,1,1,1,0,0,0,0,0,0,0,0,0,0,0
,0,0,1,1,1,2,2,2,1,0,0,0,0,0,1,1,2,2,2,2,1,1,1,0,0,0,0,0,0,0,0,1,1,1,1,2,2,2,2,1,1,0,0,0,0,0,0,1,1,2,2,2,2,2,1,1,0,0,0,0,0,0,1
,1,2,2,2,2,3,2,2,1,0,0,0,0,0,0,0,0,1,1,2,2,3,2,2,1,1,1,1,1,1,1,1,2,2,3,3,2,2,2,1,1,0,0,0,0,0,0,0,0,0,1,1,2,2,3,2,2,2,2,2,2,2,2
,2,2,3,3,3,2,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,1,1,2,2,3,3,3,3,3,3,3,3,3,3,3,3,2,2,2,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,2,2,2,2,2,3,4,4
,3,2,2,2,2,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,2,3,4,4,3,2,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,2,3,4
,4,3,2,1,0,0,0,0,0,0,0,0,0,0,0,0"

4.3 Experiments with Fringe as feature

We started our experiment with 20 classes, In this process first we took an image of character and calculated the FDM and stored all the values of pixel as a variable in a feature vector, so one feature vector consists of 1024 variables. For training we took 445 sample character from different 20 classes and 1132 character for test data and found the average accuracy of the system is 98.93 percent. The next experiment is conducted with 45 classes, we trained SVM with 2436 characters and tested 19582 characters and found the recognition accuracy of the SVM is 99.45 percent. The results are shown in Table 1.

4.4 Experiments with Inverse Fringe as feature

We started the experiment with the 20 classes, trained the SVM with small data-set and tested the recognition accuracy of the SVM. In this process we did the same as we did in experiment 4.3. For training we took 445 character samples from different 20 classes and 1132 characters for test data, and found the average accuracy of the system is 100.00 percent which is better than the SVM in which we used fringe distance map as a feature. The next experiment is conducted with the 45 different classes. We trained the SVM with a dataset of 2436 character sample and tested on large dataset consisting of 19582 and found the recognition accuracy of the SVM is 99.89 percent. The results are shown in Table 1.

4.5 Experimental Results

From the above two experiments we found that using inverse fringe distance map as a feature gives good result as compared to fringe distance map as a feature. Table 1 shows the comparison between the two SVMs and their recognition accuracy.

Table 1. Comparison of the Results of SVMs with FDM and IFDM as features

No. of Classes	Classifier's Feature	Train Data Set Size	Test Data Set Size	Correct Recognition	Incorrect Recognition	Accuracy
20	FDM	445	1132	1120	12	98.93
20	IFDM	445	1132	1132	0	100.00
45	FDM	2436	19582	19475	107	99.45
45	IFDM	2436	19582	19562	20	99.89

Table 2 shows the results of all the experiments done by using inverse fringe distance as feature vector. First we started the experiment with less number of classes and gradually we increased the number of classes, as we increased the number of classes the recognition accuracy is decreased by small value but the SVM shows good performance in handling the large number of classes with IFDM as feature.

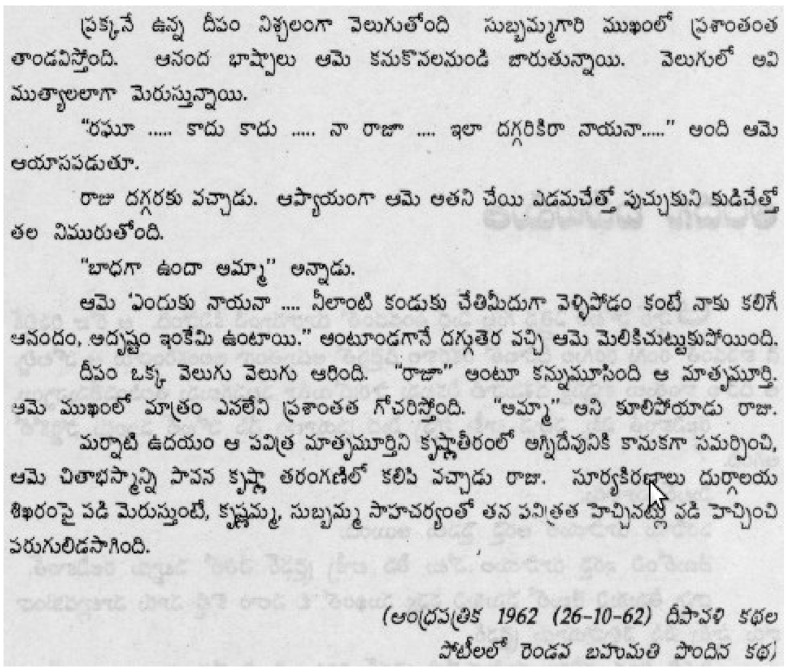
Table 2. Results of SVMs using IFDM as feature

No. of Classes	Training Data Set Size	Test Data Set Size	Correct Recognition	Incorrect Recognition	Accuracy
20	445	1132	1132	0	100.00
45	2436	19582	19562	20	99.89
75	3862	26246	26208	38	99.85
100	5239	37534	37438	96	99.74
125	7796	44480	44367	113	99.74
175	10900	62306	62059	247	99.60
208	12287	67323	66990	333	99.50
325	15543	75335	74960	375	99.50

5 Experimental Results and Analysis

After integrating the SMV into our developed OCR system [15], we have compared both the previous OCR system and improved OCR system performances over 27 novels on around 4525 pages in the corpus. Computation of error rate for a given page can be done by using a traditional string matching algorithm, Levenshtein edit distance (LED) [5]. We need to convert one string to other for a given two strings by using LED as it gives minimum number of insertion, deletion and substitution. For a given page we are taking all the UNICODES of ground truth text into one string and all UNICODE of OCR output text as another string. The computation of LED can be done between these strings for a given page. The overall error rate for a given page can be obtained by the ratio of LED to the number of UNICODES in the ground truth text. The errors in the text output are ultimately reflected as the errors of any module of OCR system.

The example shown in Fig. 3 and Fig. 4 is input and output of the previous OCR system and improved OCR system. In the previous OCR system there are 428 glyphs can be recognized by KNN as they were present in template, but in improved OCR system only 325 glyphs can be recognized by using SVM as SVM is trained with only 325 different glyphs. For the given input page, recognition accuracy is increased by 6.88 %. Figure 4 shows the results which incorrectly classified (red box) by previous system and correctly classified (green box) by improved system.



ప్రక్కనే ఉన్న దీపం నిశ్చలంగా వెలుగుతోంది సుబ్బమ్మగారి ముఖంలో ప్రశాంతంత
తాండవిస్తోంది. ఆనంద భాష్పాలు ఆమె కనుకొనలనుండి జారుతున్నాయి. వెలుగులో అవి
ముత్యాలలాగా మెరుస్తున్నాయి.
"రఘూ ..... కాదు కాదు ..... నా రాజా .... ఇలా దగ్గరికిరా నాయనా....." అంది ఆమె
ఆయాసపడుతూ.
రాజు దగ్గరకు వచ్చాడు. ఆప్యాయంగా ఆమె అతని చేయి ఎడమచేత్తో పుచ్చుకుని కుడిచేత్తో
తల నిమురుతోంది.
"బాధగా ఉందా అమ్మా" అన్నాడు.
ఆమె 'ఎందుకు నాయనా .... నీలాంటి కండుకు చేతిమీదుగా వెళ్ళిపోడం కంటే నాకు కలిగే
ఆనందం, అదృష్టం ఇంకేమి ఉంటాయి." అంటూండగానే దగ్గుతెర వచ్చి ఆమె మెలికిచుట్టుకుపోయింది.
దీపం ఒక్క వెలుగు వెలుగు ఆరింది. "రాజా" అంటూ కన్నుమూసింది ఆ మాతృమూర్తి.
ఆమె ముఖంలో మాత్రం ఎనలేని ప్రశాంతత గోచరిస్తోంది. "అమ్మా" అని కూలిపోయాడు రాజా.
మర్నాటి ఉదయం ఆ పవిత్ర మాతృమూర్తిని కృష్ణాతీరంలో అగ్నిదేవునికి కానుకగా సమర్పించి,
ఆమె చితాభస్మాన్ని పావన కృష్ణా తరంగిణిలో కలిపి వచ్చాడు రాజా. సూర్యకిరణాలు దుర్గాలయ
శిఖరంపై పడి మెరుస్తుంటే, కృష్ణమ్మ, సుబ్బమ్మ సాహచర్యంతో తన పవిత్రత హెచ్చినట్లు పడి హెచ్చించి
పరుగులిడసాగింది.

(ఆంధ్రపత్రిక 1962 (26-10-62) దీపావళి కథల
పోటీలలో రెండవ బహుమతి పొందిన కథ)

Fig. 3. Input image for previous and improved OCR system

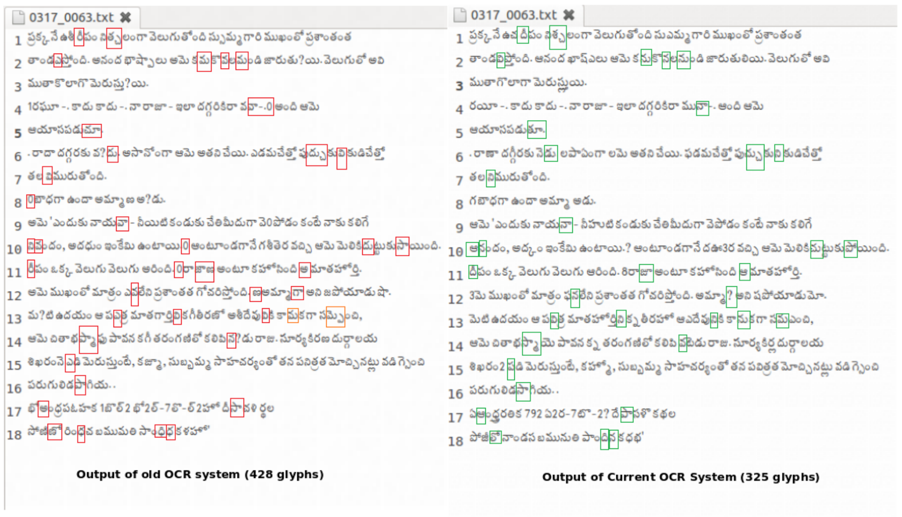

0317_0063.txt
0317_0063.txt
Output of old OCR system (428 glyphs)
Output of Current OCR System (325 glyphs)

Fig. 4. Output image for input image in Fig. 3; given to previous and improved OCR system

Table 3 shows the performance of the Previous OCR and Improved OCR system. We took different pages from different books and evaluated the performance over those pages; some of the good improvements are shown in Table 3.

Table 3. Performance of Previous and Improved OCR system on some pages.

Page Number	Novel Name	Err-Rate Previous OCR	Err-Rate Improved OCR	Performance Improvement
0320_0293.tifUNIocr	OoragaayaNavvindi	46.32	37.87	8.44
0320_0261.tifUNIocr	OoragaayaNavvindi	20.59	12.42	8.16
0320_0191.tifUNIocr	OoragaayaNavvindi	16.29	9.06	7.22
0320_0079.tifUNIocr	OoragaayaNavvindi	14.88	7.96	6.92
0317_0063.tifUNIocr	KRKMohankathalu	21.42	14.53	6.88
0317_0118.tifUNIocr	KRKMohankathalu	20.69	13.84	6.85
0317_0167.tifUNIocr	KRKMohankathalu	16.54	9.82	6.71
0317_0010.tifUNIocr	KRKMohankathalu	18.89	12.25	6.63
0011_0005.tifUNIocr	RambabuDiarypart1	13.38	5.71	7.67
0011_0122.tifUNIocr	RambabuDiarypart1	17.86	12.09	5.76
0011_0075.tifUNIocr	RambabuDiarypart1	14.36	9.65	4.70
0011_0042.tifUNIocr	RambabuDiarypart1	16.34	11.70	4.63
0120_0103.tifUNIocr	GurajadaRachanalu	71.42	50.00	21.42
0120_0079.tifUNIocr	GurajadaRachanalu	50.00	38.46	11.53
0120_0167.tifUNIocr	GurajadaRachanalu	17.88	10.12	7.75
0120_0032.tifUNIocr	GurajadaRachanalu	16.45	10.04	6.41

The table 4 contains the previous OCR error rate and improved OCR error rate and their differences (performance improvement) on 4525 pages of 27 Novels. The table 4 clearly depicts that the overall performance of the improved OCR system is increased and improved by 1.22%. For some books the performance is good and for some it is bad, the reason for this is that the numbers of glyphs present in improved OCR system are less as improved OCR system contains only 325 glyphs but the previous system contains 428 glyphs.

Table 4. Results and comparison of Previous and Improved OCR system on different books

Book Labels	Novel Name	Err-Rate Previous OCR	Err-Rate Improved OCR	Performance Improvement
0317	KRKMohankathalu	16.01	12.39	3.61
0320	OoragaayaNavvindi	18.06	14.72	3.33
0318	Vaikunthapalli	20.02	16.97	3.05
0001	Aashyapadham	13.04	10.83	2.20
0044	BankimchandraChatterjee	16.19	14.09	2.10
0319	Pillalakathalu	23.24	21.38	1.86
0010	DivamVaipu	7.39	5.61	1.78
0120	GurajadaRachanaluKavithalu	19.39	17.70	1.69
0011	RambabuDiarypart1	12.93	11.36	1.57
0321	GVSNavalalukathalu	21.95	20.42	1.53
0105	Gangajaatara	7.69	6.34	1.35
0045	Chadapurugu	8.36	7.13	1.23
0324	PalnatiViracharithra	8.75	7.55	1.20
0323	Daankwikostsaahasayaatralu	10.44	9.27	1.17
0042	RaniLakshmibai	19.73	18.58	1.15
0322	BalaGayyaluGeyakathalu	12.60	11.56	1.04
0063	Rajasekharacharitra	7.77	6.80	0.97
0103	GurajadaRachanaluKathanikalu	8.01	7.80	0.93
0357	AnnamayyaSankeerthanalu	14.15	13.41	0.74
0013	Jeevansmruthulu	9.79	9.45	0.34
0325	VijayaVilasamu	11.48	11.33	0.15
0015	RambabuDairypart2	14.36	14.23	0.13
0125	Kalaapoornodayamu	9.94	10.07	-0.13
0122	PanduranagaMahatyam	10.20	10.68	-0.48
0127	SrungaraNyshadam	8.19	8.80	-0.61
0136	Vasucharitramu	27.41	28.09	-0.68
0137	SassankaVijayamu	30.24	33.27	-3.03

5 Conclusion

In this paper we proposed the SVM with fringe as feature for Telugu OCR. The idea is to improve the recognition accuracy of the existing system. The existing OCR system with Nearest Neighbor algorithm was not giving good performance over some book as they contain broken characters. Our proposed algorithm with inverse fringe as a feature gives good results for maximum books available in Dataset. Experiments in section 4, shows that the performance of SVM is better if we use inverse fringe distance map as a feature instead of fringe distance map. From the experiments we found that SVM is also good for handling large number of classes. Then we integrated our proposed SVM with inverse fringe distance map as a feature to the previous OCR system and tested over large data-set and found that the system performance is improved and increased.

References

1. Xiao-Xiao Niu, Ching Y. Suen: A novel hybrid CNN-SVM classifier for recognizing handwritten digits. Centre for Pattern Recognition and Machine Intelligence, Concordia University, Suite EV003.403, 1455 de Maisonneuve Blvd. West, Montreal, Quebec, Canada H3G 1M8
2. S. Abe: Support vector machines for pattern classification. Springer-Verlag, London (2005).
3. R. L. Brown: The Fringe Distance Measure: An Easily Calculated Image Distance Measure with Recognition Results Comparable to Gaussian Blurring. IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS, VOL. 24, NO. I , JANUARY 1994.
4. Atul Negi, Chakravarthy Bhagvati, B. Krishna: An OCR System for Telugu. IEEE Document Analysis and Recognition, Sixth International Conference, 2001.
5. P. Pavan Kumar, C. Bhagvati, A. Negi, A. Agarwal, and B. L. Deekshatulu: Towards improving the accuracy of Telugu OCR systems. 2011 International Conference on Document Analysis and Recognition, pages 910914, 2011.
6. C. J. Burges: A tutorial on support vector machines for pattern recognition. Data Mining and Knowledge Discovery, 2(2): 121 – 167, 1998.
7. C. C. Chang and C. J. Lin: LIBSVM: a library for support vector machines. 2001, Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm
8. O. Trier and A. K. Jain: Goal-directed evaluation of binarization methods. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 17, no. 12, pp. 1191–1201, 1995.
9. K. Y. Wong, R. G. Casey, and F. M. Wahl: Document analysis system. IBM Journal of Res. Develop., vol. 26, no. 6, pp. 647–656, 1982.
10. S. Rajasekaran and B. Deekshatulu: Recognition of printed Telugu characters. Computer Graphics and Image Processing, vol. 6, no. 4, pp. 335–360, 1977.
11. A. K. Pujari, C. D. Naidu, M. S. Rao, and B. C. 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.
12. V. Govindaraju and S. Srirangaraj: Guide to OCR for Indic Scripts. Advances in Pattern Recognition, Springer (2010).
13. C. V. Lakshmi and C. Patvardhan: An optical character recognition system for printed Telugu text. Pattern Analysis and Applications, vol. 7, no. 2, pp. 190–204, 2004.
14. C. V. Lakshmi, R. Jain, and C. Patvardhan: OCR of printed Telugu text with high recognition accuracies. Computer Vision, Graphics and Image Processing, pp. 786–795, 2006.
15. P. Pavan Kumar, Chakravarthy Bhagvati, Atul Negi, Arun Agarwal, Bulusu Lakshmana Deekshatulu: Towards Improving the Accuracy of Telugu OCR Systems. ICDAR pp. 910-914, 2011.