2009 10th International Conference on Document Analysis and Recognition

# Robust Recognition of Documents by Fusing Results of Word Clusters

Venkat Rasagna[1], Anand Kumar[1], C. V. Jawahar[1], and R. Manmatha[2]

[1] Center for Visual Information Technology,
IIIT, Hyderabad, 500032, India
{rasagna@research.,anandkumar@research.,jawahar@}iiit.ac.in

[2] Department of Computer Science,
University of Massachusetts Amherst, MA 01003, USA
manmatha@cs.umass.edu

## Abstract

*The word error rate of any optical character recognition system (OCR) is usually substantially below its component or character error rate. This is especially true of Indic languages in which a word consists of many components. Current OCRs recognize each character or word separately and do not take advantage of document level constraints. We propose a* document level OCR *which incorporates information from the entire document to reduce word error rates. Word images are first clustered using a locality sensitive hashing technique. Individual words are then recognized using a (regular) OCR. The OCR outputs of word images in a cluster are then corrected probabilistically by comparing with the OCR outputs of other members of the same cluster. The approach may be applied to improve the accuracy of any OCR run on documents in any language. In particular, we demonstrate it for Telugu, where the use of language models for post-processing is not promising. We show a relative improvement of 28% for long words and 12% for all words which appear at least twice in the corpus.*

## 1. Introduction

Scanned document images are a major part of many large digital libraries. Access to such material is dependent on having good recognizers. While good recognizers exist for many European languages that perform with high accuracy at least on much commonly used material, for many non-European languages, optical character recognizers (OCRs) are not very accurate and there is a pressing need for improvement in the accuracy of these OCRs.

This paper presents a document level recognizer which incorporates a technique for correcting errors of OCRs and is applicable to all languages. The proposed method clusters word images from a collection of documents using a fast and accurate clustering technique based on locality sensitive hashing (LSH). Then, words from each cluster are recognized using an OCR. Errors in recognition are identified by aligning characters and using either a majority vote over characters or alternatively a dynamic time warping technique over the words in the cluster. The identified errors are corrected by a probabilistic character determination technique. Results are demonstrated on a scanned book in Telugu. The relative improvement in accuracy for words which occur at least twice is about 12% and for words above 5 symbols in length is 28%.

**Background** The recognition of Indian language documents is a challenging task due to the complexity of scripts and the non - availability of robust recognizers [6]. Many of these Indian languages have significant numbers of speakers. For example, Telugu has more than 70 million speakers, a large number of newspapers and a literature dating back several hundred to a thousand years. Recognition errors arising from the confusion of characters is a major problem for Indian language recognizers. Since many of the characters are similar, they randomly get misclassified. There are parts of characters where a minor degradation in one of them results in an appearance similar to the other. Figure 1 shows pairs of Telugu characters which are easily confused by a recognizer. A number of Indian languages have a large number of components or symbols in each word. Thus even when component level recognizers perform well on very good documents, the word level, and hence the document level accuracies are not acceptable in practical situations.

Previous attempts [2, 5] to recognize Telugu characters from printed documents were highly limited. The classification results were reported at character or even sub-character

978-0-7695-3725-2/09 $25.00 © 2009 IEEE
DOI 10.1109/ICDAR.2009.135

(a) (b) (c) (d) (e)

**Figure 1. Confusing character pairs in Telugu**

level. Due to the unavailability of a large corpus most of the experiments were done only on a limited number of pages. For example, [5] reports a component level accuracy of 92% on 2524 components while [2] reports a character classification accuracy of 97.87% on 1 million synthetically generated and degraded character sets. As we show later on, for Telugu even with high component level accuracies it still leads to a high word error rates.

Most OCR literature is focused on recognizing individual characters and words with little work on exploiting the constraints available in a book. Recent attempts at book level OCRs include [4] using recognition and verification and [11] which proposes mutual-entropy based model adaptation and demonstrates it on 10 pages. We investigate a different approach which exploits the similarity of word images in a book. This is based on the idea of word spotting [7, 8] which looks at creating clusters of word images by image matching. [7, 8] focused on searching handwritten word images. Dynamic time warping (DTW) [7] was shown to be a good solution and was exploited by [9] to index a collection of 500 scanned printed books in Indian languages. The one weakness of dynamic time warping is that it is slow and hence [3] showed that locality sensitive hashing could instead be used to index and search word images rapidly within a book. However, [3] did no recognition. LSH is an efficient technique to compute approximate nearest neighbours in high dimension spaces.

This paper instead uses locality sensitive hashing for clustering word images to improve recognition results and also use two alignment techniques - the first one is an approximation base technique called character majority voting and the other one based on dynamic time warping to improve OCR accuracy. It is interesting to note that Tao and Hull [10] proposed the use of word image clusters to improve OCR accuracy. Their clusters were built using simple image matching techniques like Euclidean distance matching and they used a simple majority voting technique with dictionary lookup to improve accuracy on 16 pages of English degraded using synthetic means. We note that their image matching techniques [7] are not very accurate or fast for the book level application we envisage and their majority voting technique is not good enough. The use of language models for post processing is not very promising for many Indian languages like Telugu since the large vocabulary (number of possible words) makes dictionaries infeasible and it is necessary to model joint probabilities at the sub-UNICODE level.

## 2. Word Clustering

The proposed document level OCR recognizes words using information from clusters of similar words. First, the document images are preprocessed and segmented at the word level to generate the clusters. We use a run length segmentation algorithm (RLSA). For rapid clustering we use locality sensitive hashing (LSH) [1, 3] to create clusters for every word from the documents images. LSH has been previously used for document image retrieval [3] but not recognition.

An index is built by hashing word level features of document images. Features representing words are extracted (we use the same features as in [3] and hashed using LSH functions. The key idea in the hashing technique is to hash words using several hash functions so as to ensure that, for each function, the probability of collision is much higher for words which are similar than for those which are dissimilar. A $d$ dimensional feature $x$ from a set $P$ of word features is transformed to a point $x'$ in $d' = Cd$ dimension, where $C$ is the largest coordinate in all points of P. A hash function $g_i, i = 1 \ldots l$ is used to compute a hash code into a second level of hashes which stores the corresponding index for fast retrieval. This procedure is repeated $L$ times, therefore each point will belong to $L$ tables.

**Algorithm: Word Image Clustering**

**Require:** Word Images $W_j$ and Features $F_j, j = 1, \ldots, n$
**Ensure:** Word Image Clusters **O**

**for** each $i = 1, \ldots, l$ **do**
  **for** each $j = 1, \ldots, n$ **do**
    Compute hash bucket $I = g_i(F_j)$
    Store word image $W_j$ on bucket $I$ of hash table $T_i$
  **end for**
**end for**
$k = 1$
**for** each $i = 1, \ldots, n$ and $W_i$ unmarked **do**
  Query hash table for word $W_i$ to get cluster $O_k$
  Mark word $W_i$ with $k$
  $k = k + 1$
**end for**

Clusters of similar words are obtained by querying the index with words from document images. Given a query word feature, the $L$ first level $d'$ dimensional hash codes are determined and all the words within the corresponding second-level hash tables are retrieved. The words obtained in a cluster are marked in the documents with unique cluster numbers. The query process is repeated for every unmarked word from the document images. Thus, all words from the document images are clustered into groups of similar words.

**Figure 2. A LSH cluster. Noisy variants in red.**

Words with the same stem and little form variations are grouped together in a single cluster. Figure 2 shows an example for Telugu word images. Noisy images are also grouped together even though their OCR outputs may be different. Words with extra character/symbols are also grouped if the stem content is same. However, all word form variations are not grouped together. In our experimental dataset with 19789 words, 98.1% of the words were correctly clustered.

## 3. Word Error Correction

Even when the component level recognition rate of Indian languages is high, the word level recognition rate may be much lower since words contain multiple components as can be seen in Table 1 under the OCR column headings. The different rows are for words which are short (2-3 symbols), medium (4-5 symbols) or long ($>$ 5 symbols). Even for short words, the word accuracy rate is 16% lower than the symbol (or component accuracy). For long words the word accuracy rates are about 30% lower than the symbol accuracies. We expect this since there is a higher probability of one of the symbols being erroneous in a longer word.

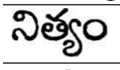
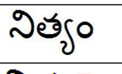
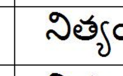
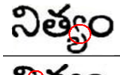
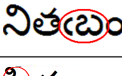
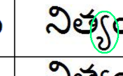
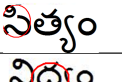
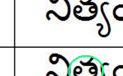
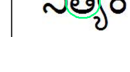
| INPUT | OUTPUT | | |
|---|---|---|---|
| Word Image | OCR | CMV | DTW |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Figure 3. Example word error correction.** 1st column word images in cluster, 2nd OCR output, 3rd CMV and 4th DTW output. Note variants and errors in red and corrections in green.

By assuming that all words in a cluster should be recognized as the same word one can improve the OCR. The first step is to use an OCR to recognize all the words in a cluster separately. The second step involves using the clusters to improve the recognition results. Two different techniques are proposed for this second step.

**Character Majority Voting (CMV):** A simple majority vote of the OCR outputs of all words would not work well. Instead we line up all the symbols of each word as in Figure 4. Then the candidates for each symbol position are chosen by selecting that character which is in the majority at each position. If no candidate is in the majority, the existing candidate is not replaced. Examples are shown later.

**Dynamic Time Warping (DTW):** The second technique utilizes dynamic programming (as in [7] to align the symbols. The algorithm for word error correction based on this approach is listed below.

**Algorithm: Word Error Correction**

**Require:** Cluster **C** of words $W_i, i = 1, \ldots, n$
**Ensure:** Clusters **O** of correct words

```
for each i = 1, ..., n do
    for each j = 1, ..., n do
        if j ≠ i then
            Align word W_i and W_j
            Record errors E_k, k = 1, ..., m in W_i
            Record possible corrections G_k for E_k
        end if
    end for
    Correct E_k if Probability p_k of correction G_k is maximum
    O ← O ∪ W_i
end for
```

Let **C** be a cluster of $n$ similar words obtained from the recognizer. Each word $w \in \mathbf{C}$ is aligned with other words $\mathbf{C} - w$ of the cluster. The alignment is obtained by a dynamic programming technique called dynamic time warping (DTW). All matching characters of two words are aligned and the unmatched characters are identified as possible errors in word $w$ of the recognizer. The unmatched characters are candidates to replace the erroneous characters in word $w$. Alignment of $w$ with all other words of the cluster gives a group $G$ of possible character replacements for errors. Hence, for every erroneous character $E_k, k = 1, \ldots m$ of $w$ there are $n - 1$ possible corrections. The probability $p_k$ of the possible correction is computed using maximum likelihood - in this case it is just the count of each candidate at that position divided by the total number of candidates at that position. If there is only one candidate at a position the probability is set to zero. A wrong character is replaced by a new character $G_k$ for which $p_k$ is maximum. The steps of this process are outlined in the error correction algorithm. This procedure is repeated to correct every word of a cluster **C**.

We will now discuss some examples on the real dataset

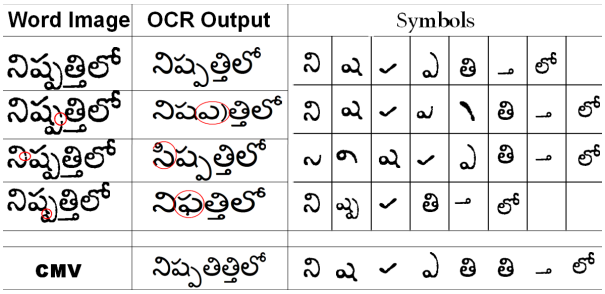



**Figure 4. CMA word correction.** 1st column word images, 2nd OCR , remaining columns the symbols are written out. Note variants and errors in red.

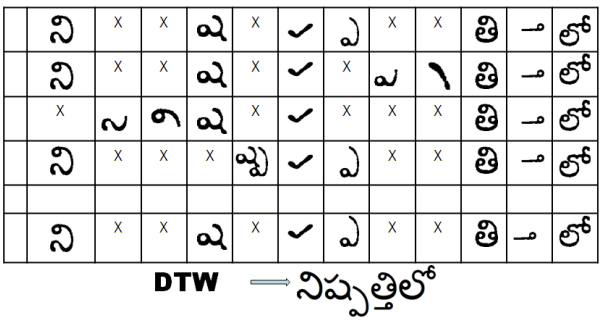



**Figure 5. DTW error correction.** First 4 rows are aligned outputs of the symbols - same words and OCR output as for the previous figure. Last row is the DTW output obtained if at least a majority of the characters agree.

(described in Section 4. Figure 3 shows an example of a cluster (first column), the OCR output (second column) and the CMV and DTW corrections. The red ovals show problems with the images in the cluster and OCR errors. The green ones show the corrected outputs. Inspite of 3 out of the 4 OCR outputs being wrong, the output can still be corrected. While a simple majority voting scheme would have failed on this example our use of CMV and DTW involves using voting over individual symbols i.e. in each position we check if a majority of the characters are correct.

Figure 4 demonstrates CMV's working. It shows word images and errors in red in the first column (some of the small red circles are breaks in the character). The OCR symbols are then expanded in order (note that Indian languages use vowels to modify consonants and hence the modifier follows the consonant in this symbol list). The majority vote technique looks at the majority in each column and assigns that to be the output which is listed at the bottom (if there is no majority one of them is arbitrarily assigned). While CMV works well in many cases in this particular case it makes mistakes.

The DTW output is shown in Figure 5. The same words and symbols as in the previous figure are used. The DTW algorithm aligns the symbols correctly creating gaps where symbols are not aligned. It produces the correct answer even though 3 out of 4 OCR words are wrong in this case!

## 4. Experiments and Results

We experiment with two different datasets. The first experiment is done on a synthetic dataset obtained by generating images of words from text and degrading them. The idea here is to assume perfect clustering and to study the effect of word length and the number of words in a cluster on the error detection and correction algorithms. The second experiment is done with a real scanned Telugu book and examines the entire process including clustering and improvements in accuracy.

### 4.1. Synthetic Dataset

This dataset has 5000 clusters. Each cluster has 20 images of the same word with different font sizes, and resolution. The words are generated from using a standard image processing library like ImageMagick. These words are then calibrated (degraded) using the Kanungo degradation model [12] to approximate real data. The word generation process makes correct annotations available for evaluating the performance of the algorithm.

The calibrated data sets were then divided into clusters with the number of words ranging from 2 to 20. This experiment examines the effect of the number of words in a cluster on accuracy. A Telugu [2] was used to generate the OCR results. We expect the error correction to get better if there are more words in a cluster but to saturate beyond a certain number of words and this is what we see in Figure 6. Figure 7 shows that accuracy also increases with word length. The OCR word error rate increases with word length which is what we would expect if the symbol error rate is constant. DTW performs consistently better than CMV.

### 4.2. Real Data

We also tested our method on a complete Telugu scanned book. Documents are preprocessed and segmented to obtain words. The component and word level accuracies of the OCRs used for recognition are shown in Table 1. Words were annotated manually for evaluation. For majority voting (CMV) the symbol accuracies go up by a few percent but word accuracies go up by 2% for short words, about 6% for medium length words and 14% for long words. DTW is even better, word accuracies go up 2% for short words, 7% for medium words and 16% for long words in absolute terms. In relative terms this is a 28% improvement. Both majority voting and dynamic programming are substantially better than just using the raw OCR.

| Size | No. of Clusters | Length Range | Word WL | No. of Words | Symbol Accuracy | | | Word Accuracy | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | OCR | CMV | DTW | OCR | CMV | DTW |
| Short | 676 | 2-3 | 2.45 | 3778 | 90.64 | 91.61 | 91.66 | 80.56 | 82.39 | 82.45 |
| Medium | 994 | 4-5 | 4.43 | 5161 | 90.78 | 92.35 | 92.42 | 73.34 | 79.14 | 80.53 |
| Long | 690 | 6-16 | 7.31 | 4587 | 89.98 | 92.15 | 92.31 | 58.64 | 72.34 | 74.82 |

**Table 1. Recognition of Clusters, WL = Word Length, CMV=Character Majority Voting**

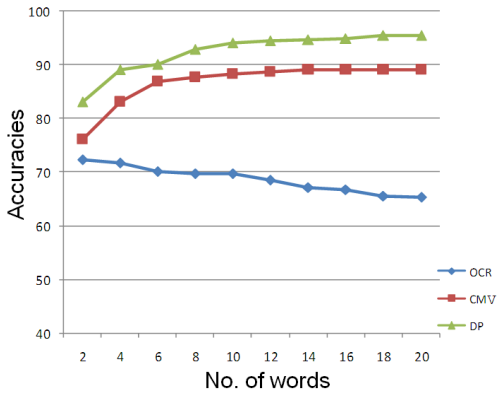



**Figure 6. Effect of Number of Words in Cluster.**

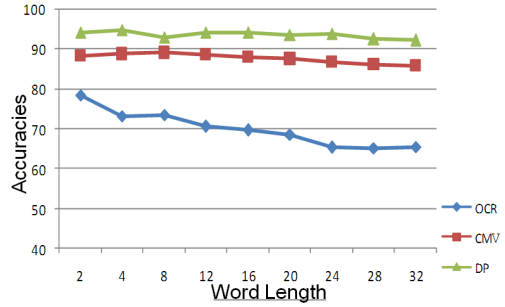



**Figure 7. Effect of Word Length.**

The average accuracy for all words which appear at least twice in the book is 70.37% for the OCR and 77.74% and 79.12% for the MVA and DTW methods respectively (or a 12% relative improvement). Our current technique can only correct words which appear at least twice (otherwise the cluster will only have one word).

## 5. Conclusion and Future Work

A document level word recognition technique is presented, which makes use of the context of similar words to improve the word level recognition accuracy. Error correction technique is presented to improve word accuracy of the raw OCR. An efficient clustering algorithm was used to speed up the process. The experimental results show that the word level accuracy can be improved significantly from about 70.37% to 79.12% for Telugu. The proposed technique may also be applied to other Indian languages. Future extensions may include the use of techniques to handle unique words by creating clusters over parts of words. We will also do tests over more books and languages.

## References


[1] M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokni. Locality-sensitive Hashing Scheme Based on p-stable Distributions. *SOCG*, pages 253–262, 2004.

[2] C. V. Jawahar, M. N. S. S. K. P. Kumar, and S. S. Ravikiran. A bilingual ocr for hindi-telugu documents and its applications. *ICDAR*, pages 408–413, 2003.

[3] A. Kumar, C. V. Jawhar, and R. Manmatha. Efficient Search in Document Image Collections. *ACCV*, pages 586–595, 2007.

[4] N. V. Neeba and C. V. Jawahar. Recognition of Books by Verification and Retraining. *ICPR*, 2008.

[5] A. Negi, C. Bhagvathi, and B. Krishna. An OCR System for Telugu. *ICDAR*, pages 1110–1114, 2001.

[6] U. Pal and B. Chaudhuri. Indian script character recognition: A survey. *Pattern Recognition*, 37(9):1887–1899, 2004.

[7] T. Rath and R. Manmatha. Word Image Matching Using Dynamic Time Warping. *CVPR*, 2:521–527, 2003.

[8] T. M. Rath and R. Manmatha. Word spotting for historical documents. *IJDAR*, 9(2):139–152, 2007.

[9] K. P. Sankar and C. V. Jawahar. Probabilistic Reverse Annotation for Large Scale Image Retrieval. *CVPR*, 2007.

[10] H. Tao and J. Hull. Improving ocr performance with word image equivalence. *Fourth Symposium on Document Analysis and Information Retrieval, UNLV, Las Vegas*, pages 177–190, 1995.

[11] P. Xiu and H. S. Baird. Whole-book recognition using mutual-entropy-driven model adaptation. *SPIE*, 2008.

[12] Q. Zheng and T. Kanungo. Morphological Degradation Models and Their Use in Document Image Restoration. *ICIP*, pages 193–196, 2001.