2011

An Efficient Framework for Searching Text in Noisy Document Images

Ismet Zeki Yalniz
R. Manmatha, *University of Massachusetts - Amherst*

An Efficient Framework for Searching Text in Noisy Document Images

Ismet Zeki Yalniz, R. Manmatha
Department of Computer Science
University of Massachusetts, Amherst
Amherst, MA, USA, 01003
Email: {zeki,manmatha}@cs.umass.edu

***Abstract*—An efficient word spotting framework is proposed to search text in scanned books. The proposed method allows one to search for words when optical character recognition (OCR) fails due to noise or for languages where there is no OCR. Given a query word image, the aim is to retrieve matching words in the book sorted by the similarity. In the offline stage, SIFT descriptors are extracted over the corner points of each word image. Those features are quantized into visual terms (visterms) using hierarchical K-Means algorithm and indexed using an inverted file. In the query resolution stage, the candidate matches are efficiently identified using the inverted index. These word images are then forwarded to the next stage where the configuration of visterms on the image plane are tested. Configuration matching is efficiently performed by projecting the visterms on the horizontal axis and searching for the Longest Common Subsequence (LCS) between the sequences of visterms. The proposed framework is tested on one English and two Telugu books. It is shown that the proposed method resolves a typical user query under 10 milliseconds providing very high retrieval accuracy (Mean Average Precision 0.93). The search accuracy for the English book is comparable to searching text in the high accuracy output of a commercial OCR engine.**

***Keywords*-document image search; image retrieval; word spotting;**

I. Introduction

One way to search scanned books is to recognize the characters and perform regular text search. However, the optical character recognition (OCR) output may have high rates of errors due to many factors such as high document degradation, unusual font type etc. As a result the search over the OCR output is less accurate. For example, old German texts printed in "Fraktur" are not recognized accurately by standard OCR engines and therefore the OCR output is typically not human-readable. Besides, there are a number of scripts such as Telugu and Ottoman for which no OCR engine is available [10], [14]. In these cases searching books using the OCR output is not applicable.

Another method is to use image search mechanisms for searching document images. The problem with image search methodologies is that they require computationally heavy operations due to the high dimensionality of the data. Typically they do not scale up for large image collections. However, there are several ways to speed up image search engines. One option is to quantize and/or index image features and retrieve them whenever necessary [2], [9], [4], [11]. Another common practice is to gain speed by sacrificing retrieval accuracy. When these two mechanisms are coupled, image search methodologies become practical for very large collections.

Here, we propose an efficient image search framework for searching text in noisy document images. The proposed methodology relies on two components: Off-line processing and a filtering stage for fast query resolution. The offline stage is run only once for each book and it consists of extracting and quantizing image features from the word images. More specifically, SIFT [5] descriptors are extracted for each corner point detected by the Fast-Corner-Detection algorithm [8]. These features are later quantized using the hierarchical K-Means clustering algorithm (HIKMEANS). The final output of the off-line processing stage is a number of word images each of which is represented by a set of corner points and their corresponding cluster IDs (i.e., "visual terms" or simply "visterms").

In the online stage (or the query resolution stage), a query word is selected by the user and all the words in the book are ranked according to their similarity to the query word. The similarity search consists of two components. The first one is called the "coverage test" and it accounts for the common visterms between the query and test image. Coverage scores for each word image are efficiently calculated using an inverted file for the visterms and they are used to filter out words which are not likely to be a match. In the second stage the configuration of visterms on the image plane is efficiently compared to those of the query image. Finally, a similarity score is calculated which accounts for both the existence and the configuration of common visterms which agree with the query image. The output of the search is a ranked list of word images from the book. The proposed framework is tested on two Telugu and one English books and it is shown to be effective in resolving queries under 0.01 second.

The paper is organized as follows. Our framework is elaborated first in Section II. Experimental results and future research directions are discussed in Sections III and IV.

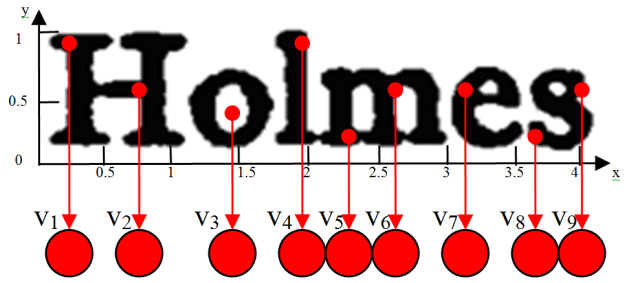

Figure 1. A height normalized word image and its visterms projected onto the X axis. Typically there are around 100 visterms per word image.

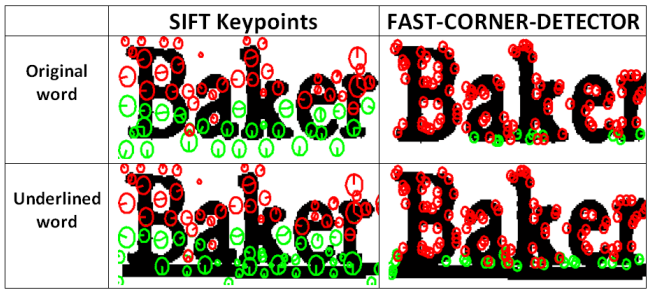

Figure 2. The top row shows all the keypoints obtained using SIFT and Fast-Corner-Detector respectively for the word image"Baker". On the bottom row, keypoints are extracted for the same image except that the word is underlined. Red circles indicate the keypoints which are preserved in both cases in spite of the noise.

II. An Efficient Framework for Searching Document Images

A. *Offline Processing*

1) Keypoint localization: The offline processing starts with defining a number of salient points in the document images (See Fig.1). These points are also called "keypoints". It is desirable to have keypoints placed on top of or near the text. Keypoints must be repeatable for matching purposes, i.e., matching keypoints must be identified for different instances of the same word image.

Two different approaches were for detecting keypoints: Fast-Corner-Detector [8] and Scale Invariant Feature Transform (SIFT) [5]. Fast-Corner-Detector finds corners points in images in a fraction of a second. These corner points can be used to extract local image features. Fast-Corner-Points are claimed to be more repeatable than well-known SIFT keypoints [8]. Unlike Fast-Corner-Detector, SIFT is capable of extracting scale and rotation invariant keypoints which are shown to be distinctive especially in natural images.

In Figure 2, extracted keypoints are depicted for the word image "Baker". The total number of keypoints are almost the same for the SIFT and Fast-Corner-Detector. It is seen that the SIFT features are distorted heavily at the bottom of the word image around the noisy region. Distortions include keypoint insertion, deletion and misplacement, or any change in the keypoint features such as the scale and orientation. On the right column, it is seen that Fast-Corner-Detector is much more repeatable than the SIFT and it is more likely to locate the same corner in spite of the noise. Therefore Fast-Corner-Detector is used for keypoint localization. Note that OCR fails on underlined words.

2) Feature Extraction: Once keypoints are identified, an image patch is placed over each of them in order to extract local image features. SIFT descriptors are used in this study. [5]. Using conventional parameters, a feature vector of size 128x1 is obtained for each keypoint. The SIFT keypoint detector provides intrinsic scale and orientation for each keypoint automatically. However, it is not the case with the Fast-Corner-Detector. Therefore, when the Fast-Corner-Detector is used, the patch size is defined to be equal to the height of the bounding box that the keypoint belongs to, and the patch orientation is assumed to be zero degrees for all keypoints. Notice that these assumptions are applicable if and only if the bounding boxes are available and the document images do not have significant page skew.

3) Feature Quantization: Using high dimensional features for matching word images is computationally expensive. One well-known practice is to map feature vectors to discrete values using clustering techniques [6]. Each feature vector is given a discrete label according to the cluster it belongs to. This label is referred as a "visterm ID". The size of the visterm vocabulary is equal to the number of clusters defined in the clustering processing.

In this framework, hierarchical K-Means (HIKMEANS) is utilized for quantizing the SIFT descriptors [12]. HIKMEANS requires the total number of clusters to be defined a priori. The vocabulary size is an important parameter because the matching performance is known to be sensitive to the vocabulary size, depending on the application. For matching natural images, use of larger vocabularies is shown to perform better [7]. However, in the context of text recognition it is desirable to find a number of matching keypoints between relevant word images despite the noise, variations and difference in font. If the vocabulary size is very large, then matching keypoints are very unlikely to get the same visterm ID even though their feature vectors are quite similar. If the vocabulary size is small, then larger number of visterms can be matched despite the noise and variations. For example, the vocabulary size is set to 4K for the word image in Figure 3. Each red dot represents a keypoint. Notice that some of the keypoints are very close to each other, therefore their local image features are also similar but not identical. Indeed these keypoints provide evidence for the existence of certain sections of the ink (for ex. the tip of the character "k"), therefore it is desirable to assign the same visterm ID for the keypoints belonging to a specific section. It is observed that using smaller vocabularies therefore yield higher matching performance.

4) Indexing Visterms: A word image is represented by its visterms which are sorted according to their X coordinates in the image plane as shown in Figure 1. An optimized version of an inverted index is also created offline for keeping all `<word ID, visterm ID>` pairs. The inverted index is later used to efficiently find the common visterms between the query word and the test image.

B. Query Resolution

Given a query word image, the aim is to identify similar word images in the context of the book. The existence of common visterms is necessary but not sufficient to qualify a word image for being a match. Their spatial configuration has also to be consistent with the ones in the query word.

Here we devise a two stage similarity search framework for matching word images. First, the common visterms are identified and weighted to eliminate false matches. This stage is referred as the coverage test. Next, a configuration score is calculated which accounts for the spatial arrangement of common visterms between the query word and each test image. Finally all word images are ranked based on a final similarity score which is a linear combination of the coverage and configuration scores:

$$Sim(I,Q) = \lambda Cover(I,Q) + (1-\lambda)Config(I,Q) \quad (1)$$

where λ is a weighting parameter, I and Q are the sequence of visterms (sorted based on their X coordinates) of the test word and the query image respectively.

One problem is that there are multiple visual terms positioned next to each other in the word image and they have exactly the same visterm ID as shown in Figure 3. Indeed, these visterms are artifacts of keypoint detectors and they do not provide any further evidence for resolving queries. It is not desirable to account for such visterms more than once for scoring. A remedy for this problem is to account for the existence but not the frequency of the visterms in word images. Therefore the coverage test does not account for the visterm frequencies.

1) Coverage Test: The coverage score simply accounts for the ratio of common visterms to the ones in the test image. There are certain visterms which are rare in the sense that they occur less frequently in the whole book but give strong evidence for the existence of certain letters. In order to incorporate this information, each visterm is given a weight which is inversely related to its collection frequency. More specifically,

$$Cover(I,Q) = \frac{\sum_{i \in I \cap Q} w_i}{\sum_{j \in I} w_j} \quad (2)$$

The weight w_i for the visterm i is defined as

$$w_i = \frac{1}{\log(f_i + 1)} \quad (3)$$

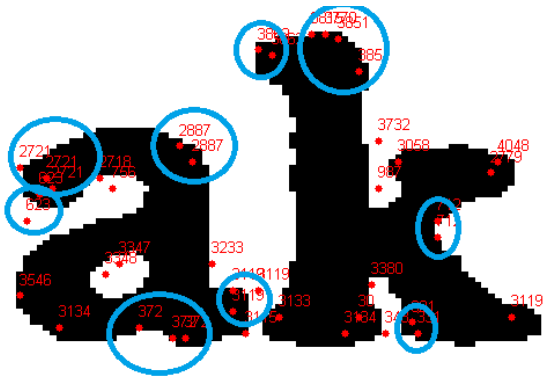

Figure 3. Corner points and corresponding visterm IDs for a letter bigram image. Visterms having the same ID are shown in circles. Notice that some visterms are spatially very close and therefore image features extracted from these regions are almost identical.

where f_i is the frequency of the visterm i in the whole book. After ranking word images based on the coverage score, we filter out the word images which are not in the top 10% of the list. The rationale behind this approach is that the total number of true matches is not expected to be larger than the frequency of the most frequent word in the language of the book. For example, "the" is the most frequent word in English and constitutes approximately 6% of an English text. The result set for the query "the" should not therefore include more than 10% of the book despite the existence of a large number of false matches. Given that typical user queries consists of infrequent words, such as names and places, it is quite unlikely to miss any true match in the filtering stage.

2) Configuration Matching: One way to verify the configuration of visterms on the image plane is to search for a transformation matrix for the visterms in the query to the test image. A well-known approach is the RANSAC algorithm [3]. In a nut-shell, RANSAC randomly selects a number of visterms in the query image and calculates a transformation matrix that maps them to the other image plane. This process is applied iteratively N times and the best transformation matrix is returned as the result. On every iteration, RANSAC fits a transformation matrix and calculates the quality of the fit by iterating over all visterms which makes it computationally expensive.

Here we devise an efficient method for testing the configuration of visterms between two word images. The reasoning is that the respective order of letters is not supposed to change along the X axis. This is true even for text written in different fonts, faces and sizes. See Figure 4. Therefore it is sufficient to project the visterms on the X axis and compare the resulting sequence of visterms. Namely, there have to be a large number of visterms having the same order in both sequences. The problem turns out to be a search for the longest common subsequence (LCS) which can be

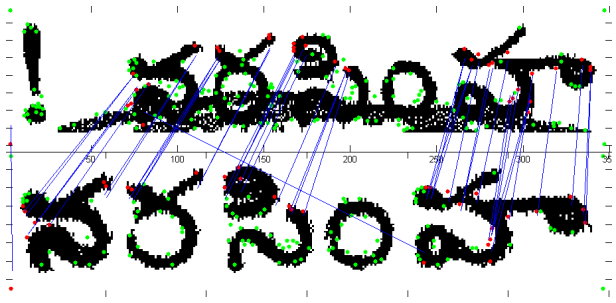

Figure 4. Matching visterms between two instances of a Telugu word from Telugu-1718 are shown. There are a large number of matching visterms following the same order even though the top image is underlined.

solved quite efficiently for short sequences using dynamic programming [1]. Here we use the length of LCS to calculate the configuration similarity as follows:

$$Config(I,Q) = \frac{\sum_{i \in LCS(I,Q)} w_i}{\sum_{j \in Q} w_j} \quad (4)$$

The numerator is the weighted sum of the visterms in the LCS(I,Q) and the denominator is the weighted sum of all visterms in the query image Q. The configuration score has a range $[0, 1]$ and it is 1 if the two sequences are identical and 0 if they do not have any common visterm.

III. Experiments

A. Datasets

Three books are used for the experiments. Two of them are printed in Telugu script and they are referred as "Telugu-1716" and "Telugu-1718". These books contain word bounding box information along with the ground truth text. The other book is "Adventures of Sherlock Holmes" written by Arthur Conan Doyle in English. Document images and the OCR output (ABBYY FineReader 8.0) are downloaded from the Internet Archive's website [1]. In total there are 363 document images including 113,008 English words. The OCR output also contains bounding box information for each recognized word. For evaluation purposes, a noise-free version of the same text is downloaded from the Project Gutenberg's website [2]. For labeling word bounding boxes, the OCR output and the ground truth text are aligned using a text alignment tool [15]. The estimated character accuracy for the whole book is 98.4%. Punctuations are ignored at all stages. A query test set is generated for each book. Each of these sets contains 50 word images respectively. These word images are randomly selected from the book itself among the ones which appear at least three times in the ground truth text. For the English book, the estimated OCR accuracy is 92.3% for the words in the query test set.

[1]The Internet Archives: Digital Library, www.archive.org, 2011
[2]Project Gutenberg: Free ebooks, www.gutenberg.org, 2011

B. Learning the Vocabulary Tree

The visual vocabulary is learned from the image features extracted from the book itself. For this purpose 10% of the pages in the book are randomly selected and the image features extracted from these pages are used for building the vocabulary tree. Once the vocabulary tree is built, image features in the rest of the documents are discretized by searching for the nearest neighbor in the vocabulary.

The size of the image vocabulary is an important parameter for matching word images. In our experiments it is seen that the use of a smaller vocabulary (of size 4K) yields better results for matching points in word images.

C. Performance Evaluation

The aim is to investigate the effectiveness of the proposed image search framework given a particular query. For this purpose two types of experiments are performed. The first one is to compare the regular text search over the OCR output to the image search. Notice that image search is case-sensitive whereas text search is not, because the image features extracted from upper and lower case letter are different because of the appearance. In order to make the evaluation fair, we only focus on single word search where text search is also case-sensitive. We do not employ any advanced query evaluation techniques for both text and image search, such as query expansion, stemming etc.

Table I
MAP scores comparing the document image search and OCR text search for the English Book

Book	Search Method	MAP
English Book	OCR text search	0.923
English Book	image search	0.93

Table I compares OCR text search to our image search framework. The Mean Average Precision (MAP) measure is used for evaluating ranked lists. OCR text search was not successful in retrieving 8% of the true positives. Therefore its MAP is estimated to be 92%. MAP score for the image search is better than the regular text search for this particular book even though the OCR accuracy is very high.

Table II
MAP scores of the proposed image search framework for the Telugu books

Book	#words	MAP
Telugu-1716	21142	0.93
Telugu-1718	4284	0.94

Table II shows the MAP scores for the Telugu books. Since there is no OCR engine for Telugu, we can not compare the image search with OCR text search for these books. However, it is clear from the MAP scores that image search is quite effective in searching Telugu books.

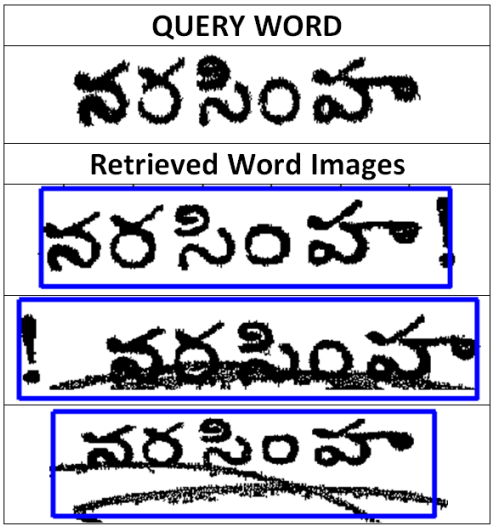

Figure 5. Example Telugu word images which are correctly retrieved using our methodology.

Figure 5 shows the returned word images for the query word at the top. Notice that connected component analysis or contour based approaches would fail when word images are underlined or connected by ink. We make use of the sections of letters which are not corrupted by the noise. This information provides strong evidence for being a match.

D. Computational Complexity

In our implementation, offline processing for a document image (12 megapixel) takes about 30 seconds. 96% of the processing is the extraction of SIFT descriptors [12]. The remaining time is spent on locating corner points and discretization. Using a GPU implementation of SIFT [13], the offline processing would take less than 5 minutes for a book with 200 pages and 100MB of main memory is sufficient for online queries. Efficient indexing of visterms ensures that resolving a single query takes about 0.01 second.

IV. Discussion and Future Work

An image search framework is proposed for searching noisy document images. It is shown that the retrieval accuracy of the proposed framework is comparable to the regular text search on books for which the OCR accuracy is very high. Image search is also shown to be effective for searching two Telugu books for which there is no OCR engine available. Future work includes improving the retrieval performance using other image features and repeating the experiments for other languages and scripts.

Acknowledgment

We would like to thank Pramod Sankar and C. V. Jawahar for providing us the Telugu datasets with ground truth. This work was supported in part by the Center for Intelligent Information Retrieval and in part by NSF grant #IIS-0910884. Any opinions, findings and conclusions or recommendations expressed in this material are the authors' and do not necessarily reflect those of the sponsor.

References

[1] S. Deorowicz. Solving longest common subsequence and related problems on graphical processing units. *Softw. Pract. Exper.*, 40:673–700, July 2010.

[2] S. Feng and R. Manmatha. A discrete direct retrieval model for image and video retrieval. In *CIVR*, pages 427–436, 2008.

[3] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24:381–395, June 1981.

[4] A. Kumar, C. V. Jawahar, and R. Manmatha. Efficient search in document image collections. In *Proceedings of the 8th Asian conference on Computer vision - Volume Part I*, ACCV'07, pages 586–595, Berlin, Heidelberg, 2007.

[5] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60:91–110, November 2004.

[6] D. Nister and H. Stewenius. Scalable recognition with a vocabulary tree. In *Proceedings of CVPR*, pages 2161–2168, 2006.

[7] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[8] E. Rosten and T. Drummond. Machine learning for high-speed corner detection. In *European Conference on Computer Vision*, volume 1, pages 430–443, May 2006.

[9] K. P. Sankar and C. V. Jawahar. Probabilistic reverse annotation for large scale image retrieval. In *CVPR*, 2007.

[10] P. Sankar K., C. V. Jawahar, and R. Manmatha. Nearest neighbor based collection ocr. In *Proceedings of the 9th IAPR International Workshop on Document Analysis Systems*, DAS '10, pages 207–214, New York, NY, USA, 2010. ACM.

[11] T.Rath, R. Manmatha, and V. Lavrenko. A search engine for historical manuscript images. In Proceedings of SIGIR'04, *pp. 297 -304*, 2004.

[12] A. Vedaldi and B. Fulkerson. Vlfeat – an open and portable library of computer vision algorithms. In *Proc. of the 18th annual ACM international conference on Multimedia*, 2010.

[13] C. Wu. SiftGPU: A GPU implementation of scale invariant feature transform. http://cs.unc.edu/~ccwu/siftgpu, 2007.

[14] I. Z. Yalniz, I. S. Altingovde, U. Güdükbay, and Ö. Ulusoy. Ottoman Archives Explorer: A retrieval system for digital Ottoman archives. *ACM Journal on Computing and Cultural Heritage*, 2(3):1–12, 2009.

[15] I. Z. Yalniz and R. Manmatha. A fast alignment scheme for automatic OCR evaluation of books. In *Proceedings of the International Conference on Document Analysis and Recognition (ICDAR)*, pages 754–758, 2011.