Segmentation-Free Keyword Retrieval in Historical Document Images

Irina Rabaev[1](✉), Itshak Dinstein[2], Jihad El-Sana[1], and Klara Kedem[1]

[1] Department of Computer Science, Ben-Gurion University, Beer-Sheva, Israel
[2] Department of Electrical and Computer Engineering, Ben-Gurion University, Beer-Sheva, Israel
{rabaev,dinstein,el-sana,klara}@cs.bgu.ac.il

Abstract. We present a segmentation-free method to retrieve keywords from degraded historical documents. The proposed method works directly on the gray scale representation and does not require any pre-processing to enhance document images. The document images are subdivided into overlapping patches of varying sizes, where each patch is described by the bag-of-visual-words descriptor. The obtained patch descriptors are hashed into several hash tables using kernelized locality-sensitive hashing scheme for efficient retrieval. In such a scheme the search for a keyword is reduced to a small fraction of the patches from the appropriate entries in the hash tables. Since we need to capture the handwriting variations and the availability of historical documents is limited, we synthesize a small number of samples from the given query to improve the results of the retrieval process.

We have tested our approach on historical document images in Hebrew from the Cairo Genizah collection, and obtained impressive results.

Keywords: Historical document processing · Keyword retrieval · Segmentation-free · Bag-of-visual-words · Kernelized locality-sensitive hashing

1 Introduction

An ongoing considerable effort for digitizing historical manuscripts have produced huge datasets. Since the documents are represented as images, it is essential to provide a search and retrieve engine that simplify and accelerate accessing and processing the manuscripts. Current Optical Character Recognition (OCR) systems perform badly when applied to degraded historical documents, which leaves keyword spotting technique as a practical alternative [15]. In keyword spotting, the retrieval is performed on the image domain, and the aim is to locate regions in the image that are similar to the keyword query image.

The majority of word spotting approaches require the input to be segmented, at least to the text line level [6,10,13,15–17]. However, in addition to the physical degradations, many handwritten documents exhibit varying line slopes and

A. Campilho and M. Kamel (Eds.): ICIAR 2014, Part I, LNCS 8814, pp. 369–378, 2014.
DOI: 10.1007/978-3-319-11758-4_40

touching characters. Segmentation of such documents often results in united or split words, and loss of ascenders and descenders. This in turn influences the results of the subsequent search algorithms. We believe that the results of keyword retrieval can be improved by employing segmentation-free approach.

In this paper we present a segmentation-free scheme to efficiently retrieve keywords in gray scale historical documents. The scheme integrates bag-of-visual-words representation (BoVW) [4] with kernelized locality-sensitive hashing (KLSH) [11] and does not require any pre-processing image enhancement. While the BoVW with KLSH have been used for object retrieval in computer vision domain [11], this is the first time such scheme is applied for segmentation-free text retrieval in document images.

In an off-line stage each document image is (logically) subdivided into overlapping patches of several sizes. The patches are described by a BoVW model, and the obtained descriptors are hashed into several hash tables. The kernelized locality-sensitive hash functions ensure, with high probability, that descriptors of visually similar patches are placed into the same entry. Thus, we pre-compute the hash entries for all the patches in our input images.

To search for a given query keyword, we generate its BoVW descriptor and obtain the hash indices of the generated descriptor in each of the hash tables. The data items from the corresponding entries are retrieved as candidates and are searched to obtain the best matches. The search is fast due to the fact that the subset of candidate patches is relatively small. Since the availability of historical documents is limited and we need to capture the handwriting variations, we synthesize a small number of samples from the given query to improve the results of the retrieval process.

The presented scheme was tested on a set of Hebrew historical documents from the Cairo Genizah collection[1], which are highly degraded. Given that our input images are not binarized, slant corrected or segmented, the results we get are very impressive.

2 Related Work

Gatos and Pratikakis [7] presented a segmentation free word spotting approach that applies binarization and skew correction, and then computes block-based image descriptors for template matching. Rusinol *et al.* [18] introduced a patch-based framework, where each document is split into a set of equal size overlapping patches, and is represented by a feature-by-patch matrix. The patches are described using bag-of-visual-words model over the extracted SIFT descriptors. The feature-by-patch matrix is further refined by applying a latent semantic indexing technique. Dovgalecs *et al.* [5] also utilized patch-based framework. First, they evaluate a distance between the features of the query and each patch. Then, the best results are filtered using longest weighted profile algorithm.

[1] The Cairo Genizah (http://www.genizah.org/) is one of the largest collections of Hebrew medieval manuscripts in the world. It contains a huge amount of documents written between the 9th and 19th centuries AD.

Almazán *et al.* [1] represented documents with a grid of HOG descriptors, which are compressed with Product Quantization in order to save the memory space. Exemplar SVM is used to learn a better representation of the keyword queries, and the regions most similar to the query are located using sliding window.

Keyword retrieval usually deals with searching a large number of items. To make large-scale search efficient, commonly an approximate nearest-neighbor (ANN) search technique is applied. However, most of the ANN algorithms suffer from the curse of dimensionality. Indyk and Motwani [8] presented a locality-sensitive hashing (LSH) technique to implement an efficient NN search on a large collection of high dimensional items. The main idea of LSH is to use several hash functions (with their corresponding hash tables) that hash similar items to the same entry with high probability. The same hash functions are used to calculate entry indices for a query, and only the items from these entries are further searched. Kumar *et al.* [12] incorporated LSH for spotting words in a collection of printed documents. The preprocessed documents are segmented into words, which are represented by a combination of scalar, profile and structural features. A Discrete Fourier Transform is applied to feature vectors and the obtained final descriptors are hashed into the hash tables. Saabni and Bronstein [19] describe the segmented word parts by multi angular descriptors. They use the boost-map algorithm for embedding the feature space with the DTW measurement to a Euclidean space. This embedding allows subsequent use of LSH for finding k-nearest neighbors of a query image. Then, the candidate images are compared to the query using the DTW distance.

3 The Methodology

The schematic overview of the presented method is depicted in Fig. 1. For our input images we pre-compute a data structure of hash tables, where document patches are stored according to their descriptors. Given a query keyword q, initial candidates similar to q are retrieved from the data structure and are further processed to obtain the final results. To capture the handwriting variations and overcome the problem of limited available samples, we synthesize a small number of various instances from the given query to improve the retrieval process.

3.1 Extracting the Patch Descriptors

The presented method begins with calculating dense SIFT descriptors on a regular grid of 5 pixels imposed over the image, similar to [5,18]. At each grid vertex three descriptors, which correspond to three spatial sizes, are calculated. These sizes are chosen with respect to the font dimensions, which are automatically approximated using the technique developed in our lab [3]. Descriptors with low magnitude are ignored, as such descriptors usually correspond to non-text areas. Once the descriptors are calculated, they are quantized into n clusters using the k-means algorithm.

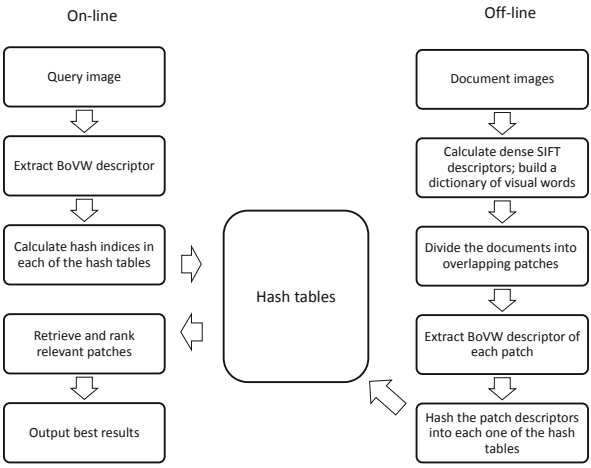


Fig. 1. The overview of the on-line and off-line stages of the presented scheme

Next, we subdivide each document into overlapping patches, sampled every p pixels in x and y directions. Previous approaches adopted equal size patches [5,18]. We chose to extract patches of several widths at each location, to take into account different word lengths. Each patch is represented by the bag-of-visual-words descriptors [4].

Let $D = \{w_1, w_2, \ldots, w_n\}$ be a dictionary of visual words. The BoVW representation is a vector $v = (v_1, v_2, \ldots, v_n)$, where v_i is the occurrence rate of w_i in the patch. Traditional BoVW representation does not take into account spatial distribution of visual words, and to overcome this limitation we impose 2×2 grid over the patch, resulting in 4 equal cells. The BoVWs of each cell are calculated and concatenated to generate the patch descriptor. This is similar to spatial pyramid matching technique presented by Lazebnik *et al.* [14], except that we use the highest pyramid level only.

3.2 Constructing the Data Structure

The aim of the data structure is to support fast search operations over a huge number of high dimensional descriptors. To accelerate the search, we use the LSH technique [8], which approximates k-nearest neighbors search on large collections of high dimensional datasets. LSH consists of l hash tables T_1, T_2, ..., T_l and l hash functions f_1, f_2, ..., f_l. Each hash function projects the objects onto randomly chosen low-dimensional Hamming space. The hash functions are constructed in such a way that the probability of the two objects to be hashed to the same entry is strictly decreasing with the distance between them. As the total number of entries may be large, to save memory space the non-empty entries are compressed using standard hashing; i.e., there are two levels of hashing: the

locality-sensitive hash functions in the first level and standard hash functions in the second level. The main assumption of the LSH is that the data objects come from Euclidean space and the distance function is Euclidean distance. In our situation we are dealing with BoVW descriptors, which are histograms, and the χ^2 distance is an appropriate measure for comparing two histograms [2]. Kulis and Grauman [11] presented kernelized locality-sensitive hashing for k-nearest neighbors searches over arbitrary kernel functions. Similar to standard LSH, the hash functions are constructed using random projections, but the projections are calculated using the kernel function and the sparse set of examples from the collection itself. We use the KLSH with χ^2-kernel, K_{χ^2}, as formulated in Eq. 1, where V_1 and V_2 are two feature vectors and d is their dimension. Finally, the extracted patch descriptors are hashed to each of the hash tables. We actually store pointers to the descriptors and not the descriptors themselves.

$$K_{\chi^2}(V_1, V_2) = \exp\left(\frac{1}{2}\sum_{i=1}^{d}\frac{(V_1(i) - V_2(i))^2}{V_1(i) + V_2(i)}\right) \tag{1}$$

3.3 The Retrieval Process

To retrieve patches similar to a query image we obtain the descriptor of the query (in the same manner as described in Section 3.1), calculate the hash indices for the hash tables, and retrieve the items from the corresponding entries. The retrieved items are ranked according to their χ^2 distance from the query. Since there are overlapping patches, from each set of patches overlapping more than 20% we pick only the patch with the smallest χ^2 distance from the query, and discard the rest. Finally, the top results are returned to the user.

Handwritten text is characterized by variations in size, slant, noise, etc. In our previous research [16] we showed that employing multiple models for a query can improve retrieval results. However, it is not always possible to get sufficient number of samples for a given pattern in historical documents. Therefore, we synthesize additional samples from the original query by applying limited resizing, slant change, dilation, erosion, and adding noise (the noise is generated according to the degradation model [9]). After generating additional samples of the query, we proceed as is described above, except that we calculate indices for all the samples of the query in each hash table. We define the distance between a patch and the samples to be the average χ^2 distance between the patch and each of the samples.

4 Experiments and Results

The proposed method was tested on 12 document images from the Cairo Genizah collection, examples of which are presented in Fig. 2. The pages exhibit a variety of degradations, such as smeared characters, bleed through, and stains.

[2] Let H_1 and H_2 be two histograms with b bins. The χ^2 distance is defined to be: $\chi^2(H_1, H_2) = \frac{1}{2}\sum_{i=1}^{b}\frac{(H_1(i)-H_2(i))^2}{H_1(i)+H_2(i)}$.

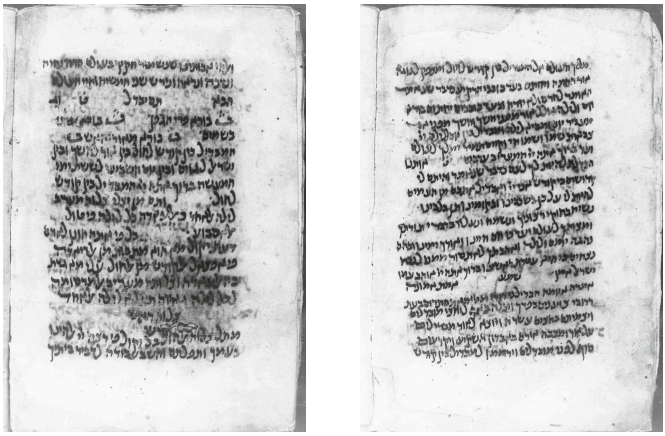


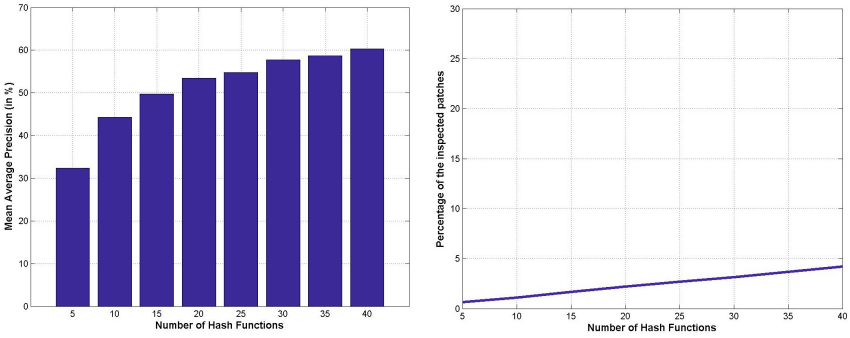
Fig. 2. Samples of the document pages on which we performed our tests

To build the dictionary, we used one of the pages and have experimented with dictionary sizes varying from 100 to 2000. The dictionary of sizes $400-500$ performed best on our document set. The patches were extracted every 25 pixels, and at each sample point we extracted patches of four sizes: 100×75, 135×75, 170×75 and 205×75 pixels (see Fig. 3a). The patches that did not contain any visual word were automatically detected and discarded. The total number of the extracted patches from all the 12 pages in our document set was 161952.

(a) (b) (c)

Fig. 3. (a) Examples of sampled patches; (b) The queries and corresponding retrieved results. The topmost image in each column is the query word; (c) The synthetically created images of two Hebrew words. The original image is the leftmost image in each group. The synthetic samples are created by re-sizing, adding noise, slant, dilating and eroding the original query.

The ground truth for the documents was manually built using the web-based system developed in our lab [2]. We randomly chose 50 queries, and the presented results were averaged over all the queries. The performance was evaluated in terms of Mean Average Precision (MAP). A retrieved patch is considered true



(a) (b)

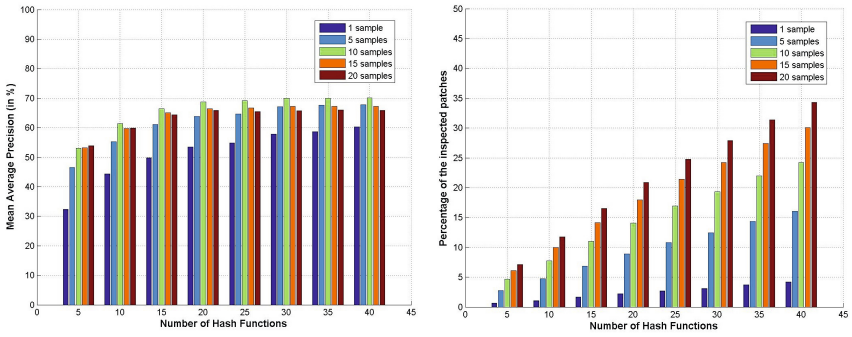
Fig. 4. (a) The performance of KLSH with varying number of hash functions and one sample per query. Percentage of the inspected patches versus the number of hash functions used. As we see, less than 5% of the patches from the database are inspected, even using 40 hash functions.

positive if it overlaps more than 50% with the bounding box of the relevant word in the document.

In the first set of experiments we analyzed how the number of hash function influences the retrieval results, when one sample per query is used. For this experiment, we varied the hash functions number from 5 to 40, and the corresponding MAPs are presented in Fig. 4a. The best being 0.6 for 40 hash functions. For comparison, the MAP of linear searches, which search over all patches, is 0.6818. As can be noted, the performance of the KLSH gets close to the results of linear search as the number of hash function increases. In contrast, the percentage of the inspected patches is less than 5% of the entire database, even when using 40 hash functions, as depicted in Fig. 4b. Due to the small fraction of inspected patches, our method (with 40 hash functions) is 10 times faster then the linear search. For comparison, we have downloaded the code provided by Almazán *et al.* [1] and used the same evaluation protocol. The results of [1] with the best configuration tuned for our documents is 0.5508. For the time being we do not compare run-time as our code still runs on Matlab and is not optimized.

Fig. 3b illustrates some retrieval results for four queries, using 30 hash functions and one query sample. The query is the topmost image in each column. As seen, the obtained results are promising for documents that have not undergone any image enhancement. Sometimes false positive words are retrieved (see the last two words in the leftmost column in Fig. 3b).

In the second set of experiments, we synthesized additional samples for each query and checked the influence of the number of samples on the performance. Fig. 3c illustrates examples of synthetic samples for two Hebrew words. The image on the left in each example is the original image, and to right of the original are its synthetically created samples. We ran experiments with 5, 10, 15



(a) (b)

Fig. 5. (a) The performance results using 5, 10, 15 and 20 samples per query; (b) Percentage of the inspected patches for varying number of samples

and 20 samples. Fig. 5a illustrates the corresponding MAPs for varying number of hash functions. We can observe a significant improvement in precision rate from 5 samples (in comparison to using one sample). On the other hand, we do not observe further improvement when we increase the number of samples above 10. This might indicate that it is enough to use 10 samples. In addition, we noticed that using small number of samples per query can compensate for the need for a large number of hash tables. For example, the results with 10 samples and 10 hash tables even slightly better than the results with one sample and 40 hash functions. Finally, Fig. 5b illustrates the influence of the number of samples on the number of the inspected patches. As seen, the fraction of the inspected patches grows rapidly with the number of samples. However, it seems that 10 samples per query and 20 hash function give the reasonable trade-off between the accuracy and the number of searched patches, which is still less than 15% of the database.

5 Conclusions and Future Work

In this paper we presented a segmentation-free approach to spot keywords in degraded handwritten historical documents. The method does not require binarization or any other image enhancement. We integrate the BoVW representation with kernalized locality-sensitive hashing to create the input data structure of hash tables and descriptors for the patches of varying size in document images. We showed that, almost without compromising accuracy, we search less that 5% of the patches even when 40 hash functions are used. Furthermore, we demonstrated that additional synthetically generated samples of the query improve the retrieval results and reduce the need for a large number of hash functions. We found that 20 hash functions suffice when we use 10 samples of the query. While our experiments focus on Hebrew handwritten historical documents, the scheme is general and can be applied to historical documents in

other languages. At future research we plan to inspect the influence of spatial pyramid co-occurrence [20] incorporated into BoVW and to perform tests on public document collections of handwritten historical documents.

Acknowledgments. This research was supported in part by the DFG-Trilateral grant no. FI 1494/3-2, the Ministry of Science and Technology of Israel, the Council of Higher Education of Israel, the Lynn and William Frankel Center for Computer Sciences and by the Paul Ivanier Center for Robotics and Production Management at Ben-Gurion University, Israel.

References

1. Almazán, J., Gordo, A., Fornés, A., Valveny, E.: Efficient Exemplar Word Spotting. In: British Machine Vision Conference, pp. 67.1–67.11 (2012)
2. Biller, O., Asi, A., Kedem, K., El-Sana, J., Dinstein, I.: WebGT: An Interactive Web-based System for Historical Document Ground Truth Generation. In: 12th International Conference on Document Analysis and Recognition, pp. 305–308 (2013)
3. Biller, O., Kedem, K., Dinstein, I., El-Sana, J.: Evolution Maps for Connected Components in Text Documents. In: International Conference on Frontiers in Handwriting Recognition, pp. 405–410 (2012)
4. Csurka, G., Dance, C., Fan, L., Willamowski, J., Bray, C.: Visual Categorization with Bags of Keypoints. In: Workshop on Statistical Learning in Computer Vision. vol. 1, pp. 1–2 (2004)
5. Dovgalecs, V., Burnett, A., Tranouez, P., Nicolas, S., Heutte, L.: Spot It! Finding Words and Patterns in Historical Documents. In: 12th International Conference on Document Analysis and Recognition, pp. 1039–1043 (2013)
6. Fischer, A., Keller, A., Frinken, V., Bunke, H.: Lexicon-free handwritten word spotting using character HMMs. Pattern Recognition Letters **33**(7), 934–942 (2012)
7. Gatos, B., Pratikakis, I.: Segmentation-free Word Spotting in Historical Printed Documents. In: 10th International Conference on Document Analysis and Recognition, pp. 271–275 (2009)
8. Gionis, A., Indyk, P., Motwani, R.: Similarity Search in High Dimensions via Hashing. In: VLDB, vol. 99, pp. 518–529 (1999)
9. Kieu, V., Visani, M., Journet, N., Domenger, J., Mullot, R.: A character degradation model for grayscale ancient document images. In: 21st International Conference on Pattern Recognition, pp. 685–688 (2012)
10. Kolcz, A., Alspector, J., Augusteijn, M., Carlson, R., Popescu, G.: A Line-Oriented Approach to Word Spotting in Handwritten Documents. Pattern Analysis and Applications **3**, 153–168 (2000)
11. Kulis, B., Grauman, K.: Kernelized Locality-Sensitive Hashing. IEEE Transactions on Pattern Analysis and Machine Intelligence **34**(6), 1092–1104 (2012)
12. Kumar, A., Jawahar, C.V., Manmatha, R.: Efficient Search in Document Image Collections. In: Yagi, Y., Kang, S.B., Kweon, I.S., Zha, H. (eds.) ACCV 2007, Part I. LNCS, vol. 4843, pp. 586–595. Springer, Heidelberg (2007)
13. Lavrenko, V., Rath, T., Manmatha, R.: Holistic Word Recognition for Handwritten Historical Documents. In: Workshop on Document Image Analysis for Libraries, pp. 278–287 (2004)

14. Lazebnik, S., Schmid, C., Ponce, J.: Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 2169–2178 (2006)
15. Manmatha, R., Croft, W.: Word Spotting: Indexing Handwritten Archives. In: Intelligent Multimedia Information Retrieval Collection, pp. 43–64 (1997)
16. Rabaev, I., Biller, O., El-Sana, J., Kedem, K., Dinstein, I.: Case Study in Hebrew Character Searching. In: 11th InternationalConference on Document Analysis and Recognition, pp. 1080–1084 (2011)
17. Rath, T., Manmatha, R.: Word Image Matching Using Dynamic Time Warping. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 521–527 (2003)
18. Rusinol, M., Aldavert, D., Toledo, R., Lladós, J.: Browsing Heterogeneous Document Collections by a Segmentation-free Word Spotting Method. In: 11th International Conference on Document Analysis and Recognition, pp. 63–67 (2011)
19. Saabni, R., Bronstein, A.: Fast Keyword Searching Using 'BoostMap' Based Embedding. In: International Conference on Frontiers in Handwriting Recognition, pp. 734–739 (2012)
20. Yang, Y., Newsam, S.: Spatial pyramid co-occurrence for image classification. In: IEEE International Conference on Computer Vision, pp. 1465–1472 (2011)