Integrating Visual and Textual Cues for Query-by-String Word Spotting

David Aldavert, Marçal Rusiñol, Ricardo Toledo and Josep Lladós
Computer Vision Center, Dept. Ciències de la Computació
Edifici O, Univ. Autònoma de Barcelona
08193 Bellaterra (Barcelona), Spain.

***Abstract*—In this paper, we present a word spotting framework that follows the query-by-string paradigm where word images are represented both by textual and visual representations. The textual representation is formulated in terms of character n-grams while the visual one is based on the bag-of-visual-words scheme. These two representations are merged together and projected to a sub-vector space. This transform allows to, given a textual query, retrieve word instances that were only represented by the visual modality. Moreover, this statistical representation can be used together with state-of-the-art indexation structures in order to deal with large-scale scenarios. The proposed method is evaluated using a collection of historical documents outperforming state-of-the-art performances.**

I. Introduction

Handwritten word spotting is the task devoted to locate the zones of a handwritten document where there is a high likelihood to find a queried word. The main interest of spotting techniques resides in the fact that they are able to provide access to collection's contents without needing to transcribe the whole document corpus. A broad taxonomy of word spotting approaches can be defined on whether the system is *learning-based*, and thus uses machine learning techniques in order to train models of the sought words, or is *example-based* and just expects to receive an example of the word to search and performs the retrieval by computing distances between the representation of the query word and the words in the collection.

Although the literature that follows the example-based paradigm is quite extensive, from the application point of view, such methods present a huge disadvantage. In order to spot a word the user needs to first locate an instance of such word that will be used as the template. In large collections, forcing the user to manually extract a template word is a really tedious task. In some cases it might even already solve the user's needs, if he just wanted to know whether a specific word appeared in the collection or not. A much more natural way of casting queries would be to allow the user to type in the keyboard the word he is searching. This is usually known in the spotting context as *query-by-string*. The first query-by-string proposed attempts such as [1], [2], [3], [4], [5] relied on the extraction of letter or glyph templates. Such character extraction was manually done [3], [4] or by means of some clustering scheme [2], [5]. When the user typed the query, such character templates were put together in order to synthetically generate an example of the sought word. Although such methods proved to be effective and user-friendly, they can just be applied in scenarios where individual characters can be easily segmented and put together again to generate the queries. In fact, they were applied to either typewritten documents [2], [3] or documents having writing styles without many typographic ligatures [1], [4].

In order to propose more generic solutions, some learning-based methods aimed at working with the query-by-string paradigm have emerged. Instead of learning a model for whole words, the main idea is to learn models for individual characters and the relationships among them. Such models, however, are trained on the whole word or even on complete text lines without needing an explicit character segmentation. Fischer et al. proposed in [6] to use HMM character models trained over complete text lines whereas Frinken et al. proposed in [7] the use of bidirectional long short-term memory neural networks to infer individual character presence. An usual inconvenient of learning-based methods is that they often require an important amount of annotated data in order to perform well. However, Rodriguez-Serrano and Perronnin recently proved in [8] that the amount of training data needed can be drastically reduced by combining handwritten examples of the word to model together with synthetic samples generated by different computer fonts.

However, we believe that such approaches still present an important drawback. Learning character models with either an HMM [6] or a NN [7] allows to *on-the-fly* generate a word model from the query the user typed in the keyboard. But this on-line generated model has to be used to process the whole collection at query time. When dealing with larger and larger datasets, the computational cost quickly becomes excessive, making such approaches not scalable at all. In contrast, example-based approaches are easily scalable to large collections since they holistically represent handwritten words by numeric feature vectors, and thus word spotting frameworks can be used together with state-of-the-art indexation strategies.

In this paper, we propose a query-by-string word spotting method where word images in the training set are represented both by a textual and visual representations. Textual descriptors are formulated in terms of character n-grams while the visual representation is based on a bag-of-visual-words scheme powered by gradient features. These two representations are merged together and projected to a sub-vector space by means of the latent semantic analysis method. This transform allows to, given a textual query, retrieve word instances that were only represented by the visual modality. To our best knowledge this is the first work that deals with the problem of query-by-string word spotting allowing to represent words in a numerical n-dimensional space which can be efficiently indexed.

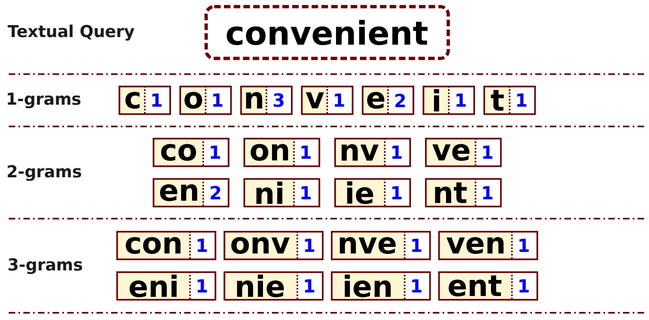

Fig. 1. Example of the n-gram textual descriptor.

The remainder of the paper is organized as follows. Both textual and visual word representations are described in Section II. Subsequently, in Section III we detail how the combination of both representations is obtained through the use of latent semantic analysis. In Section IV, we overview the retrieval step and how from a textual query we are able to retrieve word instances that are only represented visually. The experimental results are presented in Section V. We finally draw our conclusion remarks in Section VI.

II. Word Representations

As previously stated, words are represented by two different cues, one relying on textual information and the other relying on visual information. In this paper, we only focus on the word spotting problem assuming the words are already segmented by a layout analysis step. Here, segmentation errors are not considered and the word segmentation is manually generated. In this section, we will present the work-flow used to generate both representations.

A. Textual Representation

The basic block used to represent word transcriptions are n-grams of characters. The transcriptions are divided into a set of consecutive and overlapping blocks of unigrams, bigrams and trigrams respectively. This simple representation allows to extract information from which characters compile a word and encode some neighborhood information. An example of the n-gram frequencies generated by a transcription is shown in Fig. 1.

The textual descriptor $\mathbf{f}_i^t$ is obtained by simply accumulating the occurrences of each n-gram into a histogram and normalizing this histogram by its L_2-norm. The number of different n-grams available is obtained by generating a codebook which maps each n-gram into a dimension of the textual descriptor. This codebook is generated from the training set where the textual information is available. Then, in the retrieval phase, the n-grams which do not appear in the codebook are simply ignored.

B. Visual Representation

Word image snippets are represented by a descriptor obtained using the Bag-of-Visual-Words (BoVW) framework. This framework has obtained a good performance in heterogeneous problems in the computer vision albeit its simplicity and recently has shown to be a powerful tool to describe handwritten words in a word spotting task [9]. Although a thorough evaluation of the different parameters of the BoVW model is beyond the scope of this paper, we have carefully tuned the methods used at the different steps of the BoVW model generation so that we obtain good performances when visually computing similarities among words. Let us detail the followed steps.

1) Region sampling: In order to create a histogram of visual words, we first need to extract local information from the image. We densely sample local regions over the image at different scales. In this paper, the sampled local regions consist in squared regions having sizes of 20, 30 and 40 pixels which are sampled at a constant step of 5 pixels. Since we do not expect to face rotation distortions of the handwritten text, the dominant orientation of the local regions is ignored. Then, the local regions are characterized by the SIFT local descriptor using its standard configuration. Local regions having an accumulated gradient module lower than a certain threshold are rejected.

2) Encoding: Once descriptors have been sampled from the image, we need to convert them into visual words by means of a codebook. To generate the codebook, we have randomly sampled two million descriptors of the document images and used the k-means algorithm to create a codebook with 4096 codewords. Then, the local descriptors are encoded into visual words using soft-assignment which lessen the effects of encoding errors. For each descriptor, we select the three nearest codewords and weight the contribution of the selected codewords by means of the Locality-constrained Linear Coding (LLC) algorithm [10]. The number of nearest neighbors has been validated empirically using different parameters in the BoVW technique and, this value has consistently obtained better retrieval performances. Although the number of neighbors can seem small, this result is coherent with the results shown in the original paper [10] where, when using a small number of neighbors, a better performance is reached than when using a larger number neighbors.

3) Spatial Representation: The BoVW model discards all the spatial information of the local regions where the descriptors have been sampled resulting in a orderless representation. However, spatial information can greatly increase the representativity of the visual descriptor. During the creation of the visual descriptor, we have experimented using different spatial configurations and concluded that adding spatial information to the BoVW representation of handwritten words always boosts the retrieval performance of the system.

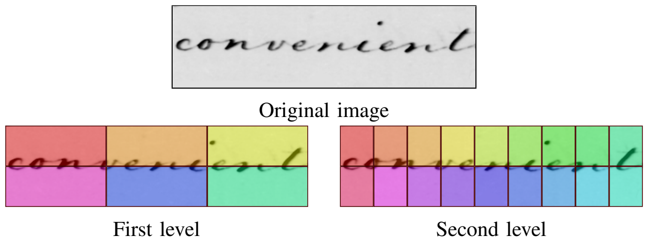

Fig. 2. Distribution of the spatial bins in the two levels of the spatial pyramid.

We have added spatial information by means of the spatial pyramid proposed by Lazebnik et al. in [11]. Experimental results show that partitions in the abscissa have a greater discriminative power than partitions in the ordinate. Therefore, we created a two level pyramid, were the first level has 3 partitions in the X axis and 2 partitions in the Y axis and the second level tripled the number of divisions in the X axis while keeping the same amount of partitions in the Y axis. Fig. 2 shows an example of the prosed configuration. Then, the histogram of visual words is generated by pooling the visual words which fall at each spatial bin in a different histogram. The histograms of visual words of each level of the pyramid are independently concatenated and normalized using the L_2-norm. This ensures that the histograms of visual words of each level of the pyramid have the same contribution to the final descriptor of visual words. Finally, the visual descriptor $\mathbf{f}_i^v$ is obtained concatenating the histograms of visual words of each pyramid level. This configuration results in 24 different spatial bins which combined with the 4096 codebook results in a 98304-dimensional visual descriptor.

4) Normalization: We use the power normalization proposed by Perronnin et al. in [12]. This method applies the following normalization function at each bin of the visual words descriptor

$$g(x) = sign(x)|x|^{\alpha},$$

where $0 < \alpha < 1$ is the power normalization factor. This normalization increases the response of the visual words with low contribution resulting in a less sparse representation. The power normalization also improves the BoVW model since it removes the assumption that visual words come from an identically and independently distributed population [13]. Finally, an L_2-normalization is applied to the whole descriptor in order to obtain the final visual representation.

III. Multimodal fusion

So far, word snippets can be represented by two kinds of information sources: a textual descriptor extracted from the available transcriptions and a visual representation generated from the visual words extracted from the document images. The first descriptor is created from the information provided by an user while the second is generated in a completely unsupervised manner. Our proposal is to bring together the textual and visual descriptors into a common representation space, so that we are able to use textual queries to retrieve word image snippets which are described solely by a visual descriptor. This is achieved by searching a transform which projects the descriptors into a different space, where the textual and the visual descriptor of the same word will be ideally projected into the same feature vector into the transformed space.

This transform is calculated by the Latent Semantic Analysis (LSA) algorithm proposed by Deerwester et al. in [14]. The LSA technique assumes that exist some underlying semantic structure between the textual and visual feature spaces. This semantic structure is defined by a set of abstract topics where each topic is a representative distribution of visual and textual features. The topics are estimated using the singular value decomposition (SVD) algorithm in an unsupervised way. This mixture of topics is going to be the new representation of the word snippets instead of their visual or textual descriptors.

In order to obtain the space transformation matrix, we first concatenate the visual descriptor $\mathbf{f}_i^v$ and textual descriptor $\mathbf{f}_i^t$ in a single merged descriptor $\mathbf{f}_i = [\mathbf{f}_i^t, \mathbf{f}_i^v]$, for each word of the training set. Then, the merged descriptors are arranged in a descriptor-by-word matrix $A \in \mathbb{R}^{D\times M}$, where D is the joint dimensionality of the visual and textual descriptors and M is the number of word examples in the training set. The LSA method obtains the transformed space by decomposing the descriptor-by-word matrix in three matrices by a truncated SVD:

$$\mathbf{A} \simeq \hat{\mathbf{A}} = \mathbf{U}_T\mathbf{S}_T\left(\mathbf{V}_T\right)^{\top},$$

where T is the dimensionality of the transformed space and $\mathbf{U}_T \in \mathbb{R}^{D\times T}$, $\mathbf{S}_T \in \mathbb{R}^{T\times T}$ and $\mathbf{V}_T \in \mathbb{R}^{M\times T}$. Note that in the LSA paradigm $T \ll D$ so that the dimensionality of the projected vectors is much smaller than the original vectors and therefore can be indexed more efficiently. Finally, the transformation matrix $\mathbf{X}_T$ is calculated as

$$\mathbf{X}_T = \mathbf{U}_T\left(\mathbf{S}_T\right)^{-1},$$

and descriptors are projected into the new feature space by simply $\hat{\mathbf{f}}_i = \mathbf{f}_i^{\top}\mathbf{X}_T$.

IV. Query-by-String Word Retrieval

The LSA model has been calculated from word descriptors where the visual and the textual information were available. However, in the corpus indexing and the retrieval phases only a single source of information is present. Only visual information is available when creating the corpus index. In this phase, the visual descriptor of each image word snipped $\mathbf{f}_i^v$ is calculated and projected into the transformed space by

$$\hat{\mathbf{f}}_i = \left[\mathbf{0}_t^{\top}, \mathbf{f}_i^{v\top}\right]\mathbf{X}_T,$$

where $\mathbf{0}_t$ is a zeros vector with the same dimensionality than the textual codebook. The projected vectors $\hat{\mathbf{f}}_i$ are the descriptors that are actually used to represent the word instances in the document.

In the query phase, an user types a textual query which is projected into the transformed space by

$$\hat{\mathbf{f}}^q = \left[\mathbf{f}_i^{t\top}, \mathbf{0}_v^{\top}\right]\mathbf{X}_T,$$

where $\mathbf{0}_v$ is a zeros vector which has the same dimensionality as the visual descriptor. Then, the cosine distance between the projected query $\hat{\mathbf{f}}^q$ and the projected visual descriptors $\hat{\mathbf{f}}_i$ is used as a similarity measure and generate a ranked list.

This procedure allows to retrieve word instances using textual queries from databases which have been described using only visual information. This is possible since the LSA algorithm has found relationships between the visual words and the n-grams in the training phase. Then, even if a source of information is not present in one of the descriptors, we are still able to rank and find relevant instances in the indexed documents.

V. Experimental Results

The proposed query-by-string word spotting method has been evaluated in the George Washington (GW) database [15], [16] consisting of 4864 segmented words. In order to train the LSA model that brings together the textual and visual information, we need a portion of the database to be

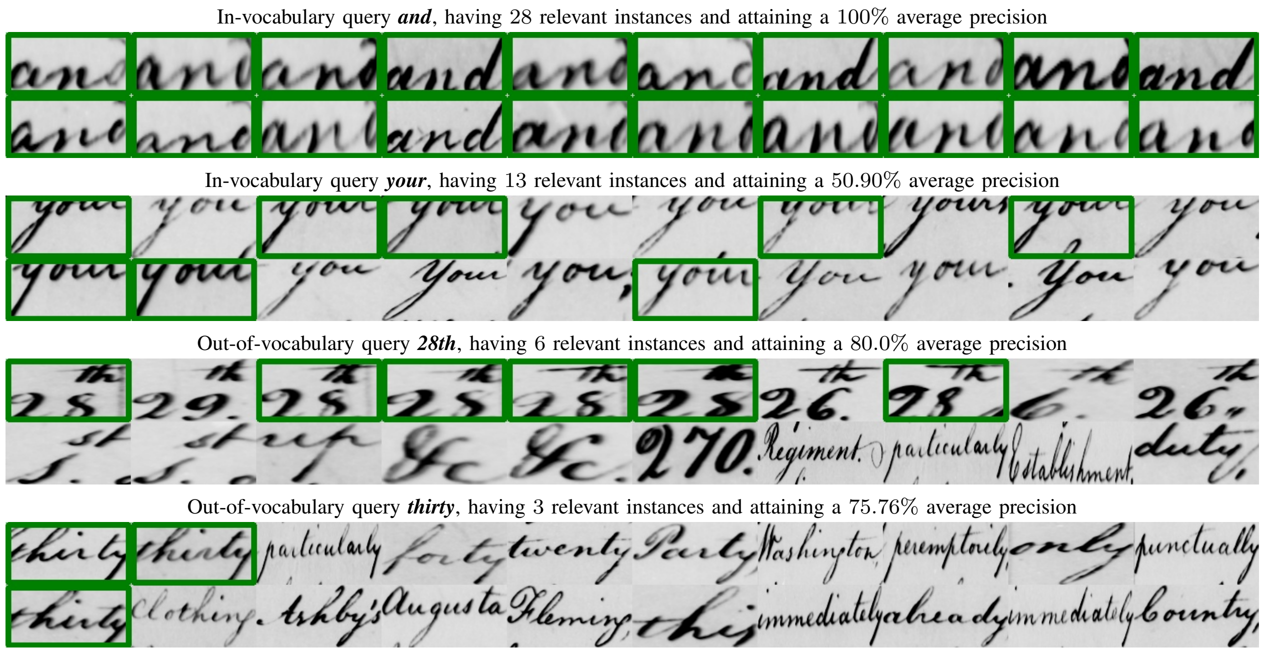
In-vocabulary query *and*, having 28 relevant instances and attaining a 100% average precision
In-vocabulary query *your*, having 13 relevant instances and attaining a 50.90% average precision
Out-of-vocabulary query *28th*, having 6 relevant instances and attaining a 80.0% average precision
Out-of-vocabulary query *thirty*, having 3 relevant instances and attaining a 75.76% average precision

Fig. 3. Examples of the 20 most similar word images returned by the system for queries.

TABLE I. STATE-OF-THE-ART QUERIED-BY-STRING WORD SPOTTING RESULTS FOR THE GW DATABASE.

Method	Segmentation	Cross-validation	Queries	mAP
Proposed	Word-level	4 folds	All words In-vocabulary words	56.54% 76.2%
Liang et al. [5]	Word-level	5 folds	38 queries	67% at rank 10
Fischer et al. [6]	Line-level	4 folds	In-vocabulary words	62.08%
Frinken et al. [7]	Line-level	4 folds	In-vocabulary words	71%

transcribed. Therefore, in order to evaluate the performance of the proposed query-by-string method, the database is divided into four different folds. Then, the system is trained using three of these folds and evaluated in the remaining one. At query time, all the words in the test set can be used. However, not all the words appearing in the test fold might be present in the train set. We will therefore report in our experiments two different measures, the performances reached when considering all 1829 words as queries and the performances reached when not considering out-of-vocabulary words, i.e. just casting the queries that are present in both train and test sets, that is 1090 queries.

A. Qualitative results

We present in Fig. 3 some qualitative results of the system. Framed in green appear the words that are considered relevant in our ground-truth. Note that in our experiments we have not filtered stop-words nor removed words with few appearances. We neither applied any stemming process, so for instance when querying the word ***your***, results as ***you*** or ***yours*** are accounted as negatives. We can also appreciate the performance difference when querying in and out vocabulary items.

B. Quantitative results

We report the obtained mean average previsions (mAP) results of our system in Fig. 4 depending on the amount of topics T.

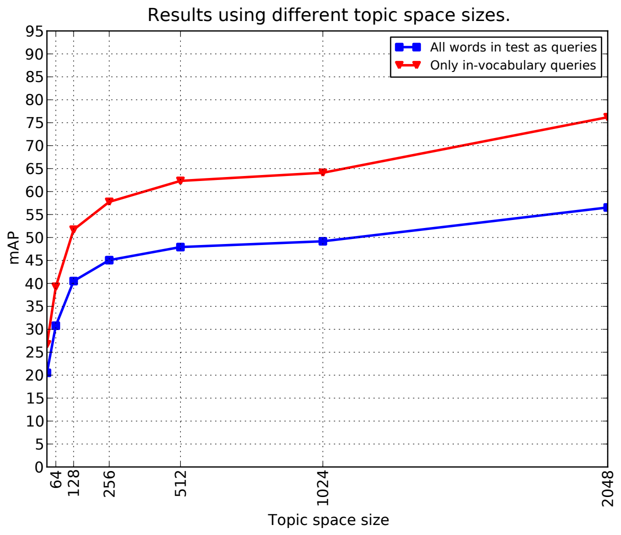

Fig. 4. mAP attained by the system using different number of dimensions in the topic space.

The resulting mAP shows that the larger the dimensionality of the topic space, the better the retrieval performance of the system. For the largest topic space size, we reached a 76.20% mAP when using only in-vocabulary queries. When considering all the queries in the test set, the performance drops to a 56.54% mAP. This is due to the out-of-vocabulary words. In the worst-case scenario in which just out-of-vocabulary terms are queried, the system yields a 27.08% mAP.

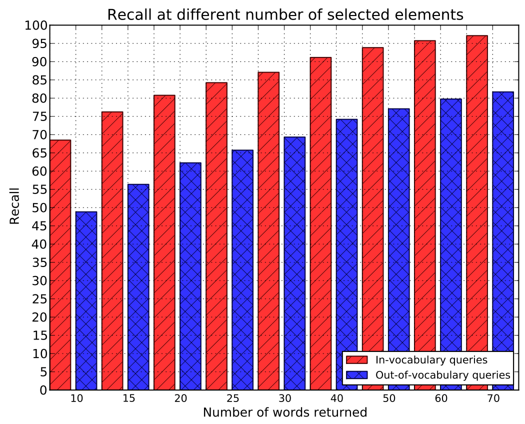

Fig. 5. Different recall values obtained while increasing the number of elements returned by the system.

It is worth noting that out-of-vocabulary words have a small number of instances and therefore are often penalized when computing the mAP measure if not ranked perfectly. Yet, the system still manages to retrieve them within the upper part of the rank. Therefore, we decided to report the recall at different ranks in Fig. 5. We can appreciate that even when just looking at the 10 first returned elements, we already achieve a 48.84% recall in the out-of-vocabulary scenario and a 68.5% recall in the in-vocabulary setup.

C. Comparison with the state of the art

Although it is not straightforward to provide a fair comparison between different methods, we report in Table I some state-of-the-art performances of query-by-string methods that also used the GW dataset. Both [6] and [7] use a very similar experimental setup like ours although they evaluate the average precisions at line level. In [5] Liang et al. used a selected set of queries to obtain the mAP just considering the first 10 elements in return. Using only these queries to evaluate the proposed method, our performance increases to a 83.12% mAP. Note as well that none of the considered methods can be easily extended to work in large-scale environments whereas the retrieval in our vector representation can be performed sub-linearly using standard indexing structures.

VI. Conclusion

In this paper, we have proposed a word spotting framework that follows the query-by-string paradigm. Textual and visual descriptors are merged together and projected to a common sub-vector space. This transform allows to cast queries represented in one modality while retrieving words that were just represented by the other modality. The proposed method has been evaluated in the GW dataset outperforming state-of-the-art performances.

The proposed vectorial representation can be efficiently encoded by means of standard indexation strategies such as random forests, locality-sensitive hashing structures or product quantizer codes. We also plan to further extend this current method in a segmentation-free paradigm as in [9], bypassing the word segmentation step.

Acknowledgment

This work has been partially supported by the Spanish Ministry of Education and Science under projects TIN2009-14633-C03-03, TIN2011-25606 and TIN2012-37475-C02-02.

References

[1] J. Edwards, Y. Teh, D. Forsyth, R. Bock, M. Maire, and G. Vesom, "Making latin manuscripts searchable using gHMM's," in *Advances in Neural Information Processing Systems*, 2004, pp. 385–392.

[2] S. Marinai, E. Marino, and G. Soda, "Font adaptive word indexing of modern printed documents," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 8, pp. 1187–1199, August 2006.

[3] T. Konidaris, B. Gatos, K. Ntzios, I. Pratikakis, S. Theodoridis, and S. Perantonis, "Keyword-guided word spotting in historical printed documents using synthetic data and user feedback," *International Journal on Document Analysis and Recognition*, vol. 9, no. 2–4, pp. 167–177, April 2007.

[4] Y. Leydier, A. Ouji, F. LeBourgeois, and H. Emptoz, "Towards an omnilingual word retrieval system for ancient manuscripts," *Pattern Recognition*, vol. 42, no. 9, pp. 2089–2105, September 2009.

[5] Y. Liang, M. Fairhurst, and R. Guest, "A synthesised word approach to word retrieval in handwritten documents," *Pattern Recognition*, vol. 45, no. 12, pp. 4225–4236, December 2012.

[6] A. Fischer, A. Keller, V. Frinken, and H. Bunke, "Lexicon-free handwritten word spotting using character HMMs," *Pattern Recognition Letters*, vol. 33, no. 7, pp. 934–942, May 2012.

[7] V. Frinken, A. Fischer, R. Manmatha, and H. Bunke, "A novel word spotting method based on recurrent neural networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 2, pp. 211–224, February 2012.

[8] J. Rodríguez-Serrano and F. Perronnin, "Synthesizing queries for handwritten word image retrieval," *Pattern Recognition*, vol. 45, no. 9, pp. 3270–3276, September 2012.

[9] M. Rusiñol, D. Aldavert, R. Toledo, and J. Lladós, "Browsing heterogeneous document collections by a segmentation-free word spotting method," in *Proceedings of the International Conference on Document Analysis and Recognition*, 2011, pp. 63–67.

[10] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong, "Locality-constrained linear coding for image classification," in *Proceedings of Computer Vision and Pattern Recognition*, 2010, pp. 3360–3367.

[11] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2006, pp. 2169–2178.

[12] F. Perronnin, J. Sánchez, and T. Mensink, "Improving the Fisher kernel for large-scale image classification," in *Proceedings of the European Conference on Computer Vision*, 2010, pp. 143–156.

[13] R. Cinbis, J. Verbeek, and C. Schmid, "Image categorization using Fisher kernels of non-iid image models," in *Proceedings of Computer Vision and Pattern Recognition*, 2012, pp. 2184–2191.

[14] S. Deerwester, S. Dumais, G. Furnas, T. Landauer, and R. Harshman, "Indexing by latent semantic analysis," *Journal of the American Society for Information Science and Technology*, vol. 41, no. 6, pp. 391–407, September 1990.

[15] V. Lavrenko, T. Rath, and R. Manmatha, "Holistic word recognition for handwritten historical documents," in *Proceedings of IEEE International Workshop on Document Image Analysis for Libraries*, 2004, pp. 278–287.

[16] T. Rath and R. Manmatha, "Word spotting for historical documents," *International Journal on Document Analysis and Recognition*, vol. 9, no. 2–4, pp. 139–152, April 2007.