Word Spotting and Recognition with Embedded Attributes

Jon Almazán, Albert Gordo, Alicia Fornés, Ernest Valveny

Abstract—This article addresses the problems of word spotting and word recognition on images. In word spotting, the goal is to find all instances of a query word in a dataset of images. In recognition, the goal is to recognize the content of the word image, usually aided by a dictionary or lexicon. We describe an approach in which both word images and text strings are embedded in a common vectorial subspace. This is achieved by a combination of label embedding and attributes learning, and a common subspace regression. In this subspace, images and strings that represent the same word are close together, allowing one to cast recognition and retrieval tasks as a nearest neighbor problem. Contrary to most other existing methods, our representation has a fixed length, is low dimensional, and is very fast to compute and, especially, to compare. We test our approach on four public datasets of both handwritten documents and natural images showing results comparable or better than the state-of-the-art on spotting and recognition tasks.

Index Terms—Word image representation, Attribute-based representation, Handwritten Text, Scene Text, Word Spotting, Word Recognition

1 Introduction

Text understanding in images is an important problem that has drawn a lot of attention from the computer vision community since its beginnings. Text understanding covers many applications and tasks, most of which originated decades ago due to the digitalization of large collections of documents. This made necessary the development of methods able to extract information from these document images: layout analysis, information flow, transcription and localization of words, etc. Recently, and motivated by the exponential increase of publicly available image databases and personal collections of pictures, this interest now also embraces text understanding on natural images. Methods able to retrieve images containing a given word or to recognize words in a picture have also become feasible and useful.

In this paper we consider two problems related to text understanding: word spotting and word recognition. In word spotting, the goal is to find all instances of a query word in a dataset of images. The query word may be a text string – in which case it is usually referred to as query by string (QBS) or query by text (QBT) –, or may also be an image, – in which case it is usually referred to as query by example (QBE). In word recognition, the goal is to obtain a transcription of the query word image. In many cases, including this work, it is assumed that a text dictionary or lexicon is supplied at test time, and that only words from that lexicon can be used as candidate transcriptions in the recognition task. In this work we will also assume that the location of the words in the images is provided, *i.e.*, we have access to images of cropped words. If those were not available, text localization and segmentation techniques [2], [3], [4], [5] could be used, but we consider that out of the scope of this work[1].

Traditionally, word spotting and recognition have focused on document images [6], [7], [8], [9], [10], [11], [12], [13], [14], where the main challenges come from differences in writing styles: the writing styles of different writers may be completely different for the same word. Recently, however, with the development of powerful computer vision techniques during the last decade, there has been an increased interest in performing word spotting and recognition on natural images [15], [16], [3], [4], [17], [5], which poses different challenges such as huge variations in illumination, point of view, typography, etc.

Word spotting can be seen as a particular case of semantic content based image retrieval (CBIR), where the classes are very fine-grained – we are interested in *exactly* one particular word, and a difference of only one character is considered a negative result – but also contain a very large intra-class variability – writing styles, illumination, typography, etc, can make the same word look very different. In the same way, word recognition can be seen as a special case

- *J. Almazán, A. Fornés and E. Valveny are with the Computer Vision Center, Dept. de Ciències de la Computació, Universitat Autònoma de Barcelona, Spain. E-mail: {almazan,afornes,ernest}@cvc.uab.es.*
- *A. Gordo is with the LEAR Team, INRIA, Grenoble, France. E-mail: albert.gordo@inria.fr.*
- *A preliminary version of this paper [1] appears in ICCV2013.*

1. One may argue that, when words are cropped, one is no longer performing word spotting but word ranking or word retrieval. However, word spotting is the commonly accepted term even when the word images are cropped, and we follow that convention in this work.

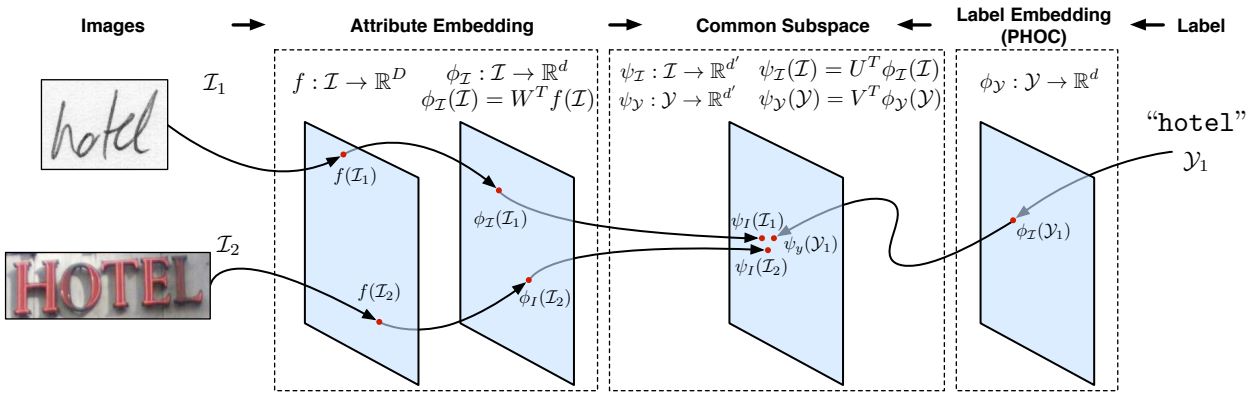


Figure 1: Overview of the proposed method. Images are first projected into an attributes space with the embedding function $\phi_{\mathcal{I}}$ after being encoded into a base feature representation with f. At the same time, labels strings such as "hotel" are embedded into a label space of the same dimensionality using the embedding function $\phi_{\mathcal{Y}}$. These two spaces, although similar, are not strictly comparable. Therefore, we project the embedded labels and attributes in a learned common subspace by minimizing a dissimilarity function $F(\mathcal{I}, \mathcal{Y}; U, V) = ||U^T \phi_{\mathcal{I}}(\mathcal{I}) - V^T \phi_{\mathcal{Y}}(\mathcal{Y})||_2^2 = ||\psi_{\mathcal{I}}(\mathcal{I}) - \psi_{\mathcal{Y}}(\mathcal{Y})||_2^2$. In this common subspace representations are comparable and labels and images that are relevant to each other are brought together.

of very fine-grained, zero-shot classification, where we are interested in classifying a word image into (potentially) hundreds of thousands of classes, for which we may not have seen any training example. The examples on Figs. 9 and 10 illustrate these issues.

In this work we propose to address the spotting and recognition tasks by learning a common representation for word images and text strings. Using this representation, spotting and recognition become simple nearest neighbor problems. We first propose a label embedding approach for text labels inspired by the bag of characters string kernels [18], [19] used for example in the machine learning and biocomputing communities. The proposed approach embeds text strings into a $d-$dimensional binary space. In a nutshell, this embedding –which we dubbed pyramidal histogram of characters or PHOC – encodes if a particular character appears in a particular spatial region of the string (*cf*. Fig 2). Then, this embedding is used as a source of character attributes: we will project word images into another $d-$dimensional space, more discriminative, where each dimension encodes how likely that word image contains a particular character in a particular region, in obvious parallelism with the PHOC descriptor. By learning character attributes independently, training data is better used (since the same training words are used to train several attributes) and out of vocabulary (OOV) spotting and recognition (*i.e.*, spotting and recognition at test time of words never observed during training) is straightforward. However, due to some differences (PHOCs are binary, while the attribute scores are not), direct comparison is not optimal and some calibration is needed. We finally propose to learn a low-dimensional common subspace with an associated metric between the PHOC embedding and the attributes embedding. The advantages of this are twofold. First, it makes direct comparison between word images and text strings meaningful. Second, attribute scores of images of the same word are brought together since they are guided by their shared PHOC representation. An overview of the method can be seen in Figure 1.

By having images and text strings share a common subspace with a defined metric, word spotting and recognition become a simple nearest neighbor problem in a low-dimensional space. We can perform QBE and QBS (or even a hybrid QBE+S, where both an image and its text label are provided as queries) using exactly the same retrieval framework. The recognition task simply becomes finding the nearest neighbor of the image word in a text dictionary embedded first into the PHOC space and then into the common subspace. Since we use compact vectors, compression and indexing techniques such as Product Quantization [20] could now be used to perform spotting in very large datasets. To the best of our knowledge, we

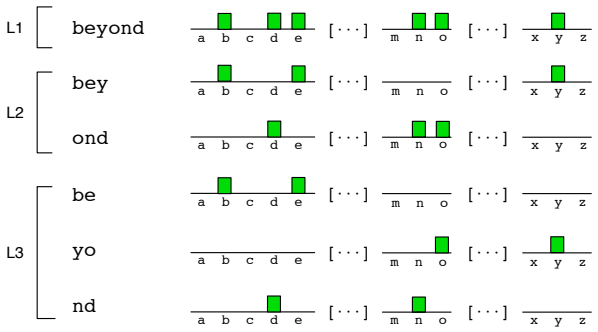


Figure 2: PHOC histogram of a word at levels 1, 2, and 3. The final PHOC histogram is the concatenation of these partial histograms.

are the first to provide a unified framework where we can perform out of vocabulary (OOV) QBE and QBS retrieval as well as word recognition using the same compact word representations.

The rest of the paper is organized as follows. In Section 2, we review the related work in word spotting and recognition. In Section 3 we describe how to encode our images into a low-dimensional attribute representation. In Section 4 we describe the proposed common subspace learning. Section 5 suggests some practices to learn the attributes space and the common subspace when training data is scarce. Section 6 deals with the experimental validation of our approach. Finally, Section 7 concludes the paper.

This paper is an extended version of the work initially published in ICCV 2013 [1]. Novel contributions include a better base feature representation tailored for word images (Section 3), a more detailed formulation of the common subspace problem (Section 4), a bagging approach to learn with scarce data (Section 5), and evaluation on recognition tasks, as well as new datasets, including two popular benchmarks based on natural images (Section 6).

2 Related Work

Here we review the works most related to some key aspects of our proposed approach.

2.1 Word Spotting and Recognition in Document Images

Word spotting in document images has attracted attention in the document analysis community during the last two decades [6], [7], [8], [9], [10], [11], [12], and still poses lots of challenges due to the difficulties of historical documents, different scripts, noise, handwritten documents, etc.

Because of this complexity, most popular techniques on document word spotting have been based on describing word images as sequences of features of variable length and using techniques such as Dynamic Time Warping (DTW) or Hidden Markov Models (HMM) to classify them. Variable-length features are more flexible than feature vectors and have been known to lead to superior results in difficult word-spotting tasks since they can adapt better to the different variations of style and word length [6], [7], [10], [11], [12], [14], [21].

Unfortunately, this leads to two unsatisfying outcomes. First, due to the difficulties of learning with sequences, many supervised methods cannot perform OOV spotting, *i.e.*, only a limited number of keywords, which need to be known at training time, can be used as queries. Second, because the methods deal with sequences of features, computing distances between words is usually slow at test time, usually quadratic with respect to the number of features. With efficient implementations they may be fast enough for some practical purposes (*e.g.*, making a search constrained in a particular book [14]), although dealing with very large volumes of data (*e.g.* millions of images) at testing time would be very inefficient.

Indeed, with the steady increase of datasets size there has been a renewed interest in compact, fast-to-compare word representations. Early examples of holistic representations are the works of Manmatha *et al*. [8] and Keaton *et al*. [22]. In [8], a distance between binary word images is defined based on the result of XORing the images. In [22], a set of features based on projections and profiles is extracted and used to compare the images. In both cases, the methods are limited to tiny datasets. A more recent work [23] exploits the Fisher kernel framework [24] to construct the Fisher vector of a HMM. This representation has a fixed length and can be used for efficient spotting tasks, although the paper focuses on only 10 different keywords. Finally, recent approaches that are not limited to keywords can be found in [25], [26], [27]. Gatos *et al*. [25] perform a template matching of block-based image descriptors, Rusiñol *et al*. [26] use an aggregation of SIFT descriptors into a bag of visual words to describe images, while Almazán *et al*. [27] use HOG descriptors [28] combined with an exemplar-SVM framework [29]. These fast-to-compare representations allow them to perform word spotting using a sliding window over the whole document without segmenting it into individual words. Although the results on simple datasets are encouraging, the authors argue that these fixed-length descriptors do not offer enough flexibility to perform well on more complex datasets and especially in a multi-writer scenario.

Through this paper we follow these recent works [27], [26] and focus on fixed-length representations, which are faster to compare and store and can be used in large-scale scenarios. Our proposed approach based on attributes directly addresses the aforementioned problems: our attributes framework very naturally deals with OOV query words at test time, while producing discriminative, compact signatures that are fast to compute, compare, and store.

Regarding word recognition, handwritten recognition still poses an important challenge for the same reasons. As in word spotting, a popular approach is to train HMMs based on grapheme probabilities [13]. A model is first trained using labeled training data. At test time, given an image word and a text word, the model computes the probability of that text word being produced by the model when fed with the image word. Recognition can then be addressed by computing the probabilities of all the lexicon words given the query image and retrieving the nearest neighbor. As in the word spotting case, the main drawback here is the comparison speed, since computing these probabilities is orders of magnitude slower than computing an Euclidean distance or a dot

product between vectorial representations.

2.2 Word Spotting and Recognition in Natural Images

The increasing interest in extracting textual information from real scenes is related to the recent growth of image databases such as Google Images or Flickr. Some interesting tasks have been recently proposed, *e.g*. localization and recognition of text in Google Street View images [15] or recognition in signs harvested from Google Images [30]. The high complexity of these images when compared to documents, mainly due the the large appearance variability, makes it very difficult to apply traditional techniques of the document analysis field. However, with the recent development of powerful computer vision techniques some new approaches have been proposed.

Some methods have focused on the problem of end-to-end word recognition, which comprises the tasks of text localization and recognition. Wang *et al*. [15] address this problem by combining techniques commonly applied in object recognition, such as Random Ferns and Pictorial Structures. They first detect a set of possible character candidates windows using a sliding window approach and then each word in the lexicon is matched to these detections. Finally, the one with the highest score is reported as the predicted word. Neumann and Matas [3], [4] also address this problem of end-to-end word recognition. In [3] they pose the character detection problem as a sequential selection from the set of Extremal Regions. Then, the recognition of candidate regions is done in a separate OCR stage using synthetic fonts. In [4] they introduce a novel approach for character detection and recognition where they first detect candidate characters as image regions which contain strokes of specific orientations and then, these characters are recognized using specific models trained for each character and grouped into lines to form words. Bissacco *et al*. [5] take advantage of recent progress in machine learning, concretely in deep neural networks, and large scale language modeling. They first perform a text detection process returning candidate regions containing individual lines of text, which are then processed for text recognition. This recognition is done by identifying candidate character regions and maximizing a score that combines the character classifier and language model likelihoods. Although they use the lexicon information in a post-process to correct some recognition errors, one important advantage of this method is that it does not requiere an available lexicon for a full recognition of the image words.

Different problems are also explored by Mishra *et al*. [30], [16], [31]. In [30] they focus only on the problem of recognition and present a framework that uses the n-gram information in the language by combining these priors into a higher order potential function in a Conditional Random Field model defined on the image. In [31] they propose a method to perform image retrieval using textual cues. Instead of relying on a perfect localization and recognition to retrieve images containing a given text query, they propose a query-driven search: they find approximate locations of the characters in the text query, and then impose spatial constrains. By contrast, [16] address this problem from a different point of view. Rather than pre-selecting a set of character detections, they define a global model that incorporates language priors and all potential characters. They present a framework that exploits bottom-up cues, derived from Conditional Random Field models on individual character detections, and top-down cues, obtained from a lexicon-based prior. Goel *et al*. [32] address the problem of recognition as a retrieval framework: lexicon words are transformed in a collection of synthetic images and the recognition is posed as retrieving the best match from the lexicon image set. They use gradient-based features to represent the images and a weighted Dynamic Time Warping to perform the matching.

In general, the main structure of these methods consists of a first step of probabilistic detection of character candidate regions in the image, and a second step of recognition using character models and grouping constrains. This leads to models tailored for recognition, but with a limited usability for other tasks such as comparing two word images (for QBE), or storing word images using a compact representation for indexing purposes. Our model, by contrast, addresses these issues in a natural way and is useful both for recognition and retrieval tasks.

2.3 Zero-Shot Learning and Label Embedding

To learn how to retrieve and recognize words that have not been seen during training, it is necessary to be able to transfer knowledge between the training and testing samples. One of the most popular approaches to perform this zero-shot learning in computer vision involves the use of visual attributes [33], [34], [35], [36]. In this work we use character attributes to transfer the information between training and testing samples. Although the idea of separating words into characters and learning at the character level has been used before (see, *e.g*., the character HMM models of [6], [37]), these approaches have been tied to particular HMM models with sequence features, and so their performance has been bounded by them. In our case, we propose a broader framework since we do not constrain the choice of features or the method to learn the attributes.

Our work can also be related to label embedding methods [38], [39], [17], where labels are embedded into a different space and a compatibility function between images and labels is defined. Of those, the work of Rodriguez and Perronnin [17] is the most

related to our work, since it also deals with text recognition and presents a text embedding approach (spatial pyramid of characters or SPOC) very similar to ours[2]. The main difference stems from how the embedding is used. While in our case, we use it as a source of attributes, and only then we try to find a common subspace between the attributes and the PHOCs, Rodriguez and Perronnin try to find a common subspace directly between their image representation (Fisher vectors [40]) and their SPOCs using a structured SVM framework. Our approach can be seen as a more regularized version of theirs, since we enforce that the projection that embeds our images into the common subspace can be decomposed into a matrix that projects the images into an attributes space.

3 Attributes Based Word Representation

In this section we describe how we obtain the attributes based-representation of a word image. We start by motivating our pyramidal histogram of characters (PHOC) representation, which embeds label strings into a $d-$dimensional space. We then show how to use this PHOC representation to encode word images.

One of the most popular approaches to perform supervised learning for word spotting is to learn models for particular *keywords*. A pool of positive and negative samples is available for each keyword, and a model (usually a HMM) is learned for each of them. At test time, it is possible to compute the probability of a given word being generated by that keyword model, and that can be used as a score. Note that this approach restricts one to keywords that need to be learned offline, usually with large amounts of data. In [12], this problem is addressed by learning a semicontinuous HMM (SC-HMM). The parameters of the SC-HMM are learned on a pool of unsupervised samples. Then, given a query, this SC-HMM model can be adapted, online, to represent the query. This method is not restricted to keywords and can perform OOV spotting. However, the labels of the training words were not used during training.

One disadvantage of these approaches that learn at the word level is that information is not shared between similar words. For example, if learning an HMM for a "car" keyword, "cat" would be considered a negative sample, and the shared information between them would not be explicitly used. We believe that sharing information between words is extremely important to learn good discriminative representations, and that the use of attributes is one way to achieve this goal. Attributes are semantic properties that can be used to describe images and categories [34], and have recently gained a lot of popularity for image retrieval and classification tasks [34], [33], [41], [42], [36]. Attributes have also shown ability to transfer information in zero-shot learning settings [33], [34], [35], [36] and have been used for feature compression since they usually provide compact descriptors. These properties make them particularly suited for our word representation task, since they can transfer information from different training words and lead to compact signatures. The selection of these attributes is commonly a task-dependent process, so for their application to word spotting we should define them as word-discriminative and appearance-independent properties. In the following subsection we describe our label embedding approach, which embeds text strings into a binary vectorial space, and later we will show how to use this embedding as a source of word-discriminative visual attributes.

3.1 Text Label Embedding with PHOCs

One straightforward approach to embed text strings is to construct a (binary) histogram of characters. When using digits plus the English alphabet, this leads to a histogram of 36 dimensions[3], where each dimension represents whether the text string contains a particular character or not. In an attributes context, these can be understood as the labels for attributes defined as "word contains an x" or "word contains a y".

However, this label embedding is not word-discriminative: words such as "listen" and "silent" share the same representation. Therefore, we propose to use a pyramid version of this histogram of characters, which we dubbed PHOC (see Fig. 2). Instead of finding characters on the whole word, we focus on different regions of the word. At level 2, we define attributes such as "word contains character x on the first half of the word" and "word contains character x on the second half of the word". Level 3 splits the word in 3 parts, level 4 in 4, etc. In practice, we use levels 2, 3, 4, and 5, leading to a histogram of $(2 + 3 + 4 + 5) \times 36 = 504$ dimensions. Finally, we also add the 50 most common English bigrams at level 2, leading to 100 extra dimensions for a total of 604 dimensions. These bigrams let us encode relations between adjacent characters, which may help to disambiguate when finding a low-dimensional common subspace (*cf*. Section 4). In this case, when using a pyramidal encoding and bigrams, "listen" and "silent" have significantly different representations.

Given a transcription of a word we need to determine the regions of the pyramid where we assign each character. For that, we first define the normalized occupancy of the k-th character of a word of length n

2. Both the conference version of this paper [1] and the work of Rodriguez and Perronnin [17] appeared simultaneously.

3. We do not make any distinction between lower-case and upper-case letters, which leads to a case-insensitive representation. It is trivial to modify it to be case-sensitive, at the cost of adding another 26 attributes. It is also straightforward to include other punctuation marks.

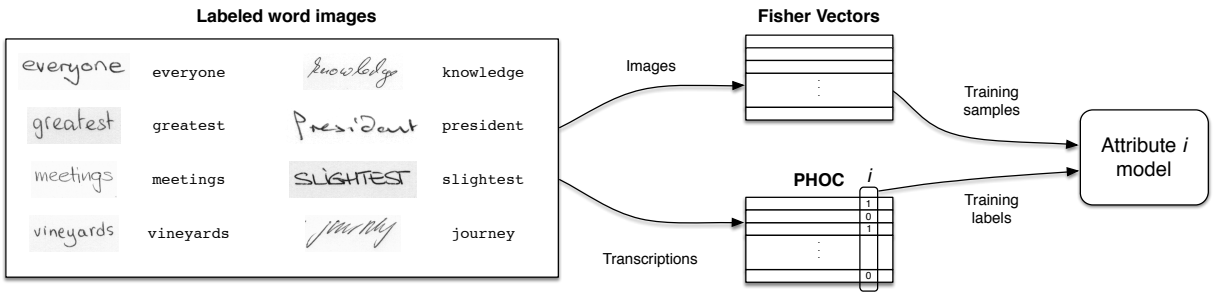


Figure 3: Training process for i-th attribute model. An SVM classifier is trained using the Fisher vector representation of the images and the i-th value of the PHOC representation as label.

as the interval $Occ(k,n) = [\frac{k}{n}, \frac{k+1}{n}]$, where the position k is zero-based. Note that this information is extracted from the word transcription, not from the word image. We remark that we do not have access to the exact position of the characters on the images at training time, only their transcription is available. We use the same formula to obtain the occupancy of region r at level l. Then, we assign a character to a region if the overlap area between their occupancies is larger or equal than 50% the occupancy area of the character, *i.e.*, if $\frac{|Occ(k,n) \cap Occ(r,l)|}{|Occ(k,n)|} \geq 0.5$, where $|[a,b]| = b - a$. This is trivially extended to bigrams or trigrams.

3.2 Learning Attributes with PHOCs

As we mentioned, the PHOC histograms can be seen as labels of attributes asking questions such as "word contains character x on the first half of the word" or "word contains character y on the second half of the word". These attributes are word-discriminative, since they are based on the word-discriminative PHOC embedding. If the attributes are learned using data coming from different writers or sources, the resulting models will also be robust to changes in appearance, style, etc.

To learn these attributes we use linear SVMs. Word images are first encoded into feature vectors, and these feature vectors are used together with the PHOC labels to learn SVM-based attribute models. The approach is illustrated in Figure 3. To represent the images, we use Fisher vectors (FV) [40], a state-of-the-art encoding method [43] which works well with linear classifiers. The FV can be seen as a bag of visual words [44] that encodes not only word counts but also higher-order statistics. In a nutshell, at training time, low-level descriptors (SIFTs in our case) are extracted from the training images and used to learn a Gaussian mixture model (GMM) $\lambda = \{w_k, \mu_k, \Sigma_k, k = 1 \dots K\}$, where w are the mixing weights, μ are the means, Σ the (diagonal) covariances, and K is the number of Gaussians. Then, to compute the representation of one image, one densely extracts its low-leveldescriptors and aggregates the gradients of the GMM model with respect to its parameters (usually only means and variances are considered, since weights add little extra information) evaluated at those points. This leads to a highly discriminative, high-dimensional signature. We note however that there is absolutely no requirement to use Fisher vectors or SVMs. Any encoding method and classification algorithm that transforms the input image into attribute scores could be used to replace them. We chose SVMs and FVs for their simplicity and effectivity.

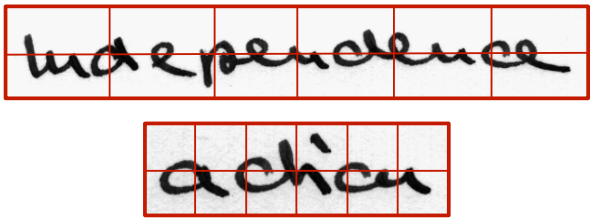


Figure 4: Spatial pyramids on word images. The sizes and contents of each spatial region are very dependent on the length of the word.

3.3 Adding Spatial Information

One problem with many image encoding methods, including the FV, is that they do not explicitly encode the position of the features, which is extremely important to describe word images. If the spatially-aware attributes allow one to *ask* more precise questions about the location of the characters, spatial information on the image representation is needed to be able to correctly *answer* those questions.

One well-established approach to add spatial information is to use spatial pyramids [45]. Instead of aggregating the descriptors of the whole image, the image is first divided in k regions, and the features of each region are aggregated independently. This produces k independent descriptors that are concatenated to produce the final descriptor. When dealing with word images, however, this poses a problem: words of different lengths will produce regions of very different sizes and contents, see Fig. 4.

A different approach that works well in combination with FVs was proposed by Sánchez et al [46].

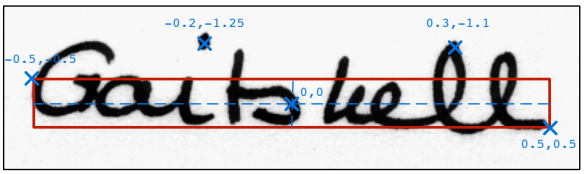


Figure 5: Word image and the automatically adjusted reference box that defines the coordinates system.

In a nutshell, the SIFT descriptors of the image are enriched by appending the normalized x and y coordinates and the scale they were extracted at. Then, the GMM is learned not on the original SIFT features but on these enriched ones. When computing the FV using these features and GMM, the representation implicitly encodes the position of the features inside the word. They showed that, for natural images, this achieved results comparable to spatial pyramids with much lower-dimensional descriptors. When dealing with natural images, the x and y coordinates were normalized between -0.5 and 0.5. In the case of word images we follow the same approach. However, cropping differences during annotation lead to changes of the word position inside the image, making these coordinates less robust. Because of this, instead of using the whole word image as a coordinate system, we automatically and approximately find the begining and end, as well as the baseline and the median line of the word (by greedily finding the smallest region that contains 95% of the density of the binarized image) and use that for our reference coordinates system. The center of that box corresponds to the origin, and the limits of the box are at $[-0.5, 0.5]$. Pixels outside of that reference box are still used with their corresponding coordinates. See Figure 5 for an illustration.

Either when using spatial pyramids or xy enriching, the GMM vocabulary is learned using the whole image. We propose to improve these representations by learning region-specific GMMs. At training time, we split the images in regions similar to spatial pyramids, and learn an independent, specialized vocabulary on each region. These GMMs are then merged together and their weights are renormalized to sum 1.

We evaluated these different representations on the IAM dataset (cf. Section 6 for more details regarding the dataset). The goal here is to find which representation leads to better results at predicting the attributes at the right location; correctly predicting the attributes is of paramount importance, since it is deeply correlated with the final performance at retrieval and recognition tasks. We used the training set of IAM to learn the attributes, and then evaluate the average precision of each attribute on the test set and report the mean average precision. Figure 6 shows the results. It is clear that some type of spatial information is needed, either xy enriching or spatial pyramid. The specialized GMMs do not work well

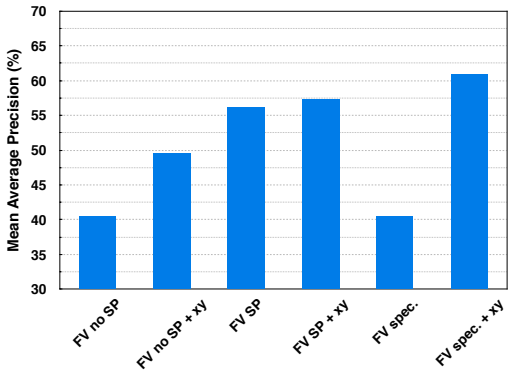


Figure 6: Results of the attributes classifiers for different Fisher vector configurations on the IAM dataset.

when used directly, which is not surprising, since the distribution of characters in words is in general (close to) uniform, and so the specialized GMMs are actually very similar. However, when learning specialized GMMs on enriched SIFT features, the coordinates add some information about the position that the specialized GMM is able to exploit independently on each region. The final result is that the specialized GMMs on enriched SIFTs lead to the best performance, and is the representation that we will use through the rest of our experiments.

4 Attributes and Labels Common Subspace

Through the previous section we presented an attributes-based representation of the word images. Although this representation is robust to appearance changes, special care has to be put when comparing different words, since the scores of one attribute may dominate over the scores of other attributes. Directly comparing embedded attributes and embedded text labels is also not well defined: although both lie in a similar space of the same dimensionality, the embedded text labels are binary, while the attribute scores are not and have different ranges. Even if directly comparing those representations yields reasonable results due to their similarities, such direct comparison is not well principled. Therefore, some calibration of the attribute scores and PHOCs is necessary.

One popular approach to calibrate SVM scores is Platts scaling. It consists of fitting a sigmoid over the output scores to obtain calibrated probabilities, $P(y = 1|s) = (1 + exp(\alpha s + \beta))^{-1}$, where α and β can be estimated using MLE. In the recent [47], Extreme Value Theory is used to fit better probabilities to the scores and to find a multi-attribute space similarity. After the calibration, all scores are in the range $[0 - 1]$, which makes them more comparable between themselves –useful for the QBE task–, as well as more comparable to the binary PHOC representation – useful for the QBS and recognition tasks.

One disadvantage of such approaches is that they do not take into account the correlation between the different attributes. In our case this is particularly important, since our attributes are very correlated due to multilevel encoding and the bigrams. Here we propose to perform the calibration of the scores jointly, since this can better exploit the correlation between different attributes. To achieve this goal, we first propose to address it as a ridge regression problem. However, this only takes into account the correlation between the attribute scores, and ignores the correlations between the attributes themselves. Therefore, we also propose a common subspace regression (CSR) that leads to a formulation equivalent to Canonical Correlation Analysis.

Let $\mathcal{I} = \{\mathcal{I}_n, n = 1, \ldots, N\}$ be a set of N images available for training purposes, and let $\mathcal{Y} = \{\mathcal{Y}_n, n = 1, \ldots, N\}$ be their associated labels. Let also $A = \phi_{\mathcal{I}}(\mathcal{I}) \in \Re^{d \times N}$ be the N images embedded in the $d-$dimensional attribute space, and let $B = \phi_{\mathcal{Y}}(\mathcal{Y}) \in \{0,1\}^{d \times N}$ be the N labels embedded in the $d-$dimensional label space. Then, one straightforward way to relate the attribute scores of A to the embedded labels of B is to define a distance function $F(\mathcal{I}_i, \mathcal{Y}_i; P) = ||P^T \phi_{\mathcal{I}}(\mathcal{I}_i) - \phi_{\mathcal{Y}}(\mathcal{Y}_i)||_2^2$, with $P \in \Re^{d \times d}$, and to minimize the distance across all the samples and their labels,

$$\begin{aligned} &\underset{P}{\operatorname{argmin}} \sum_i^N \frac{1}{2} F(\mathcal{I}_i, \mathcal{Y}_i; P) + \frac{1}{2}\Omega(P) \quad = \\ &\underset{P}{\operatorname{argmin}} \frac{1}{2}||P^T A - B||_F^2 + \frac{1}{2}\Omega(P), \end{aligned} \tag{1}$$

and where $\Omega(P) = \alpha||P||_F^2$ is a regularization term and α controls the weight of the regularization. In this case this is equivalent to a ridge regression problem and P has a closed form solution $P = (AA^T + \alpha I)^{-1} AB^T$, where I is the identity matrix. Since d is the number of attributes, which is low, solving this problem (which needs to be solved only once, at training time) is extremely fast.

As we mentioned, however, this formulation only exploits the correlation of the attribute scores and ignores the correlations between the attributes themselves. We therefore modify it to project both views into a common subspace of dimensionality d' (see Fig. 7). We define a new distance function $\hat{F}(\mathcal{I}_i, \mathcal{Y}_i; U, V) = ||\psi_{\mathcal{I}}(\mathcal{I}_i) - \psi_{\mathcal{Y}}(\mathcal{Y}_i)||_2^2$, with $\psi_{\mathcal{I}}(\mathcal{I}) = U^T \phi_{\mathcal{I}}(\mathcal{I})$ and $\psi_{\mathcal{Y}}(\mathcal{Y}) = V^T \phi_{\mathcal{Y}}(\mathcal{Y})$ being two linear embedding functions that use projection matrices $U, V \in \Re^{d \times d'}$ to embed $\phi_{\mathcal{I}}(\mathcal{I})$ and $\phi_{\mathcal{Y}}(\mathcal{Y})$ into a common subspace. Then, analogous to the previous case, the goal is to minimize the distance across all the samples and their labels,

$$\begin{aligned} &\underset{U,V}{\operatorname{argmin}} \sum_i^N \frac{1}{2} \hat{F}(\mathcal{I}_i, \mathcal{Y}_i; U, V) + \frac{1}{2}\Omega(U) + \frac{1}{2}\Omega(V) \quad = \\ &\underset{U,V}{\operatorname{argmin}} \frac{1}{2}||U^T A - V^T B||_F^2 + \frac{1}{2}\Omega(U) + \frac{1}{2}\Omega(V), \\ &\quad \text{s.t.} \\ &\quad\quad \psi_{\mathcal{I}}(\mathcal{I})\psi_{\mathcal{I}}(\mathcal{I})^T = I \\ &\quad\quad \psi_{\mathcal{Y}}(\mathcal{Y})\psi_{\mathcal{Y}}(\mathcal{Y})^T = I, \end{aligned} \tag{2}$$

where the orthogonality constrains ensure that the solutions found are not trivial.

By using lagrangian multipliers, taking derivatives with respect to U and V, and making them equal to zero, one arrives to the following equalities:

$$\begin{aligned} \lambda(AA^T + \alpha I)u_k &= AB^T v_k \\ \lambda(BB^T + \alpha I)v_k &= BA^T u_k, \end{aligned} \tag{3}$$

where u_k and v_k are the $k-$th columns of matrices U and V, and λ appears due to the lagrangian multipliers. When solving for u_k, one arrives to the following generalized eigenvalue problem:

$$AB^T(BB^T + \alpha I)^{-1} BA^T u_k = \lambda^2 (AA^T + \alpha I)u_k \tag{4}$$

The first k generalized eigenvectors form the first k columns of the projection matrix U. This allows one to choose the final dimensionality d'. An analogous process can be used to obtain the projection matrix V. We can observe how, in this case, we explicitly use more relations between the data than in the regression case, which leads to better models. This model also allows one to control the output dimensionality and perform dimensionality reduction. In some of our experiments we will reduce the final dimensionality of our representations down to 80 dimensions while still obtaining state-of-the-art results. As in the regression case, the matrices U and V are very fast to obtain since they depend on the dimensionality of the attribute space, which is low.

Interestingly, these equations are also the solution to the Canonical Correlation Analysis (CCA) problem, where one tries to find the projections that maximize the correlation in a common subspace [48]. CCA is a tool to exploit information available from different data sources, used for example in retrieval [49] and clustering [50]. In [51], CCA was used to correlate image descriptors and their labels, which brought significant benefits for retrieval tasks. We believe this is the most similar use of CCA to our approach. However, while [51] combined images and labels with the hope of bringing some semantic consistency to the image representations, our goal here is to bring the imperfect predicted scores closer to their perfect value in a common subspace.

The optimization shown in Equation (2) aims at minimizing the distance between the images and

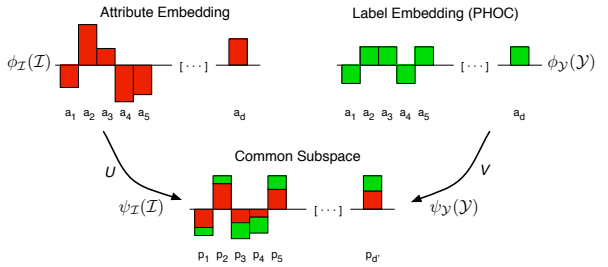


Figure 7: Projection of predicted attribute scores and attributes ground truth into a more correlated subspace with CSR.

their corresponding labels, but makes no effort in pushing apart negative labels and learning to rank. It is inviting to, instead, learn the parameters of $\hat{F}$ that optimize the ranking loss directly, as done for example in Weston *et al*. [52]. However, we found that the results of the CSR were similar or superior to those obtained optimizing the ranking loss directly. We believe this is due to the non-convexity of the optimization problem. Interestingly, similar results were obtained in the text embedding method of [17], where the structured SVM approach used to optimize their (similar) compatibility function barely improved over the initialization of the parameters based on regression.

One may also note that the relation between the attribute scores and the binary attributes may not be linear, and that a kernelized approach (KCSR) could yield larger improvements. In this case, we follow the approach of [51]: we explicitly embed the data using a random Fourier feature (RFF) mapping [53], so that the dot-product in the embedded space approximately corresponds to a Gaussian kernel $K(x, y) = exp(-\gamma||x - y||^2)$ in the original space, and then perform linear projection on the embedded space. In this case, at testing time, a sample is first projected into the attribute space, then embedded using the RFF mapping, and finally projected into the common subspace using the learned projections.

5 Learning with Scarce Data

One inconvenience of learning the attribute space and the common subspace in two independent steps is the need of sufficiently large amounts of training data. This is because the data used to learn the common subspace should be different than the data used to learn the attribute space. The reason is that, if we embed the same data used to train the attributes into the attributes space, the scores of the SVMs will be severely overfit (most of them will be very close to -1 or 1), and therefore the common subspace learned using that data will be extremely biased, leading to inferior results. If enough training data is available, one can construct two disjoint training sets, train the attribute on one of the sets, and train the common subspace using the other set. However this does not fully exploit the training data, since each training sample is used only to learn the attributes or the common subspace, but not both.

To overcome this problem, we propose to use a variant of bagging. The training data is split in several folds of training and validation partitions. The training and validation data of each fold is disjoint, but different folds will have overlapping data. In each fold, a model is learned using the training data of that fold, and this model is used to score the validation data. Therefore, the scores on the validation data are (almost) unbiased. Through several folds, the validation scores are added, and, for each sample, a counter that indicates how many times it has been scored is kept. At the end of the process, a global model is produced by averaging all the local models. By normalizing the score of every sample by the number of times it was scored, we also produce unbiased scores of the train set, which can be used to learn the common subspace without problems. The process to learn the model of one attribute and score the training set is depicted in Algorithm 1 using a Matlab-like notation. Note that some care needs to be taken to ensure that all training samples appear at least once in the validation set so they can be scored.

6 Experiments

We start by describing the datasets we use through our experiments. We then describe the most relevant implementation details of our approach. After that, we present our results and compare them with the published state-of-the-art.

6.1 Datasets:

We evaluate our method in four public datasets: two datasets of handwritten text documents, and two datasets of text in natural scenes.

The IAM off-line dataset [4] [54] is a large dataset comprised of $1,539$ pages of modern handwritten English text written by 657 different writers. The document images are annotated at word and line level and contain the transcriptions of more than $13,000$ lines and $115,000$ words. There also exists an official partition for *writer independent text line recognition* that splits the pages in three different sets: a training set containing $6,161$ lines, a validation set containing $1,840$ lines and a test set containing $1,861$ lines. These sets are writer independent, *i.e.*, each writer contributed to one and only one set. Thorough our spotting and recognition experiments we will use this official partition, since it is the one most widely used and eases comparison with other methods.

4. http://www.iam.unibe.ch/fki/databases/iam-handwriting-database

Algorithm 1 Learn attribute model with bagging

Input: Training data $X \in \Re^{D \times N}$
Input: Training labels $Y \in \{0,1\}^N$
Input: Number of folds F
Output: Model $W \in \Re^D$
Output: Training data embedded onto the attribute space $A \in \Re^N$
$W = \text{zeros}(1, D)$
$A = \text{zeros}(1, N)$
$count = \text{zeros}(1, N)$
$f = 1$
while $f \leq F$ **do**
Split data in train and val partitions
$TrainIdx, ValIdx = \text{split}(N, f)$
$TrainData = X(:, TrainIdx)$
$TrainLabels = Y(TrainIdx)$
$ValData = X(:, ValIdx)$
$ValLabels = Y(ValIdx)$
Learn model using training data. Use validation set to validate the parameters.
$W_f = \text{learnSVM}(TrainData, TrainLabels, ValData, ValLabels)$
Encode the validation set into the attributes space and keep track of the number of updates
$A(ValIdx) = A(ValIdx) + W_f^T ValData$
$count(ValIdx) = count(ValIdx) + 1$
Add W_f **to the global model** W
$W = W + W_f$
$f = f + 1$
end while
Normalize and end
$W = W/F$
$A = A/count$
End

The George Washington (GW) dataset[5] [10] contains 20 pages of letters written by George Washington and his associates in $1,755$. The writing styles present only small variations and it can be considered a single-writer dataset. The dataset contain approximately $5,000$ words annotated at word level. There is no official partition for the GW dataset. We follow the approach of [7], [55] and split the GW dataset in two sets at word level containing 75% and 25% of the words. The first set is used to learn the attributes representation and the calibration, as well as for validation purposes, and the second set is used for testing purposes. The experiments are repeated 4 times with different train and test partitions and the results are averaged.

The IIIT 5K-word (IIIT5K) dataset[6] [30] contains $5,000$ cropped word images from scene texts and born-digital images, obtained from Google Image engine search. This is the largest dataset for natural image word spotting and recognition currently available. The official partition of the dataset contains two subsets of $2,000$ and $3,000$ images for training and testing purposes. These are the partitions we use in our experiments. The dataset also provides a global lexicon of more than half million dictionary words that can be used for word recognition. Each word is associated with two lexicon subsets: one of 50 words, and one of $1,000$ words.

The Street View Text (SVT) dataset[7] [15] is comprised of images harvested from Google Street View where text from business signs and names appear. It contains more than 900 words annotated in 350 different images. In our experiments we use the official partition that splits the images in a train set of 257 word images and a test set of 647 word images. This dataset also provides a lexicon of 50 words per image for recognition purposes.

6.2 Implementation Details

We use Fisher vectors [40] as our base image representation. SIFT features are densely extracted at 6 different patch sizes (bin sizes of 2, 4, 6, 8, 10, and 12 pixels) from the images and reduced to 62 dimensions with PCA. Then, the normalized x and y coordinates are appended to the projected SIFT descriptors. To normalize the coordinates, we use the automatically detected reference boxes on the IAM and GW datasets. On IIIT5K and SVT, we observed that the minibox approach did not perform as well due to the nature of the backgrounds, which difficults the fitting of the bounding box, so we use the whole image as reference system. These features are then aggregated into a FV that considers the gradients with respect to the means and variances of the GMM generative model.

To learn the GMM we use 1 million SIFT features extracted from words from the training sets. We use 16 Gaussians per GMM, and learn the GMM in a structured manner using a 2×6 grid leading to a GMM of 192 Gaussians. This produces histograms of $2 \times 64 \times 192 = 24,576$ dimensions. For efficiency reasons, however, on the IAM dataset, we use SIFT features reduced to 30 dimensions instead of 62. This produces histograms of $12,288$ dimensions. This reduction improved the training speed and storage costs while barely affecting the final results. The descriptors are then power- and L2- normalized. Please *cf*. [56] for more details and best practices regarding the construction of FV representations.

When computing the attribute representation, we use levels 2, 3, 4 and 5, as well as 50 common bigrams at level 2, leading to 604 dimensions when considering digits plus the 26 characters of the English alphabet.

5. http://www.iam.unibe.ch/fki/databases/iam-historical-document-database/washington-database
6. http://cvit.iiit.ac.in/projects/SceneTextUnderstanding/IIIT5K.hmtl
7. http://vision.ucsd.edu/~kai/svt/

Table 1: Retrieval results on the IAM, GW, IIIT5K and SVT datasets. Accuracy measured in mean average precision.

	IAM		GW		IIIT5K		SVT	
	QBE	QBS	QBE	QBS	QBE	QBS	QBE	QBS
FV	15.66	–	62.72	–	24.21	–	22.88	–
Att.	44.60	39.25	89.85	67.64	55.71	35.91	48.94	60.32
Att. + Platts	48.09	66.86	**93.04**	**91.29**	62.05	62.30	51.47	76.01
Att. + Reg.	46.59	60.95	90.54	87.02	61.06	58.12	52.51	71.88
Att. + CSR.	52.61	73.54	92.46	90.81	**63.79**	**66.24**	**55.86**	**79.65**
Att. + KCSR.	**55.73**	**73.72**	92.90	91.11	63.42	65.15	**55.87**	79.35

We learn the attributes using the bagging approach of Algorithm 1 with 10 folds on all datasets. Since the training set of SVT is very small (only 257 images), we augment it by using the whole 5K dataset as training set. Note that other recent works that evaluate on SVT also augment the data, either producing synthetic training [32] or by using in-house datasets [5].

When learning and projecting with CSR and KCSR, the representations (both score attributes and embedded labels) are first L2-normalized and mean centered. We use CSR to project to a subspace of 80 dimensions on all datasets. For KCSR, we project into 160 dimensions on all datasets. Then, once projected, the representations are L2 normalized once again. This L2 normalization is important to compensate for the loss of energy after the dimensionality reduction [57] and significantly improved the overall accuracy of the methods. After L2 normalization, both euclidean distance and dot product produce equivalent rankings since both measures are proportional after L2 normalization. We therefore use the dot product, since we observed it to be approximately 20 times faster than using the euclidean distance on our system.

6.3 Word Spotting

6.3.1 Protocol

In the word spotting task, the goal is to retrieve all instances of the query words in a "database" partition. Given a query, the database elements are sorted with respect to their similarity to the query. We then use mean average precision as the accuracy measure. Mean average precision is a standard measure of performance for retrieval tasks, and is essentially equivalent to the area below the precision-recall curve. Note that, since our search is exhaustive, the recall is always 100%. We use the test partition of the datasets as database, and use each of its individual words as a query in a leave-one-out style. When performing query-by-example, the query image is removed from the dataset, and queries that have no extra relevant words in the database are discarded. When performing query-by-string, only one instance of each string is considered as a query, *i.e.*, words that appear several times in the dataset are only used as a query once. In the IAM dataset it is customary to *not* use stopwords as queries. However, they still appear in the dataset and act as distractors. We follow this approach and not use stopwords as queries in the IAM dataset. The IAM dataset also contains a set of lines marked as "error", where the transcription of the line is dubious and may or may not be correct. We have filtered out those lines, and they are not used neither at training nor at testing.

Some methods in the literature have used slightly different protocols, that we will adopt when comparing to them. On the QBS experiments of Table 2, we report results using only queries that also appear on the training set, as the other approaches do, even if all the methods are able to perform OOV spotting. An exception is Aldavert *et al*. [55], that on GW reports results both using only queries that appear on training, and using all the queries. We follow the same approach. The results between parenthesis and marked with an asterisk denote that all queries were used. Furthermore, on the QBS experiments on IAM, we follow the state-of-the-art approach of [7] and perform *line spotting* instead of *word spotting*, *i.e.*, we retrieve whole lines that are correct if they contain the query word. To do so we group all the words in each line as a single entity, and define the distance between a query and a line as the distance between the query and the closest word in the line. Frinken *et al*. [7] also use only a subset of the test set containing approximately half of the test lines. We obtained the exact lines after contacting the authors and use only those lines when comparing to them.

For the spotting task on IIIT5K, we compare ourselves with the results of Rodríguez and Perronnin [17]. However, they use precision at 1 instead of mean average precision as the accuracy metric. Furthermore, instead of retrieving the test queries on the test partition, they retrieve test queries directly on the training partition. We follow the same approach when comparing to them in Table 3.

6.3.2 Results

The results of our approach on the word spotting task are shown on Table 1. For each dataset, we compare

the FV baseline (which can only be used in QBE tasks), the uncalibrated attributes embedding (Att.), the attributes calibrated with Platts (Att. + Platts), the one-way regression (Att. + Reg), the common subspace regression (Att. + CSR), and the kernelized common subspace regression (Att. + KCSR). We highlight the following points:

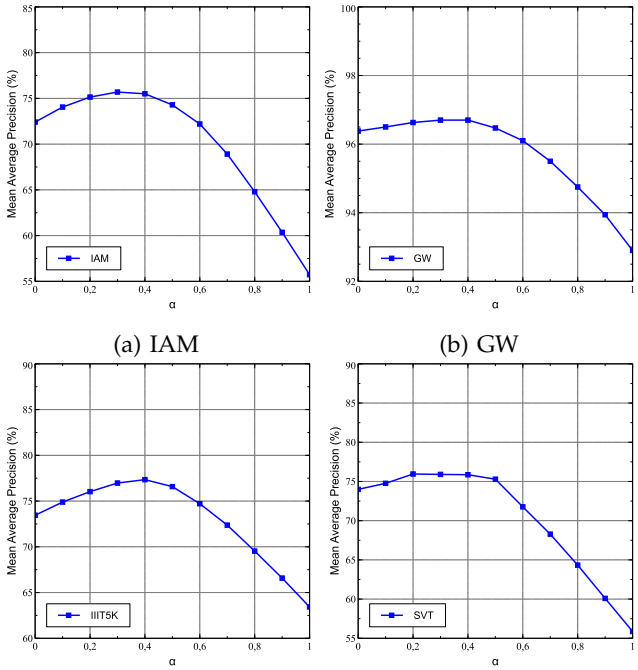
FV baseline vs attributes. It is clear how the use of attributes dramatically increases the performance, even when no calibration at all is performed. This is not surprising, since the attributes space has been learned using significant amounts of labeled training data. It reinforces the idea that exploiting labeled data during training is very important to obtain competitive results. The QBS results however are not particularly good, since the direct comparison between attribute scores and PHOCs is not well principled.

Platts vs reg vs CSR. By learning a non-linear calibration using Platts, the attribute results significantly improve for all datasets. Although Platts does not find a common subspace, it puts the attributes embedding and the label embedding in the same range of values, which obviously helps the performance, particularly in the QBS case. The results obtained with regression are unstable. Although they outperform the uncalibrated attributes, they only bring a slight improvement over Platts, and only in some cases. While regression exploits the correlation of attribute scores, the nonlinearity of Platts seems to give it an edge. However, when considering the common subspace regression, which exploits the correlations of both the attribute scores and the embedded labels, the results increase drastically, always outperforming Platts or regression except on the GW dataset, where they are very close.

CSR vs KCSR. The kernelized version of CSR obtains results very similar to CSR. Our intuition is that, due to the higher dimensionality of Random Fourier Features, the method is more prone to overfit, and requires more training data to show its benefits. Indeed, in the IAM dataset, which contains larger amounts of training data, KCSR clearly outperforms CSR.

Hybrid retrieval. We also explore a hybrid spotting approach, where both an embedded image $\psi_{\mathcal{I}}(\mathcal{I}_i)$ and its embedded transcription $\psi_{\mathcal{Y}}(\mathcal{Y}_i)$ are available as a query. Since both representations lie in the same space after the projection, we can create a new hybrid representation by a weighted sum, *i.e.*, $\psi_{\mathcal{H}}(\mathcal{I}_i, \mathcal{Y}_i) = \alpha\psi_{\mathcal{I}}(\mathcal{I}_i) + (1-\alpha)\psi_{\mathcal{Y}}(\mathcal{Y}_i)$ and use it as a query. Figure 8 shows the results of this hybrid approach on our datasets as a function of the α weight. We observe how the results improve when using both representations at query time.

Comparison with the state-of-the-art. We compare our approach with recently published methods on document and natural images. For the document datasets (Table 2), we first focus on QBE and compare our approach with the FV baseline and a DTW approach based on *Vinciarelli* [58] features. On GW, we report the results of [12] on DTW as well as their results with semi-continuous HMM (SC-HMM). Although the results of [12] are not exactly comparable, since partitions are different (we followed [7], [55] instead of [12]), we provided both our DTW results and the ones reported on their paper to at least provide an approximate idea of the expected differences due to the partition.



(a) IAM (b) GW

(c) IIIT5K (d) SVT

Figure 8: Hybrid spotting results with KCSR as a function of the weight α assigned to the visual part of the query.

Table 2: Word spotting comparison with the state-of-the-art on IAM and GW. Results on QBS only use queries that also appear on the training set, except those marked with an asterisk. QBS results on IAM perform *line spotting* instead of word spotting, and use only half of the lines of the test set.

	IAM		GW	
QBE	Baseline FV	15.66	Baseline FV	62.72
	DTW	12.30	DTW	60.63
			DTW [12]	50.00
			SC-HMM [12]	53.00
	Proposed (Platts)	48.09	Proposed (Platts)	93.04
	Proposed (KCSR)	**55.73**	Proposed (KCSR)	**92.90**
QBS	cHMM [6], [7]	36.00	cHMM [6], [7]	60.00
			Aldavert *et al.* [55]	76.20 (56.54*)
	Frinken *et al.* [7]	78.00	Frinken *et al.* [7]	84.00
	Proposed (KCSR)	**80.64**	Proposed (KCSR)	**93.93 (91.11*)**

We observe how the FV baseline is already comparable or outperforms some popular methods on both datasets. This is in line with the findings of [23],

where the FV of a HMM outperforms the standard HMM on keyword classification tasks. Also, despite the possible differences in partitions, the advantage of the proposed method over methods that do not perform supervised learning is clear. For the QBS case, we compare ourselves with the recent methods of [7] and [55], as well as with the character HMM approach of [6] as evaluated in [7]. All these works use labeled training data, which translates into more competitive results than in the QBE case. However, the expressiveness of the attributes, the use of discriminative image representations, and the learning of a common subspace, give an edge to the proposed method. We also note how our results do not particularly suffer when using words that were not seen during training (93.93 vs 91.11 on GW), as opposed to the approach of [55] (76.2 vs 56.54), showing a much nicer adaptation to unseen words.

We also compare our approach on natural images, see Table 3. We compare ourselves with the recent label-embedding method of [17], as well as their DTW baseline. Note that in this case the reported measure is precision at 1 instead of mean average precision. We observe how our approach clearly outperforms their results. Part of this improvement may however be due to using better Fisher vectors, since we use structured GMMs with enriched SIFT descriptors and Rodriguez and Perronnin use spatial pyramids.

Table 3: Word spotting comparison with the state-of-the-art in IIIT5K dataset for the QBE task.

Method	Top-1 acc.
FV	40.70
DTW [17]	37.00
Rodríguez and Perronnin [17]	43.70
Proposed (KCSR)	**72.28**

Qualitative results. We finally show qualitative results with samples from the IAM and IIIT5K datasets on Figure 9. We observe how some difficult words are correctly retrieved. Some common errors include words with common patterns (like a double tt) or different terminations ("window" vs "windows", "billboards" vs "billboard").

6.4 Word Recognition

6.4.1 Protocol

In word recognition, the goal is to find the transcription of the query word. In our experiments, the transcription is limited to words appearing in a lexicon. IIIT5K and SVT have officially associated lexicons. In SVT, each query word has an associated lexicon of 50 words, one of which corresponds to the transcription of the query. IIIT5K has two associated lexicons per word, one of 50 words and one of $1,000$ words. For IAM, we use a closed lexicon that contains all the words that appear in the test set, as in one of the experiments of [13]. In our case, since we embed both images and strings into a common subspace, the transcription problem is equivalent to finding the nearest neighbor of the query image in a dataset containing the lexicon embedded into the common subspace. In IIIT5K and SVT, the standard evaluation metric is precision at one, *i.e.*, is the top retrieved transcription correct? In document datasets, more common measures are the word error rate (WER) and character error rate (CER). The CER between two words is defined as the edit distance or Levensthein distance between them, *i.e.*, the minimum number of character insertions, deletions, and substitutions needed to transform one word into the other, normalized by the length of the words. We report the mean CER between the queries and their top retrieved transcription. The WER is defined similarly, but considering words in a text line instead of characters inside a word. This is done typically because many transcription methods work on whole lines at the same time. If words are already segmented, WER is equivalent to a Hamming distance between words in a line, and the dataset WER is the mean percentage of words that are wrongly transcribed in each line.

6.4.2 Results

The results on word recognition on IAM are on Table 4, where we compare with the state-of-the-art approach of [13], which uses a combination of HMMs and neural networks to clean and normalize the images and produce word models. We compare our results in terms of WER and CER, where a lower score is better. Although we do not match the results of [13], our results are competitive without performing any costly preprocessing on the images and with much faster recognition speeds.

Table 4: Recognition error rates on the IAM dataset.

Method	WER	CER
España-Bosquera *et al*. [13]	**15.50**	**6.90**
Proposed (KCSR)	20.01	11.27

Table 5 shows results on the IIIT5K an SVT datasets. We observe how our approach clearly outperforms all published results on IIIT5K. On SVT, our method is only outperformed by Google's very recent PhotoOCR [5] approach. However, [5] uses 2.2 *million* training samples labeled at the *character* level, while we use only 5 *thousand* images labeled at the *word* level.

Finally, Figure 10 shows some qualitative results on image recognition with samples from the IAM, IIIT5K and SVT datasets. Common errors include very similar-looking words ("Heather" and "feather"), low-resolution images ("Mart" and "man"), or unorthodox font styles ("Neumos" vs "bimbos").

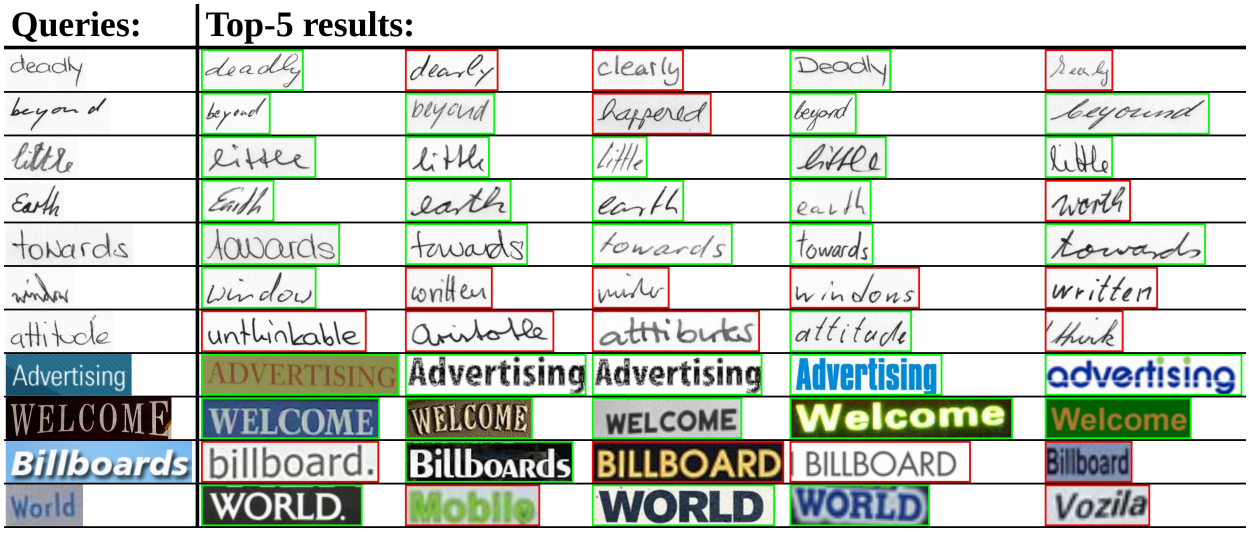


Figure 9: Qualitative results on word spotting on the IAM and IIIT5K. Relevant words to the query are outlined in green.

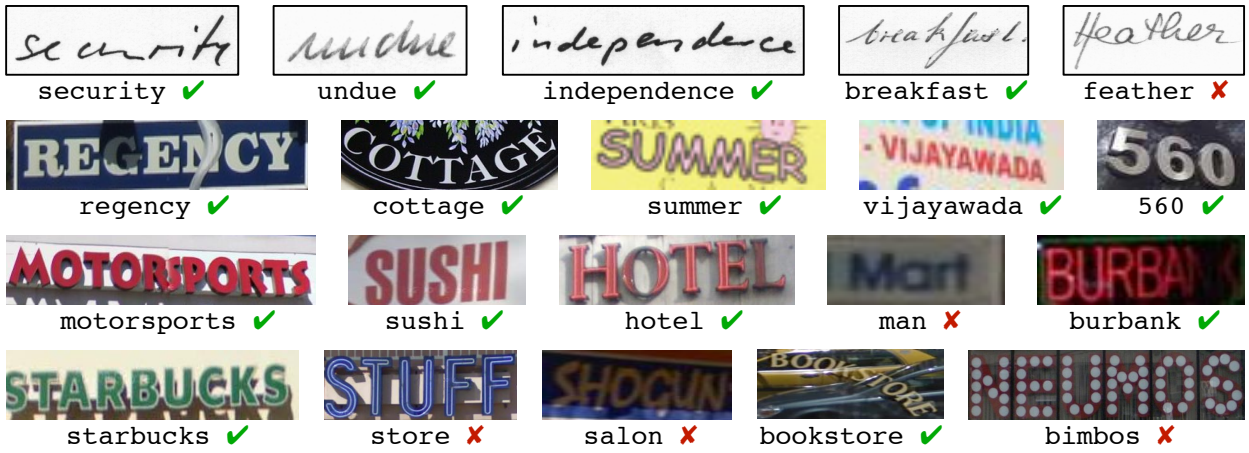


Figure 10: Qualitative results on word recognition on the IAM, IIIT5K, and SVT datasets.

Table 5: Recognition results on the IIIT5K and SVT dataset. Accuracy measured as precision at 1.

Dataset	Method	$\|y\| = 50$	$\|y\| = 1000$
IIIT5K	Mishra *et al.* [30]	64.10	57.50
	Rodríguez and P. [17]	76.10	57.40
	Proposed (KCSR)	**88.57**	**75.60**
SVT	ABBY [32]	35.00	-
	Mishra *et al.* [16]	73.26	-
	Goel *et al.* [32]	77.28	-
	PhotoOCR [5]	**90.39**	-
	Proposed (KCSR)	87.01	-

6.5 Computational Analysis

The improvements of our approach are not only in terms of accuracy and memory use. Our optimized DTW implementation in C took more than 2 hours to compare the $5,000$ queries of IAM against the $16,000$ dataset words on an 8-core Intel Xeon W3520 at 2.67GHz with 16Gb of RAM, using one single core. By contrast, comparing the same queries using our attributes embedded with CSR involves only one matrix multiplication and took less than 1 second on the same machine, about 0.2 milliseconds per query. For recognition tasks we only need to compare the query with the given lexicon. Recognizing a query with a lexicon of $1,000$ in IIIT5K takes less than 0.02 milliseconds. At query time we also need to extract the FV representation of the query image, which involves the dense SIFT extraction and the FV encoding, and then embed it into the CSR/KCSR subspace. This process takes, on average, 0.77 seconds per image.

In general, these numbers compare favorably with other approaches. The method proposed by [7] takes a few milliseconds to process a single line in the IAM for the QBS task. PhotoOCR [5] reports times of around 1.4 seconds to recognize a cropped image using a setup tuned for accuracy. An unoptimized implementation of the method proposed in [4] takes 35 seconds to locate and recognize the words in an

image, and the same task takes around 8 seconds with the method in [31].

Regarding the cost of learning an attribute model with an SVM, learning a single model on the IAM database using our SGD implementation on a single core with more than $60,000$ training samples took, on average, 35 seconds, including the crossvalidation of the SVM parameters. That is, the complete training process of the attributes, including the bagging, can be done in about 2 days on a single CPU. Since attributes and folds are independent, this is trivially parallelizable. Training the Deep Neural Network proposed in [5] took 2 days on a 200 cores cluster. Learning the CSR and KCSR projections is also very fast since the dimensionality of the attribute space is low: approximately 1.5 seconds for CSR and approximately 60 seconds for KCSR. As the attribute models, this needs to be learned only once, offline.

7 Conclusions and Future work

This paper proposes an approach to represent and compare word images, both on document and on natural domains. We show how an attributes-based approach based on a pyramidal histogram of characters can be used to learn how to embed the word images and their textual transcriptions into a shared, more discriminative space, where the similarity between words is independent of the writing and font style, illumination, capture angle, etc. This attributes representation leads to a unified representation of word images and strings, resulting in a method that allows one to perform either query-by-example or query-by-string searches, as well as image transcription, in a unified framework. We test our method in four public datasets of documents and natural images, outperforming state-of-the-art approaches and showing that the proposed attribute-based representation is well-suited for word searches, whether they are images or strings, in handwritten and natural images.

Regarding future work, we have observed empirically that the quality of the attribute models is quite dependent on the available number of training samples, and that the models for rare characters in rare positions were not particularly good. We believe that having larger training sets could improve the quality of those models and lead to better overall results. Towards this goal, we have experimented with augmenting the training sets by applying transformations such as changes in slant to the available images. Preliminary results indicate that this can significantly boost the results. As an example, we improved the QBE results on IAM from 55.73 to 59.62. In the same line, our learning approach is currently based on whole word images and does not require to segment the individual characters of the words during training or test, which we consider an advantage. However, it has been shown that learning on characters can lead to large improvements in accuracy [5]. We want to study how we could learn on individual segmented characters (using existing datasets such as Char74K [59]) and transfer that information into our system without needing to actually segment the characters of the target dataset at any time. We are also interested in lifting the need to have cropped word images by integrating the current word representation in an efficient sliding window framework.

Acknowledgements

J. Almazán, A. Fornés, and E. Valveny are partially supported by the Spanish projects TIN2011-24631, TIN2009-14633-C03-03, TIN2012-37475-C02-02, by the EU project ERC-2010-AdG-20100407-269796 and by a research grant of the UAB (471-01-8/09).

References

[1] J. Almazán, A. Gordo, A. Fornés, and E. Valveny, "Handwritten word spotting with corrected attributes," in *ICCV*, 2013.
[2] R. Manmatha and J. Rothfeder, "A scale space approach for automatically segmenting words from historical handwritten documents," *IEEE TPAMI*, 2005.
[3] L. Neumann and J. Matas, "Real-Time Scene Text Localization and Recognition," in *CVPR*, 2012.
[4] L. Neumann and J. Matas, "Scene Text Localization and Recognition with Oriented Stroke Detection," in *ICCV*, 2013.
[5] A. Bissacco, M. Cummins, Y. Netzer, and H. Neven, "PhotoOCR: Reading Text in Uncontrolled Conditions," in *ICCV*, 2013.
[6] A. Fischer, A. Keller, V. Frinken, and H. Bunke, "HMM-based word spotting in handwritten documents using subword models," in *ICPR*, 2010.
[7] V. Frinken, A. Fischer, R. Manmatha, and H. Bunke, "A novel word spotting method based on recurrent neural networks," *IEEE TPAMI*, 2012.
[8] R. Manmatha, C. Han, and E. M. Riseman, "Word spotting: A new approach to indexing handwriting," in *CVPR*, 1996.
[9] T. Rath, R. Manmatha, and V. Lavrenko, "A search engine for historical manuscript images," in *SIGIR*, 2004.
[10] T. Rath and R. Manmatha, "Word spotting for historical documents," *IJDAR*, 2007.
[11] J. A. Rodríguez-Serrano and F. Perronnin, "Local gradient histogram features for word spotting in unconstrained handwritten documents," in *ICFHR*, 2008.
[12] J. A. Rodríguez-Serrano and F. Perronnin, "A model-based sequence similarity with application to handwritten word-spotting," *IEEE TPAMI*, 2012.
[13] S. España-Bosquera, M. Castro-Bleda, J. Gorbe-Moya, and F. Zamora-Martinez, "Improving offline handwritten text recognition with hybrid HMM/ANN models," *IEEE TPAMI*, 2011.
[14] I. Yalniz and R. Manmatha, "An efficient framework for searching text in noisy documents," in *DAS*, 2012.
[15] K. Wang, B. Babenko, and S. Belongie, "End-to-end Scene Text Recognition," in *ICCV*, 2011.
[16] A. Mishra, K. Alahari, and C. V. Jawahar, "Top-down and bottom-up cues for scene text recognition," in *CVPR*, 2012.
[17] J. A. Rodríguez-Serrano and F. Perronnin, "Label embedding for text recognition," in *BMVC*, 2013.
[18] C. Leslie, E. Eskin, and W. Noble, "The spectrum kernel: A string kernel for SVM protein classification," in *Pacific Symposium on Biocomputing*, 2002.
[19] H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, and C. Watkins, "Text classification using string kernels," *JMLR*, 2002.
[20] H. Jégou, M. Douze, and C. Schmid, "Product quantization for nearest neighbor search," *IEEE TPAMI*, 2011.

[21] S. Lu, L. Li, and C. L. Tan, "Document image retrieval through word shape coding," *IEEE TPAMI*, 2008.
[22] P. Keaton, H. Greenspan, and R. Goodman, "Keyword spotting for cursive document retrieval," in *DIA*, 1997.
[23] F. Perronnin and J. A. Rodríguez-Serrano, "Fisher kernels for handwritten word-spotting," in *ICDAR*, 2009.
[24] T. Jaakkola and D. Haussler, "Exploiting generative models in discriminative classifiers," in *NIPS*, 1999.
[25] B. Gatos and I. Pratikakis, "Segmentation-free word spotting in historical printed documents," in *ICDAR*, 2009.
[26] M. Rusiñol, D. Aldavert, R. Toledo, and J. Lladós, "Browsing heterogeneous document collections by a segmentation-free word spotting method," in *ICDAR*, 2011.
[27] J. Almazán, A. Gordo, A. Fornés, and E. Valveny, "Efficient exemplar word spotting," in *BMVC*, 2012.
[28] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *CVPR*, 2005.
[29] T. Malisiewicz, A. Gupta, and A. Efros, "Ensemble of Exemplar-SVMs for object detection and beyond," in *International Conference on Computer Vision, 2011*, 2011.
[30] A. Mishra, K. Alahari, and C. V. Jawahar, "Scene text recognition using higher order language priors," in *BMVC*, 2012.
[31] A. Mishra, K. Alahari, and C. V. Jawahar, "Image Retrieval using Textual Cues," in *ICCV*, 2013.
[32] V. Goel, A. Mishra, K. Alahari, and C. V. Jawahar, "Whole is Greater than Sum of Parts: Recognizing Scene Text Words," in *ICDAR*, 2013.
[33] C. H. Lampert, H. Nickisch, and S. Harmeling, "Learning to detect unseen object classes by between-class attribute transfer," in *CVPR*, 2009.
[34] A. Farhadi, I. Endres, D. Hoiem, and D. Forsyth, "Describing objects by their attributes," in *CVPR*, 2009.
[35] M. Rohrbach, M. Stark, and B. Schiele, "Evaluating knowledge transfer and zero-shot learning in a large-scale setting," in *CVPR*, 2011.
[36] C. Wah and S. Belongie, "Attribute-Based Detection of Unfamiliar Classes with Humans in the Loop," in *CVPR*, 2013.
[37] A. Fischer, A. Keller, V. Frinken, and H. Bunke, "Lexicon-free handwritten word spotting using character HMMs," *PRL*, 2012.
[38] S. Bengio, J. Weston, and D. Grangier, "Label embedding trees for lagre multi-class tasks," in *NIPS*, 2010.
[39] Z. Akata, F. Perronnin, Z. Harchaoui, and C. Schmid, "Label-embedding for attribute-based classification," in *CVPR*, 2013.
[40] F. Perronnin, J. Sánchez, and T. Mensink, "Improving the Fisher kernel for large-scale image classification," in *ECCV*, 2010.
[41] B. Siddiquie, R. S. Feris, and L. S. Davis, "Image Ranking and Retrieval Based on Multi-Attribute Queries," in *CVPR*, 2011.
[42] L. Torresani, M. Szummer, and A. Fitzgibbon, "Efficient object category recognition using classemes," in *ECCV*, 2010.
[43] K. Chatfield, V. Lempitsky, A. Vedaldi, and A. Zisserman, "The devil is in the details: an evaluation of recent feature encoding methods," in *BMVC*, 2011.
[44] G. Csurka, C. R. Dance, L. Fan, J. Willamowski, and C. Bray, "Visual categorization with bags of keypoints," in *Workshop on Statistical Learning in Computer Vision, ECCV*, 2004.
[45] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond bags of features: spatial pyramid matching for recognizing natural scene categories," in *CVPR*, 2006.
[46] J. Sánchez, F. Perronnin, and T. de Campos, "Modeling the spatial layout of images beyond spatial pyramids," *PRL*, 2012.
[47] W. J. Scheirer, N. Kumar, P. N. Belhumeur, and T. E. Boult, "Multi-attribute spaces: Calibration for attribute fusion and similarity search," in *CVPR*, 2012.
[48] H. Hotelling, "Relations between two sets of variants," *Biometrika*, 1936.
[49] D. R. Hardoon, S. Szedmak, and J. Shawe-Taylor, "Canonical correlation analysis; an overview with application to learning methods," tech. rep., Department of Computer Science, Royal Holloway, University of London, 2003.
[50] M. B. Blaschko and C. H. Lampert, "Correlational spectral clustering," in *CVPR*, 2008.
[51] Y. Gong, S. Lazebnik, A. Gordo, and F. Perronnin, "Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval," *IEEE TPAMI*, 2012.
[52] J. Weston, S. Bengio, and N. Usunier, "Large scale image annotation: Learning to rank with joint word-image embedding," in *ECML*, 2010.
[53] A. Rahimi and B. Recht, "Randomm features for large-scale kernel machines," in *NIPS*, 2007.
[54] U.-V. Marti and H. Bunke, "The IAM-database: An english sentence database for off-line handwriting recognition," *IJDAR*, 2002.
[55] D. Aldavert, M. Rusiñol, R. Toledo, and J. Lladós, "Integrating Visual and Textual Cues for Query-by-String Word Spotting," in *ICDAR*, 2013.
[56] J. Sánchez, F. Perronnin, T. Mensink, and J. Verbeek, "Image classification with the Fisher vector: Theory and practice," *IJCV*, 2013.
[57] H. Jégou and O. Chum, "Negative evidences and cooccurrences in image retrieval: the benefit of pca and whitening," in *ECCV*, 2012.
[58] A. Vinciarelli and S. Bengio, "Offline cursive word recognition using continuous density hidden markov models trained with PCA or ICA features," in *ICPR*, 2002.
[59] T. de Campos, B. Babu, and M. Varma, "Character recognition in natural images," in *VISAPP*, 2009.

Jon Almazán received his B.Sc. degree in Computer Science from the Universitat Jaume I (UJI) in 2008, and his M.Sc. degree in Computer Vision and Artificial Intelligence from the Universitat Autònoma de Barcelona (UAB) in 2010. He is currently a PhD student in the Computer Science Department and the Computer Vision Center at UAB, under the supervision of Ernest Valveny and Alicia Fornés. His research work is mainly focused on document image analysis, image retrieval and word spotting and recognition.

Albert Gordo received his PhD in computer vision in 2013 from the Universitat Autònoma de Barcelona / Computer Vision Center, in collaboration with the Xerox Research Centre Europe in Grenoble, France. After that, he spent a year as a postdoc at the LEAR group at INRIA. He is currently a postdoc at XRCE. His research interests range from document analysis and word spotting and recognition to fine-grained object classification and detection, with a special focus on large-scale problems.

Alicia Fornés received the B.S. degree from the Universitat de les Illes Balears (UIB) in 2003 and the M.S. degree from the Universitat Autònoma de Barcelona (UAB) in 2005. She obtained the Ph.D. degree on writer identification of old music scores from the UAB in 2009. She was the recipient of the AERFAI (Image Analysis and Pattern Recognition Spanish Association) best thesis award 2009-2010. She is currently a postdoctoral researcher in the Computer Vision Center. Her research interests include document analysis, symbol recognition, optical music recognition, historical documents, handwriting recognition and writer identification.

Ernest Valveny is an Associate Professor at the Computer Science Department of the Universitat Autònoma de Barcelona (UAB), where he obtained his PhD degree the in 1999. He is also member of the Computer Vision Center (CVC) at UAB. His research work has mainly focused on document analysis and pattern recognition, and more specifically in the fields of shape representation, text recognition and retrieval, document classification and graph matching. He is currently a member of IAPR and of the editorial board of the International Journal on Document Analysis and Recognition and has served as a reviewer for many international journals and conferences. He has published more than 100 papers in international journals and conferences.