Learning to Represent Handwritten Shapes and Words for Matching and Recognition

A dissertation submitted by **Jon Almazán** at Universitat Autònoma de Barcelona to fulfil the degree of **Doctor of Philosophy**.

Bellaterra, September 1, 2014

Director	**Dr. Ernest Valveny** Dept. Ciències de la Computació & Centre de Visió per Computador Universitat Autònoma de Barcelona
Co-director	**Dr. Alicia Fornés** Dept. Ciències de la Computació & Centre de Visió per Computador Universitat Autònoma de Barcelona
Thesis committee	**Dr. Karteek Alahari** INRIA Grenoble, France **Dr. Marçal Rusiñol** Computer Vision Center Barcelona, Spain **Dr. José A. Rodríguez Serrano** Xerox Research Center Europe Grenoble, France
International evaluators	**Dr. Volkmar Frinken** Kyushu University Fukuoka, Japan **Dr. Mickaël Coustaty** Université de la Rochelle La Rochelle, France

This document was typeset by the author using LaTeX 2ε.

The research described in this book was carried out at the Computer Vision Center, Universitat Autònoma de Barcelona.

ISBN: XXXXX

Printed by Ediciones Gráficas Rey, S.L.

To my parents . . .

Acknowledgements

> It is easy not to believe in monsters,
> considerably more difficult to escape their
> dread and loathsome clutches.
>
> Stanisław Lem - *The Cyberiad*

I would like to start by deeply thanking my thesis directors Ernest Valveny and Alicia Fornés for their excellent supervision and for making possible that I am here today writing these last lines of my thesis. For your excellence guidance, caring, and patience. Thanks for your endless bag of ideas. I could not have imagined having a better advisors and mentors.

I would like to thank my friend and colleague Albert Gordo. Thanks for the countless amount of hours that we have shared designing methods, coding experiments and analyzing results. Thanks for sharing with me your brilliant knowledge and ideas. Thanks for pushing me further than I could have ever imagined. And thanks for all the great times that we have had at work, and specially for those out from work.

I would also like to thank all the people that have helped me to shape this work during these years. First, to all my colleges in the Computer Vision Center, and specially those in the Document Analysis Group. Thanks for your useful comments and suggestions. Thanks to my colleges at the Osaka Prefecture University, where I had a short, but profitable and intense research stay. And to my colleges at the University Jaume I, whom helped me to realize that this is what I really wanted to do. Finally, to professor Jaap van de Loosdrecht at the VisionLab in the Noordelijke Hogeschool Leeuwarden, whom first planted this seed for my interest in Computer Vision.

Since the first day I moved to Barcelona to start my doctorate studies I have had the luck to find the best possible group of friends, which rapidly became what I consider now my family in Barcelona. Thanks for the all the great times and fun that we have had. And I would also like to thank my friends in Castellón. It was really hard to separate me from you after more than 20 years together, but it is amazing to feel every time that I come back like I had never left. Thanks for your support and unconditional friendship.

There are not enough words to thank Elena for her endless love and support. For being there when I most needed it. And for bearing with me during my long absences far from home and during my long absences in our desk at home. Thanks for the amazing times we have had in this stage of our life, and for those that for sure we are going to have. *Gracias pequeña.*

Finally, I would like to thank my whole family for making me the person I am, and specially to my parents and sister. Thanks for your unconditional love, for thinking that I am the smartest guy on Earth and for always supporting and encouraging me. *Muchísimas gracias, os quiero.*

Abstract

Writing is one of the most important forms of communication and for centuries, handwriting had been the most reliable way to preserve knowledge. However, despite the recent development of printing houses and electronic devices, handwriting is still broadly used for taking notes, doing annotations, or sketching ideas. In order to be easily accessed, there is a huge amount of handwritten documents, some of them with uncountable cultural value, that have been recently digitized. This has made necessary the development of methods able to extract information from these document images.

Transferring the ability of understanding handwritten text or recognizing handwritten shapes to computers has been the goal of many researches due to its huge importance for many different fields. However, designing good representations to deal with handwritten shapes, *e.g.* symbols or words, is a very challenging problem due to the large variability of these kinds of shapes. One of the consequences of working with handwritten shapes is that we need representations to be *robust*, *i.e.*, able to adapt to large intra-class variability. We need representations to be *discriminative*, *i.e.*, able to learn what are the differences between classes. And, we need representations to be *efficient*, *i.e.*, able to be rapidly computed and compared. Unfortunately, current techniques of handwritten shape representation for matching and recognition do not fulfill some or all of these requirements.

Through this thesis we focus on the problem of learning to represent handwritten shapes aimed at retrieval and recognition tasks. Specifically, on the first part of the thesis, we focus on the general problem of representing handwritten shapes for classification and matching. We first present a novel shape descriptor based on a deformable grid that deals with large deformations by adapting to the shape and where the cells of the grid can be used to extract different features. Then, we propose to use this descriptor to learn statistical models, based on the Active Appearance Model, that jointly learns the variability in structure and texture of a given shape class. Then, on the second part, we focus on a concrete application, the problem of word spotting, where the goal is to find all instances of a query word in a dataset of images. First, we address the segmentation-free problem and propose an unsupervised, sliding-window-based approach that achieves state-of-the-art results in two public datasets. Second, we address the more challenging multi-writer problem, where the variability in words exponentially increases. We describe an approach in which both word images and text strings are embedded in a common vectorial subspace, and where those that represent the same word are close together. This is achieved by a combination of label embedding and attributes learning, and a common subspace regression. This leads to a low-dimensional, unified representation of word images and strings, resulting in a method that allows one to perform either image and text searches, as well as image transcription, in a unified framework. 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.

Resumen

La escritura es una de las formas más importantes de comunicación y durante siglos ha sido el método más fiable para preservar conocimiento. Sin embargo, a pesar del reciente desarrollo de las imprentas y dispositivos, la escritura manuscrita todavía se utiliza ampliamente para tomar notas, hacer anotaciones, o dibujar bocetos. Con el fin de hacerlos fácilmente accesibles, hay una enorme cantidad de documentos escritos a mano, algunos de ellos con un valor cultural incalculable, que han sido recientemente digitalizados. Esto ha hecho necesario el desarrollo de métodos capaces de extraer información de este tipo de imágenes.

Transferir a los ordenadores la capacidad de comprender y reconocer texto y formas escritas a mano ha sido el objetivo de muchos investigadores debido a su gran importancia para muchos campos diferentes. Sin embargo, el diseõ de buenas representaciones para lidiar con formas manuscritas es un problema muy difícil debido a la gran variabilidad en este tipo de formas. Una de las consecuencias de trabajar con formas escritas a mano es que necesitamos representaciones que sean *robustas*, es decir, capaces de adaptarse a la gran variabilidad interna de la clase. Necesitamos representaciones que sean *discriminativas*, es decir, capaces de aprender cuáles son las diferencias entre las clases. Y necesitamos representaciones que sean *eficientes*, es decir, capaces de ser calculadas y comparadas con rapidez. Desafortunadamente, las técnicas actuales de representación de formas manuscritas para la recuperación y el reconocimiento no cumplen todos o algunos de estos requisitos.

En esta tesis nos centramos en el problema de aprender a representar formas manuscritas dirigido a tareas de recuperación y reconocimiento. En concreto, en la primera parte de la tesis, nos centramos en el problema general de la representación de formas manuscritas para clasificación y reconocimiento. Primero presentamos un descriptor de forma basado en una rejilla deformable que se adapta a grandes deformaciones y donde las celdas de la cuadrícula se utilizan para extraer diferentes características. Seguidamente, proponemos utilizar este descriptor para aprender modelos estadísticos basados en el Active Appearance Model, que aprende de forma conjunta la variabilidad en la estructura y la textura de una determinada clase. En la segunda parte nos centramos en una aplicación concreta, el problema de *word spotting*, donde el objetivo es encontrar todas las instancias de una palabra dada en un conjunto de imágenes. En primer lugar, abordamos el problema sin segmentación previa y proponemos un enfoque no supervisado, basado en ventana deslizante que supera el estado del arte en dos datasets públicos. En segundo lugar, se aborda el problema de *word spotting* con varios escritores, donde la variabilidad de palabras aumenta exponencialmente. Se describe un método en el que las imágenes de texto y cadenas de texto se proyectan en un subespacio vectorial común, y donde aquellos vectores que representan la misma palabra están más próximos. Esto se logra gracias a una combinación de *label embedding* y aprendizaje de atributos, y una regresión a subespacio común. Evaluamos nuestro método en bases de datos públicas de documentos manuscritos e imágenes naturales que muestran resultados comparables o mejores que el estado del arte en tareas de búsqueda y reconocimiento.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Representing Handwritten Shapes

Since the first writing systems appeared in the first big cities in Mesopotamia to the present day with the enormous increase of electronic devices, handwriting has been a basic way of communication in constant evolution. Writing was first introduced as a way to organize cities, their population and the stock contained within their walls, but for many centuries since then it has been considered as the most reliable way to preserve knowledge. Therefore, there is a huge amount of information with incalculable cultural value contained in handwritten books, and, although this drastically changed some centuries ago with the appearance of the first printing systems and even more with the recent increase of electronic devices, there is still a huge amount of documents produced every day that contain handwritten text or hand-drawn symbols: personal notebooks, handwritten annotations in printed documents, forms that constantly arrive to companies, signatures in official documents, sketches and drawings, *etc*.

Besides the cultural differences and barriers, humans are able to interpret handwritten shapes or to understand words that may have been written by different persons with completely different writing styles, as long as they share the same alphabet. Transferring this capacity to computers has been the goal for many researchers due to its importance for many different fields, such as historians, sociologists or companies.

A lot of effort from the Document Image Analysis community has been devoted to solve this problem, since it is a core aspect of many important tasks in this field. Symbol recognition, word spotting, word recognition, document retrieval, or writer identification are some examples, which, in general, have addressed the problem of dealing with handwritten shapes form different view points and with different techniques. However, they all share a common objective: how to robustly represent them.

Designing good representations for handwritten shapes is a very challenging problem due to the large variability that is usually common in these shapes. Distortions such as deformations in strokes, inaccuracy in junctions, missing parts, stroke duplication, overlapping, gaps, or problems like over-tracing directly affects to the intra-class compactness and inter-class separability. This variability exponentially increases when different writers are involved: each person has his/her own writing style and this affects the way they draw strokes or the way they write characters. To illustrate some of these problems, we show in Figure 1.1 some examples of words and symbols handwritten by different writers. To deal with them, we need representations that learn what makes a shape unique, what variability a given class

may have, and what differentiates it from others.

Figure 1.1: Example of the variability in handwritten shapes that comes from different writing styles.

1.2 Objectives of this Thesis

In the previous section we noted the common problems when dealing with handwritten shapes. To overcome these problems, we need representations that efficiently adapt to the variability while providing a high discriminative power. In other words, we need methods able to learn what are the common deformations inside a class and what are the most discriminative characteristics to recognize it. In particular, these are the desirable key points when designing a good representation:

- **Robustness:** a large intra-class variability is very common in handwritten shapes. Shapes from the same class may present large variations in different strokes, or changes in the way these strokes are spatially arranged. This is specially pronounced when different writers are involved. As a consequence of this, when designing a good representation for handwritten shapes, this has to be able to adapt to these deformations and learn the intra-class variability in order to correctly match shapes from the same class.
- **Discriminativeness:** in many tasks, specially when dealing with words, classes are very fine-grained, and a small variation in a single stroke may be the only difference between several classes. This directly conflicts with the previous property, since the representation has to be able to detect that small variations while adapting to those large intra-class variations. In other words, methods should learn what are the deformations that better discriminate between classes.
- **Efficiency:** with the recent increase of digitized documents, developing efficient methods has become of the utmost importance. Therefore, a good and useful representation has to be fast to compute, but most importantly, fast to compare, in order to allow one to efficiently search in large collections of images.

In the literature, as we will see in following chapters, many methods and representations have been proposed to deal with these kinds of shapes, some of them specifically designed for the handwriting domain and others adapted from different domains. Unfortunately, most of these representations do not fulfill these desirable properties. The main obstacle is due to the fact that, in many applications and tasks, these properties are seemingly conflicting: methods should be able to adapt to large variability but should be discriminative enough to detect those, possibly small, relevant deformations, while keeping a compact and

efficient representation. Some works propose complex and variable-length representations that succeed when it comes to adapting to the handwritten shape but require costly similarity distances, and others propose simple and efficient methods that have difficulties to adapt to large variability.

In this thesis, we first focus on designing efficient methods in order to be able to deal with large datasets and collections of documents: fixed-length representations that are fast to compare and can be efficiently compressed and indexed. Then, in order to propose robust and discriminative methods, we argue that they should be able to *learn* both inter-class and intra-class variabilities. Without this learning process, adapting to the deformations that may come from new writers or unseen classes becomes a very difficult, and sometimes impossible, task. Following this design philosophy, we will first address the general problem of representing any kind of handwritten shape. Then, we will focus on a specific application, the representation of words.

Considering this, two are the main objectives of this thesis:

- To propose a general method to represent any kind of handwritten shape that can be applied to different tasks, *e.g.*, retrieval and recognition of symbols, digits or words. For this purpose, we need first a robust descriptor that adapts to large deformations, and in addition to that, a method that learns this inter- and intra-class variability in order to model and recognize a given class.
- To propose a method that deals with the problem of word spotting, a concrete case of handwritten shape retrieval. First, we address the problem of segmentation-free word spotting, where we need a method that can efficiently process pages in large collections of documents. Then, we address the problem of multi-writer word spotting, where the intra-class variability exponentially increases. To overcome this, we should be able to learn this variability and transfer this knowledge to new classes and unseen writers. However, we still aim to design a compact and efficient low-dimensional representation.

In following sections we will first describe the organization of this thesis and then we will summarize the main contributions.

1.3 Organization of this Thesis

This thesis follows a compendium of publications format, therefore each of the following chapters corresponds to one of the publications included for this compendium. Then, it closes with a final chapter that presents a global conclusion from these individual works. These chapters are self-contained, *i.e.*, they first introduce the problem to address while reviewing the state-of-the-art, then propose a methodology, and finally show the experimental validation and conclude the work. In this section, we highlight the main contributions of these works and detail the connections between them.

The thesis is divided in two parts, where both parts propose methods to deal with handwritten shapes and words, although for different tasks. The first part comprises the publications that address the first objective of this thesis: the problem of representing handwritten shape for matching and recognition. Then, we focus on a concrete application, the problem of word spotting. Therefore, the publications in the second part of this thesis address this problem, in both segmentation-free and multi-writer scenarios.

Concretely, the publications in Part 1 deal with the following problems:

- In chapter 2, we deal with the problem of representing handwritten shapes by learning patterns of variability. We first propose two deformable feature extraction methods, which are based on a non-rigid grid that efficiently adapts to the shape to described.

These descriptors will allow us to extract information about the appearance of the shape, by extracting features in the final locations of the grid's cells, and the structure of the shape, by tracking the deformation of the grid. Then, we propose a method for generating statistical models based on the Active Appearance Model that, in combination with the deformable descriptor, jointly learns the variability im appearance and structure of the shape, *i.e.*, matches variations in appearance and structure. Finally, we integrate this model in two different classification schemes and evaluate them for handwritten digit and hand-drawn symbol recognition tasks.

- In chapter 3, we propose to use the deformable grid introduced in the previous chapter to extend the Histogram of Oriented Gradients (HOG) descriptor for the specific problem of representing handwritten shapes. HOG has been shown to obtain excellent results in tasks such as object recognition or pedestrian detection. However, we argue that the rigid grid have difficulties to adapt to the variability in handwritten shapes. The new descriptor proposed overcomes the rigidity of the original HOG descriptor while maintaining its discriminative power, resulting in a robust and fixed-length representation that efficiently adapts to the variability of handwritten shapes and captures fine-grained details. Finally, this novel feature extraction method is evaluated for two different tasks, showing the generality of the descriptor: hand-drawn symbol recognition and handwritten word retrieval. Both tasks will show the ability of the feature extraction method when applied to writer independent scenarios and images with different scales and aspect ratios, and that the flexibility of the deformable grid considerably improves the performance of the original rigid grid.

In this chapter 3 we show how a gradient-based representation can perform well when applied to handwritten words. This motivated us to go in depth into the problem of word spotting and word recognition, which we address in the second part of this thesis. In particular, the publications in Part 2 deal with the following problems:

- In chapter 4, we address the problem of segmentation-free word spotting in document images. First, we review the family of segmentation-free word-spotting approaches and argue that current methods can be improved in several ways. Then, we propose an unsupervised method where documents are represented with a grid of HOG descriptors, and a sliding-window approach is used to locate the document regions that are most similar to the query. HOG provides a good trade-off between speed and memory requirements and discriminative power, and, although the deformable grid proposed in previous chapters provides a considerable performance improvement, its grid-based representation makes it suitable for a sliding window-based search. We leave the combination of the deformable grid with the sliding window as a future research line.

  To produce a better representation of the query in an unsupervised way, we propose to use the Exemplar SVM framework. Then, we use a more discriminative representation to rerank the best regions retrieved, and the most promising ones are used to expand the Exemplar SVM training set and improve the query representation. Finally, the document descriptors are precomputed and compressed with Product Quantization (PQ), providing us an important boost in efficiency. Encoding the descriptors with PQ would allow us to reduce the size of the descriptors and to preserve a much larger amount of images in RAM at the same time. As a side effect, computing the scores of the sliding window also becomes significantly faster since distances between quantized HOG descriptors can be precomputed. We show that our results significantly outperform other segmentation-free methods in the literature, both in accuracy and in speed and memory usage.

- In chapter 5, we deal with the problems of word spotting and word recognition on images in a multi-writer, multi-domain scenario. The variability in this kind of scenario is usually very large, and unsupervised methods as that presented in chapter 4 would have difficulties. To overcome this problem, we propose to learn a semantic representation of the word images in a supervised way by using character attribute models.

  We start by reviewing the works most related to some key aspects of our proposed approach. Then, 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 in a very low dimensional space. For this purpose, we first propose a label embedding approach for text labels, which is then is used as a source of character attributes that will be used to encode the word image. However, due to some differences, direct comparison is not principled and some calibration is needed. We finally propose to learn a low-dimensional common subspace with an associated metric between the label embedding and the attributes embedding. Contrary to most other existing methods, an important advantage is that our representation has a fixed length, is low dimensional, and is very fast to compute and, especially, to compare. And, since we use compact vectors, compression and indexing techniques such as Product Quantization could now be used to perform spotting in very large datasets. Moreover, the method results in a unified framework that indistinctly allows one to perform either query-by-example or query-by-string searches, as well as image transcription. We finally 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.

1.4 Contributions of this Thesis

Finally, we summarize the major contributions of this thesis:

1. We propose a novel descriptor for handwritten shapes based on a deformable grid. This grid efficiently adapts to the variability of the shape and it is used to represent the shape by extracting features in the final locations of the grid's cells. We have explored different deformation procedures and compared density and gradient-based features.
2. We present a method for generating statistical models based on the Active Appearance Model that jointly learns the variability in structure and appearance of a given shape class. We also present two different classification schemes for shape recognition tasks where models from different classes are integrated.
3. We propose an unsupervised, segmentation-free method for word spotting in document images. We show how the Exemplar-SVM framework in combination with a reranking and a query expansion steps can be used to improve the query representation, and how the documents can be precomputed and compressed with Product Quantization. Evaluation on two public datasets shows how this method obtains results beyond the current state-of-the-art methods at a fraction of their cost.
4. We present a method for multi-writer, multi-domain word spotting and word recognition on images. We propose to use character attributes to learn a semantic representation of the word images and then perform a calibration of the scores with Canonical Correlation Analysis that puts images and text strings in a common subspace. After that, spotting and recognition become simple nearest neighbor problems in a very

low dimensional space. To the best of our knowledge, we are the first to provide a unified framework where we can perform out of vocabulary query-by-example and query-by-string retrieval as well as word recognition using the same compact word representations. Evaluation on four public datasets of both handwritten documents and natural images shows results comparable or better than the state-of-the-art on spotting and recognition tasks.

In addition, we want to highlight as a parallel contribution the code that we have publicly released, which implements the major contributions of this thesis [5, 6]. We hope that this would ease the comparison with future methods, and would allow other researchers to extend and improve our research or apply our methods to new tasks.

Part I

Shape Recognition

Chapter 2

A Non-Rigid Appearance Model for Shape Description and Recognition[1]

Abstract

In this paper we describe a framework to learn a model of shape variability in a set of patterns. The framework is based on the Active Appearance Model (AAM) and allows one to combine shape deformations with appearance variability. We have used two modifications of the Blurred Shape Model (BSM) descriptor as basic shape and appearance features to learn the model. These modifications allows to overcome the rigidity of the original BSM, adapting it to the deformations of the shape to be represented. We have applied this framework to representation and classification of handwritten digits and symbols. We show that results of the proposed methodology outperform the original BSM approach.

2.1 Introduction

Objects can be easily interpreted by humans, and their concept can be abstracted despite colors, textures, poses or deformations. A lot of effort has been devoted for many years in order to translate this quality to computers. Thus object recognition has become one of the classic problems in Computer Vision. It is commonly divided in different sub-problems, such as segmentation, feature extraction, object representation, detection or classification, which are tackled with different techniques or from different points of view. In our case, we are interested in two of these problems: techniques related to object representation and, mainly, to feature extraction. In this sense, the description and identification of objects can be done using different visual cues such as shape, color or texture. Among them, shape is probably one of the most widely considered. Anyway, this visual cue is not exempt from problems, and some difficulties such as noise, degradation, occlusions or deformations can be found. Therefore, shape descriptors should be capable to deal with these problems in order to guarantee intra-class compactness and inter-class separability.

Reviewing the literature, many shape descriptors, capable to deal with some of the problems, have been proposed. A survey on shape recognition can be found in [114]. These

[1] This chapter corresponds to the publication "J. Almazán, A. Fornés and E. Valveny, A non-rigid appearance model for shape description and recognition. *Pattern Recognition*, 2012".

descriptors can be broadly classified in two kinds of categories: statistical and structural approaches. Statistical approaches use a feature vector derived from the image to describe the shape. Several examples, using different approaches, may be found into this category. For instance, the curvature scale space (CSS) descriptor [76] uses the external contour for coding the shape. It successively blurs the image by convolving it with a Gaussian kernel, where the scale is increased at each level of blurring. It is tolerant to deformations but it can only be used for closed contours. Zernike moments [54] introduces a set of rotation-invariant features based on the magnitudes of a set of orthogonal complex moments of the image. Scale and rotation invariance are obtained by normalizing the image with respect to these parameters. Another well-known descriptor is Shape Context [13], which is based on the relation between shape pixels. It selects n points from the contour of the shape, and for each of them, computes the distribution of the distance and angle with respect to the other points. It is tolerant to deformations, and is able to deal with open regions. SIFT descriptor [65] uses local information and has been mainly applied for object recognition. It selects local points of interest of the image and describes them in order to provide a "feature description" of the object. The other family of strategies corresponds to structural approaches, which are based on representing the different parts of the shape and also the relation between them using structures, such as strings, grammars or graphs that permit to describe these parts. The comparison between structures is done by means of specific techniques in each case, like graph matching or parsing [17, 61].

In our case, we are interested in shape descriptors that could be applied to Document Analysis applications, mainly in handwritten character recognition and hand-drawn symbol recognition. These are challenging applications for shape descriptors in terms of intra-class compactness and inter-class separability due to the variability of handwriting. Thus, when selecting or designing a good descriptor, the particular characteristics of handwritten symbols have to be taken into account. Mainly because of many kinds of distortions, such as inaccuracy in junctions, missing parts, elastic deformations, overlapping, gaps or errors like over-tracing. Furthermore, depending on the number of writers, the variability between the symbols appearance, caused by the different writing styles, considerably increases. An example of this variability is shown in Figure 2.1. Nowadays, although some techniques have been applied with good results, deformations are still an open problem for descriptors. Shape descriptors mentioned above can be applied, but there are others descriptors that are specific for this domain (a survey on symbol recognition methods can be found in [63]). Among them, the Blurred Shape Model descriptor (BSM) [29], which encodes the spatial probability of appearance of the *shape* pixels and their context information, has shown good results in handwritten symbol recognition tasks. However, the tolerance to large shape deformations is still a challenging problem, and it is mainly caused because of the rigidity of the method's representation.

To deal with deformations methods based on deformable models have been proposed. They have been quite popular within the image segmentation field, and thus the literature on deformable models within this field is large and extensive. They are commonly based on a compromise between external and internal forces, which leads to an iterative energy minimization problem. Probably, the most known approach, and one of the pioneers, is the Active Contours Model (ACM) [51] (also known as "snakes"), which has recently been revisited [75, 77]. For further details and a categorization of this kind of methods we refer the reader to [80]. However, in our case, we are specially interested in methods that have been applied to shape matching, and not only to image segmentation. Thus, following with the energy minimization-based approaches, one example is the *thin-plate splines* (TPS). It is based on a fixed grid that adapts it distribution using an energy minimization function, and has been mainly applied to image alignment and shape matching [20, 92]. In a different way,

(a) (b)

Figure 2.1: Example of the variability caused by different writers for (a) two different music clefs and (b) two symbols from the NicIcon [112] dataset.

we can find in the literature another group of deformable models [12, 19] that uses a Bayesian framework in order to combine prior knowledge of the object and its deformation with the data obtained from the image. Alternatively, methods described by Perronin *et al.* [81], Kuo and Agazzi [55] and Keysers *et al.* [53] are non-linear image deformation models, and are based on pixel matching but applying different constraints on the deformations allowed. And finally, independently of how deformations are modeled, there is a group of methods that try to obtain a model by analyzing the deformations of a shape that are found in the training set. Then, this model is matched with new samples in order to compute a similarity measure for recognition tasks. The deformable part-based model [34] and the Active Appearance Models (AAM) [23] are some examples of this kind of approaches.

In this paper we propose a method for generating statistical models of shape based on the AAM [23] using an adaptation of the BSM descriptor as the basic appearance features. The BSM descriptor resulted a robust technique when classifying symbols with high variability, and it has been applied with success to problems related to hand-drawn symbols. However, due to the rigidity of its grid-based representation, it has an open problem when large deformations may cause high differences in the spatial pixel distribution. For this reason, we have proposed [4, 9] an extension of this descriptor by integrating it with a deformation model. First, we modify the BSM grid-based representation, to provide more flexibility, and make it deformable. Then, we apply a deformation procedure in order to adapt it to the shape to be described: a non-linear deformation model [9] and a *region partitioning procedure* by computing geometrical centroids [4]. These new resulting descriptors are capable to deal with large deformations due to their adaptive representation. Moreover, they allow us to extract information related to the *shape pixels* distribution and the structure of the shape. The structure of the shape is captured in both deformation processes through the final state of the descriptor after the deformation. However, the final description is encoded as a single feature vector. Therefore, according to the categorization of shape descriptors done by [114], our descriptor would fall into the category of statistical approaches.

Then, based on these new descriptors, the main contribution of this paper is the proposal of a non-rigid model able to learn patterns of variability. This is performed by combining both modified versions of the BSM descriptor, independently, with the AAM [23] for learning the variability. It will result in a combined model that matches shape pixel distribution variations and structure variations. Moreover, this model will be integrated in two different classification schemes proposed for shape recognition tasks. Results show that the proposed methodology outperforms the original BSM approach.

The rest of the paper is organized as follows: Section 2.2 is devoted to explain the new proposed descriptor, while Section 2.3 explains the process to build the model of appearance. The explanation of the classification schemes is conducted in Section 2.4. Then, performance results, as well as the comparison with the original BSM, are shown in Section 5.6. Finally, Section 5.7 concludes the paper.

2.2 Adaptive Blurred Shape Model

Our proposed deformable shape descriptor results from the adaptation of the Blurred Shape Model (BSM) [29] with a process of deformation. We use two different deformation approaches: a non-linear deformation model (the Image Distortion Model (IDM) [53]), and a *region partitioning* procedure by geometrical centroid estimation (based on the Adaptive Hierarchical Density Histogram (AHDH) [105]). Both deformation processes are very fast to compute compared to others in the literature, such as the energy minimization-based (TPS [20]) or the Bayesian-based [19]. The main disadvantage of these methods is that they consist in an iterative process extended until convergence, which makes them computationally costly. On the contrary, the IDM and the AHDH deformation processes let us accomplish our purpose of adapting the descriptor to the shape in a suitable and a very efficient way.

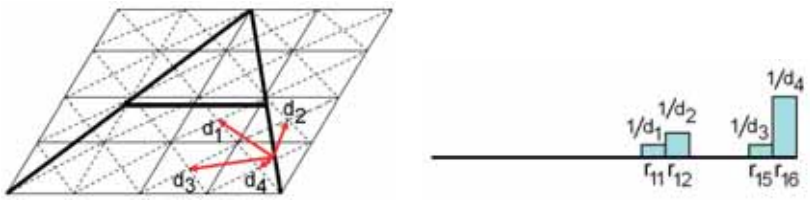
So, applying these deformation processes, our objective is to encode the pixel distribution of a given image by first adapting the structure of the descriptor to the shape and then computing the pixel density measure using the BSM feature extraction procedure. Therefore, the first step is to modify the original grid-based representation of the BSM (Section 2.2.1) into a flexible *focus-based* representation (Section 2.2.2). Then, we will integrate the IDM and the *region partitioning* procedure in order to deform this new structure (Section 2.2.3 and Section 2.2.4 respectively).

2.2.1 Blurred Shape Model

The main idea of the BSM descriptor [29] is to describe a given shape by a probability density function encoding the probability of pixel densities of a certain number of image sub-regions. Given a set of points forming the shape of a particular symbol, each point contributes to compute the BSM descriptor. This is done by dividing the given image in a $n \times n$ grid with equal-sized sub-regions (cells). Then, each cell receives votes from the *shape pixels* located inside its corresponding cell, but also from those located in the adjacent cells. Thereby, every pixel contributes to the density measure of its sub-region cell, and its neighboring ones. This contribution is weighted according to the distance between the point and the centroid of the cell receiving the vote. In Fig. 2.2 an example of the contribution for a given pixel is shown. The output is a vector histogram, where each position contains the accumulated value of each sub-region, and contains the spatial distribution in the context of the sub-region and its neighbors.

2.2.2 Focus representation

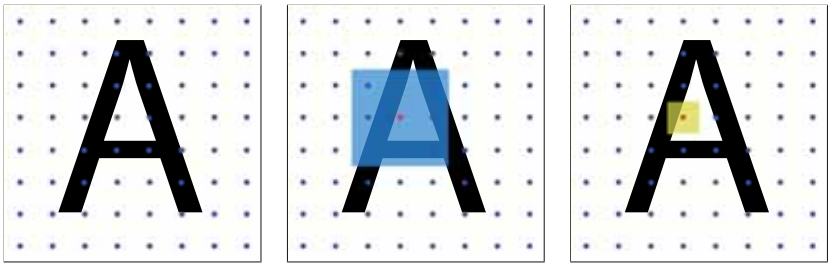
As it has been explained, BSM is based on placing a fixed regular grid over the image. Therefore, in order to allow deformations of the grid we must adopt a slightly different representation. Instead of a regular grid of size $k \times k$ we will place over the image a set of $k \times k$ points, equidistantly distributed. These points, denoted as *focuses*, will correspond to the centroids of the original regular grid and, as in the original approach, will accumulate votes of the neighboring pixels weighted by their distance. Concretely, the contribution of a pixel p to a focus f will be equal to $\frac{1}{d(p,f)}$, where d is the euclidean distance. However,



(a) (b)

Figure 2.2: BSM density estimation example. (a) Distances of a given shape pixel to the neighboring centroids. (b) Vector descriptor update in regions r using distances of (a).

instead of defining the neighborhood as a set of fixed cells of the grid, it will be defined as an arbitrary *influence area* centered on the focus, in order to provide flexibility. The deformation of the grid will be obtained by moving independently each of the focuses along with their respective influence area. In order to limit the amount of deformation, each focus will be allowed to move only inside a pre-defined *deformation area*. In Fig. 2.3 we show an example of the focus representation and their influence and deformation areas. This resulting representation provides more flexibility and allows the focus deformation tracking.



(a) (b) (c)

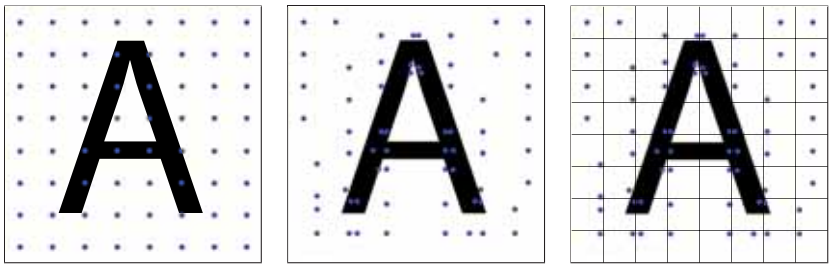
Figure 2.3: (a) Focuses representation. (b) Influence area. (c) Deformation area.

2.2.3 Focus deformation by non-linear deformation model

Using this new representation of $k \times k$ equidistantly distributed focuses (Figure 2.4(a)), the adaptation of the original Image Distortion Model (IDM) [53] is relatively straightforward. The ***non-linear deformation process*** of the IDM consists in matching independently every pixel in a *test image* to a new location following a given criterion. The criterion used by [53] is to minimize the matching difference (using its context) with the pixels in a *reference image*. The amount of deformation (*i.e.*, the displacement) allowed for every pixel of the test image is limited by a fixed *deformation area*. In an analog way, we will also move independently every focus inside their own defined *deformation area*, but following a different criterion than in the original approach. Considering that our objective is to adapt the focuses distribution to the shape to be described (Figure 2.4(b)), a suitable criterion will be to maximize the density, around the focus, of pixels belonging to the shape (*shape pixels*). This can be

achieved by maximizing the *BSM value* accumulated in the focus, which is only computed from *shape pixels* inside the *influence area*. Note that this *influence area* moves along with the focus, so the focus will have a different value depending on its position. Thus, for a given image, every focus will be moved independently inside the *deformation area* to maximize the accumulated BSM value (*i.e.*, the density of *shape pixels* around the focus) (Figure 2.4(c)). Figure 2.4 shows an example of this process. As a result, a given shape will be represented with two output descriptors:

- A vector histogram $\mathbf{t} \in \mathbb{R}^{k^2}$, which contains the density measure of nearby pixels of each focus.
- A vector $\mathbf{s} \in \mathbb{R}^{2k^2}$, which contains x and y coordinates of each focus, which are normalized by the width and height of the image, respectively, in order to be scale invariant.



(a) (b) (c)

Figure 2.4: Example of the focuses deformation. (a) Initial position of the focuses. (b) Final position of the focuses after the maximization of their values. (c) Deformation area used.

For now on, we will name this shape descriptor, resulting from the integration of the BSM with the IDM, as the Deformable Blurred Shape Model (DBSM).

2.2.4 Focus deformation by region partitioning

The DBSM descriptor unifies in a single procedure the deformation of the focuses and the computation of the pixel density measure around them. Now, we propose a different approach to compute these two processes in two independent steps. However, in an analog way, this new approach follows the same idea of the DBSM: focuses will be distributed over the image in regions containing a high pixel density in order to adapt them to the structure of the shape. This new approach is based on the ***region partitioning*** procedure of the Adaptive Hierarchical Density Histogram (AHDH) [105], which consists in iteratively producing regions of the image using the geometrical centroid estimation. The coordinates of the *focuses* will be the position of these geometrical centroids.

First, we consider the binary image as a distribution of *shape pixels* in a two-dimensional space-background (Figure 2.5(a)). The set of shape pixels is defined as S and their number as N. Furthermore, we define as R_i^l, $i = \{1, 2, \ldots, 4^l\}$ the i-th rectangular region obtained in the iteration (or 'level') l of the partitioning algorithm, and as $F^l \in \mathbb{R}^2$ the set of geometrical centroids of the regions in R^l. For each level l, the ***region partitioning*** procedure estimates the geometric centroid of all regions R_i^l and then splits each region into four sub-regions using

as a center the geometric centroid. The new sub-regions generated will form the new set of regions R^{l+1} .The initial region, R^0, is the whole image, and F^0 would contain the geometrical centroid of this region (Figure 2.5(b)). Considering a separate cartesian coordinates system for each region R_i^l, the geometrical centroid F_i^l is computed using equations

$$\mathbf{x_c} = \frac{\sum_{(x,y)\in S_i^l} \mathbf{x}}{\mathbf{N_i^l}},\ \mathbf{y_c} = \frac{\sum_{(x,y)\in S_i^l} \mathbf{y}}{\mathbf{N_i^l}}, \tag{2.1}$$

where N_i^l denotes the number of shape pixels set S_i^l in the processed region R_i^l, and $(\mathbf{x}, \mathbf{y})$ are the pixel coordinates. This iterative procedure finishes when a termination level L is reached. Then, the final coordinates of the *focuses* will be the geometrical centroids computed in the level L, that is F^L. Thus, the number of focuses to represent the shape (4^L) can be determined using this termination level L. An example of the distribution of focuses for different levels is shown in Figure 2.5.

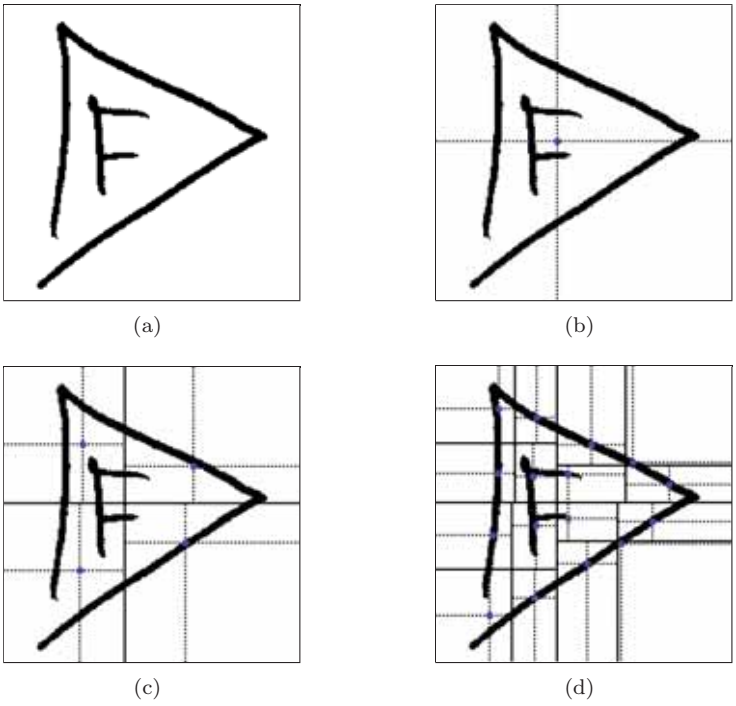


Figure 2.5: Focuses distribution computation based on the region partitioning algorithm: (a) original image, (b), (c) and (d) focuses (in blue) at level 0, 1 and 2 respectively.

Once the vector $\mathbf{s} \in \mathbb{R}^{2\times 4^L}$ containing the position of the focuses for a given shape is obtained, we compute the vector histogram $\mathbf{t} \in \mathbb{R}^{4^L}$, which contains the density measure of nearby pixels of each focus. It is done in a similar way to the DBSM case: previously mentioned $h \times w$ *influence area* is used to calculate the pixel density around each focus. Focuses will receive votes from neighboring shape pixels, which are those inside this *influence area*. Based on the BSM [29], this vote is weighted according to the distance between the

pixel and the focus. Finally, x and y coordinates of the position of the focus in vector $\mathbf{s}$ are normalized by the width and height of the image, respectively, in order to achieve scale invariance. In the following, this method will be denoted as Non-Rigid Blurred Shape Model (nrBSM).

2.3 Non-rigid Appearance Model

The deformable extensions of the Blurred Shape Model (DBSM and nrBSM) can be seen as descriptors that extract information related to the structure and the texture of a shape. The information related to the structure of the shape can be obtained from the deformation that each focus has suffered (in terms of location). And, the BSM value of the focuses, that is the pixel density measure around them, can be seen as a texture-related feature. Using these information extracted from the DBSM or the nrBSM, we will generate statistical models by learning patterns of variability from the training set, based on the Active Appearance Model (AAM) [23]. It results in a model for structure variation and a model for texture variation. Moreover, after capturing the variability in *structure* and *texture* of the shape independently, we are going to generate a final *model of appearance*. This statistical model of appearance matches variations of the structure and texture simultaneously by combining their respective statistical models of variation.

2.3.1 Learning patterns of variability

In this section we are going to detail the process of building a combined model of variation, which is based on the method developed by T.F. Cootes *et al.* [23]. In order to build the model, first, they require a training set of annotated images where corresponding points have been marked on each example. However, in our case, this pre-process is automatically done with the focus-based representation of the Adaptive BSM (*i.e.*, DBSM or nrBSM): using the same number of $k \times k$ focuses for all the images in the training set we can use their own correspondence to track the variability and build the statistical model. Moreover, the Adaptive BSM also extracts both kind of features necessary to build the combined model.

Once the Adaptive BSM is computed in all the images of the training set, we obtain two output vectors for every image: a vector $\mathbf{s} \in \mathbb{R}^{2k^2}$ containing the final coordinates of the focuses, and a vector $\mathbf{t} \in \mathbb{R}^{k^2}$ containing the density measure of pixels around each focus. With these two vectors we are going to build two different statistical models by learning the variability of the deformations in the focuses positions (related to the structure of the shape) and the variability in the pixel density (related to the texture). This is done by first constructing two different matrices with $\mathbf{s}$ and $\mathbf{t}$ vectors and applying principal component analysis (PCA) to both matrices, resulting in a *structure model* and a *texture model*. A property of these models is that they make possible the reconstruction of the shape and texture information of the training images using

$$\begin{aligned} \mathbf{s} &= \bar{\mathbf{s}} + \mathbf{Q}_\mathbf{s}\mathbf{b}_\mathbf{s} \\ \mathbf{t} &= \bar{\mathbf{t}} + \mathbf{Q}_\mathbf{t}\mathbf{b}_\mathbf{t}, \end{aligned} \tag{2.2}$$

where $\bar{\mathbf{s}}$ is the mean *structure*, $\bar{\mathbf{t}}$ the mean *texture information*, $\mathbf{Q}_\mathbf{s}$, $\mathbf{Q}_\mathbf{t}$ are the matrices of eigenvectors that describe the modes of variation derived from the training set, and $\mathbf{b}_\mathbf{s}$, $\mathbf{b}_\mathbf{t}$ are the vectors of weights that represent *structure* and *texture*, respectively. Vectors $\mathbf{b}_\mathbf{s}$ and $\mathbf{b}_\mathbf{t}$ can be seen as the parameters of the model, or the representation of descriptors $\mathbf{s}$ and $\mathbf{t}$ in the PCA space.

Using this property, the following step consists in learning correlations between *structure* and *texture* using their respective models. We obtain the representation in the PCA space of structure and texture of all training images using

$$\begin{aligned} \mathbf{b_s} &= \mathbf{Q_s}^T(\mathbf{s} - \bar{\mathbf{s}}) \\ \mathbf{b_t} &= \mathbf{Q_t}^T(\mathbf{t} - \bar{\mathbf{t}}). \end{aligned} \tag{2.3}$$

The result is that every image in the training set is represented by a vector containing *structure* model parameters and a vector containing *texture* model parameters ($\mathbf{b_s}$ and $\mathbf{b_t}$ weight vectors). Then, the final step consists in concatenating both vectors of every sample image in a single vector $\mathbf{c}$, construct a new matrix, and apply PCA again, extracting the combined modes of variation. In order to give *structure* and *texture* variation approximately equal significance, before applying PCA we scale *structure* parameters so their cumulative variance within the training set is equal to the cumulative variance of the *texture* parameters. The resulting appearance model has parameters, $\mathbf{b_a}$, controlling *structure* and *texture* models parameters (which control *structure* and *texture* descriptions) according to

$$\mathbf{a} = \bar{\mathbf{a}} + \mathbf{Q_a}\mathbf{b_a}, \tag{2.4}$$

where $\mathbf{a}$ is the concatenation of *structure* and *texture* models parameters, $\bar{\mathbf{a}}$ is the mean appearance, and $\mathbf{Q_a}$ is the matrix of eigenvectors that describes the modes of variation of the appearance. Finally, the vector of $\mathbf{b_a}$ can be seen as the feature vector that represents an image in the combined *model of appearance*. Given a model of appearance, it can be computed using

$$\mathbf{b_a} = \mathbf{Q_a}^T(\mathbf{a} - \bar{\mathbf{a}}). \tag{2.5}$$

2.4 Classification

The Non-Rigid Appearance Model (NRAM) generates statistical models of appearance, which combines *structure* and *texture* variations learned from a training set. Therefore, we can generate a model that represents independently every different class in the dataset. We have designed two different classification schemes using the Non-Rigid Appearance Model for shape recognition tasks. On one hand, a scheme based on the ability of the appearance model to generate "synthetic" representations of a given shape. On the other hand, a scheme using the parameters of the model, *i.e.*, the descriptors represented in the PCA space, to train a Support Vector Machine (SVM) for each class.

2.4.1 Distance to the model

The representation of the shape obtained with the model in the PCA space can be used to obtain a reconstruction of the shape in the original space. This reconstruction will reflect the utility of the model to represent the shape. So, it is expected that for shapes belonging to the class it will be similar. Therefore, we can use this property to, given a new image and its respective *structure* and *shape* feature vectors, generate a synthetic sample with a model of a given class that matches it as closely as possible and design a measure of similarity. We have integrated it into a matching process for shape classification. It consists in, given an image I and an appearance model M, first computing the structure $\mathbf{s_I}$ and texture $\mathbf{t_I}$ descriptors of that image. Then we approximate these descriptors to the corresponding parameters of the structure and texture model of M using the expressions in Equation 2.3, resulting in two

new vectors $\mathbf{b_{s_I}}$ and $\mathbf{b_{t_I}}$. Following, we concatenate them using the normalization learned in the training step to make equal both contributions. And then we approximate again this new vector $\mathbf{a_I}$ to the model of combined appearance in M using Equation 2.5.

Finally, the resulting parameters of the appearance model, $\mathbf{b_{a_I}}$, are split in $\mathbf{b_{s_J}}$ and $\mathbf{b_{t_J}}$ in order to generate the new synthetic descriptors $\mathbf{s_J}$ and $\mathbf{t_J}$ by back-projecting them with the models of structure and texture, respectively. These are the descriptors that best match to I according to the appearance model M. Thus, we can use this new descriptors of structure, $\mathbf{s_J}$, and texture, $\mathbf{t_J}$, to compute a distance with the descriptors of the original image I. For this purpose, we use the euclidean distance between each corresponding vector. Furthermore, we want to add some information about the necessary deformation applied to adjust the model to the test image. For that end, we also compute the euclidean distance between the generated descriptors and the mean values of the model, both for structure $\bar{\mathbf{s}}$ and texture $\bar{\mathbf{t}}$. Then, the final distances are

$$\begin{aligned} d_s &= dist(\mathbf{s_I}, \mathbf{s_J}) + \beta \cdot dist(\mathbf{s_J}, \bar{\mathbf{s}}) \\ d_t &= dist(\mathbf{t_I}, \mathbf{t_J}) + \beta \cdot dist(\mathbf{t_J}, \bar{\mathbf{t}}) \end{aligned} \tag{2.6}$$

where β is the factor that weights the contribution of the information of deformation in the final distance, and d_s and d_t structure and texture distances, respectively. Finally, the structure and texture measures of similarity are combined using θ as another factor of contribution, resulting in the following expression

$$d_a = d_s \cdot \theta + d_t \cdot (1 - \theta) \tag{2.7}$$

This distance can be used for classification tasks, being applied to, for example, a nearest neighbor classifier: given a test image, and a set of models representing shape classes, we assign the image to the class which results in the minimum distance from the representation synthetically generated.

2.4.2 Support Vector Machine-based scheme

The second scheme we propose uses the representation in the space of the appearance model space, *i.e.*, the vector of weights $\mathbf{b_a}$. It is used as a feature vector to describe the shapes contained in the dataset to train a different Support Vector Machine for each class.

In the training step (Figure 2.6) we compute first *structure* and *texture* descriptors, $\mathbf{s}$ and $\mathbf{t}$, and we generate a Non-Rigid Appearance Model M_i for each one of the n classes in the dataset using the procedure explained in Section 2.3.1. Then, we train a binary Support Vector Machine for each class. This is, for class i, we use as positive samples those training samples belonging to class i, projected in the appearance model space of the model M_i. And as negative samples the rest of the training set (*i.e.*, those which do not belong to class i) also projected with model M_i.

Then, given a test sample, it is projected in the PCA space of all the appearance models of all the classes. And then, the score is computed with all the SVMs, using their corresponding vector. Each score is normalized [28] by subtracting the mean and then divided by the average score norm computed for each SVM. Finally, the test sample is assigned to the class which results in the highest score. A scheme of the process is shown in Figure 2.7.

2.5 Experiments

In this section we are going to show the performance of the proposed Non-Rigid Appearance Model for shape recognition tasks using two different datasets.

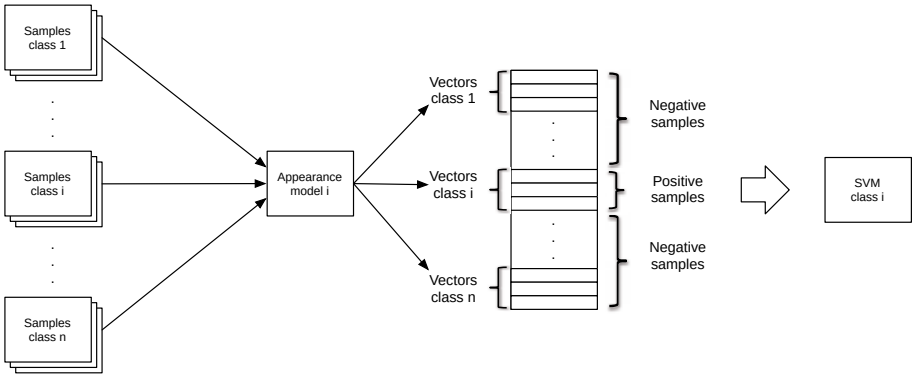


Figure 2.6: Training procedure for the SVM-based scheme.

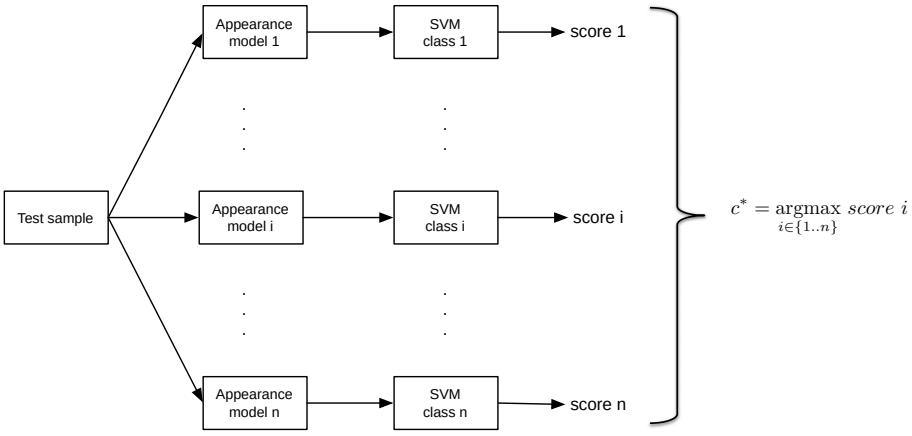


Figure 2.7: Test procedure for the SVM-based scheme.

2.5.1 Datasets

We have tested our methods for shape recognition tasks, and for this purpose, we have used the MNIST and NicIcon datasets. Following, we describe these datasets as well as the experimental protocol used in each one.

MNIST. The MNIST [58] (Figure 2.8) is a database of handwritten digits from different writers and it is divided in a training set of $60,000$ examples, and a test set of $10,000$ examples. The digit size is normalized and centered in a fixed-size image of 28×28 pixels. We have re-centered the digits by their bounding box, as it is reported in [58] to improve error rates when classification methods like SVM or K-nearest neighbors are applied. This dataset has been commonly used in learning techniques and pattern recognition methods.

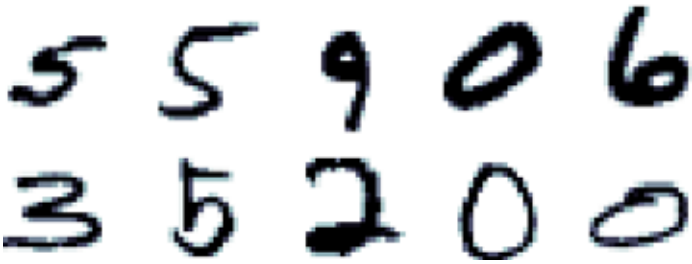


Figure 2.8: Digit samples of MNIST dataset.

NicIcon. The NicIcon dataset [112] (Figure 2.9) is composed of 26,163 handwritten symbols of 14 classes from 34 different writers and it is commonly used for on-line symbol recognition, but off-line data is also available. The dataset is already divided in three subsets (training, validation and test) for both *writer dependent* and *independent* settings. Approximately, and depending on the setting, $9,300$, $6,200$ and $10,700$ symbols are contained in the training, validation and test sets, respectively. We have selected the off-line data with both configurations as a benchmark to test our method. It is worth to mention that off-line data is presented as scanned forms where writers where said to draw the symbols. So first, we have extracted individually every symbol from the scanned forms, and then binarized and scale-normalized in an image of 256×256 pixels.

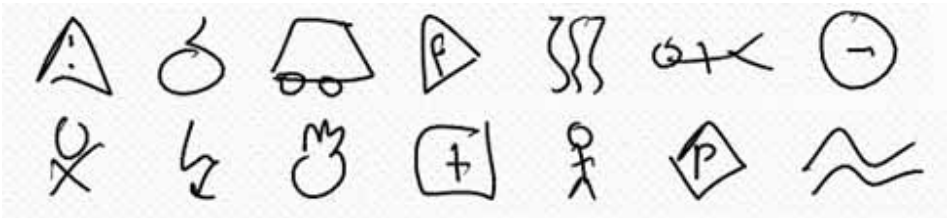


Figure 2.9: Samples of the 14 different classes of the NicIcon dataset.

Finally, it is worth to mention that, in order to be able to compare our results with the state of the art, we have followed the common protocols for both datasets. That is, we have used all the elements in the sets for training, validating (only in the NicIcon) and testing as they are provided originally by their authors.

2.5.2 Results

We now show the benefits of the proposed method using the datasets introduced in Section 2.5.1. We test our Non-Rigid Appearance Model (NRAM) (Section 2.3) by learning the variability using as basic features both Adaptive Blurred Shape Model descriptors proposed, the **DBSM** and the **nrBSM** (Section 2.2.3 and Section 2.2.4 respectively). It results in two different configurations: NRAM+DBSM and NRAM+nrBSM. We apply them for shape recognition tasks using both classification schemes proposed in Section 2.4.

Table 2.1: Accuracy rate (%) comparison of the Non-Rigid Appearance Model (NRAM) (in combination with the DBSM and the nrBSM) with the original BSM and the DBSM and nrBSM, using a NN classifier.

Method		BSM [29]	DBSM	nrBSM	NRAM+DBSM	NRAM+nrBSM
MNIST		92.65	94.39	94.78	89.29	94.65
NicIcon	WD	93.73	95.45	95.37	90.88	97.70
	WI	90.02	90.29	91.09	86.35	95.18

First, we report in Table 2.1 results over the nearest neighbor classifier using the distance to the model (Section 2.4.1). We compare the performance with the original approach, the BSM [29], and also with the DBSM and nrBSM descriptor without applying the Non-Rigid Appearance Model. We can appreciate that, while the combination of the NRAM with the DBSM (NRAM+DBSM) features results in a lower performance, the appearance models obtained with the nrBSM features (NRAM+nrBSM) outperform the rest of approaches. These results lead us to conclude that the NRAM methodology is not able to learn the variation models in a suitable way when structure and texture features are extracted using the DBSM. However, the validity of the model is shown when we use features extracted from the nrBSM, where accuracy increase considerably compared to the situation where we do not apply the NRAM. This difference in performance is due to the deformation procedure. Analyzing the focuses distribution, we can appreciate that, in the case of the nrBSM, focuses distribute along the whole shape, which is contrary to the DBSM case. This is because the nrBSM is not limited by a pre-defined initial position of the focuses or a fixed *deformation area*, while the DBSM adaptability is affected by both factors. Thus, small deformation areas lead the focuses to stay close to their initial position, while large areas make that all the focuses converge to the same location. Therefore, the better adaptability of the nrBSM makes it more suitable to learn the variability, and the variation models obtained with these features are more representative. The NRAM benefits from this fact, and results in a better performance compared to the DBSM features. Finally, note that, in all the cases, DBSM and nrBSM descriptors outperform the original BSM. This shows that the integration of deformations to the fixed grid-based representation leads to a better performance when large shape distortions are present.

Regarding the SVM classification scheme, the results are shown in Table 2.2. We can appreciate that performance increases for both descriptors, being remarkable in the cases of the MNIST and *writer dependent* NicIcon datasets. Note that results obtained over the MNIST do not reach the state of the art [58]. This is mainly due to the fact that the original BSM descriptor has not been specifically designed for the task of handwritten character recognition. However, note also that results with the proposed model are much better than results with the original BSM descriptor. Thus, it is expected that combining the NRAM methodology with a specific feature extraction method for character could result

in a competitive performance, comparable to the current state of the art.

Table 2.2: Results of the Non-Rigid Appearance Model combined with the DBSM and the nrBSM using the SVM classification scheme.

Method		NRAM+DBSM	NRAM+nrBSM
MNIST		97.76	97.50
NicIcon	WD	96.52	97.35
	WI	93.26	94.29

Concerning the NicIcon dataset, the state of the art, which only exists for on-line data, achieves 98.57% and 92.63% of accuracy rate in classification, using a SVM, for WD and WI, respectively [112]. Comparatively, we see that the best recognition rate that we obtain in our approach is slightly below in the case of WD configuration, but higher for the WI. Furthermore, we only use off-line data, which makes the problem much more difficult. Thus, we can consider the obtained results very competitive. It is also remarkable the significant increase of performance in relation to the original BSM approach. Moreover, note that we obtain a high accuracy in the difficult WI configuration, where the training set does not contain samples from writers that appear in the test set and vice versa. These facts reinforce the idea that the NRAM combined with the nrBSM representation leads to a good representation of the shape, tolerant to large variations and different writing styles.

2.5.3 Parameter selection

Our Adaptive BSM descriptors (*i.e.* DBSM and nrBSM) have two parameters (leaving aside the *deformation area* of the DBSM) to be adjusted: the number of $k \times k$ focuses (defined by termination level L for the nrBSM), and the $h \times w$ size of the *influence area*. The influence area is defined as a rectangular region where height h and width w are adjusted wrt k and the height and width of the image using following equations

$$h = \alpha * \frac{H}{k}, \;\; w = \alpha * \frac{W}{k}. \tag{2.8}$$

In order to select the best α, which controls the size of the influence area, we need to reach a trade-off between the *locality* and the *globality* of the encoded information. With large influence areas, each focus captures more global information than using small influence areas. Experimentally, we appreciate that using small influence areas performs better in the combination of the NRAM with the nrBSM descriptor. This is due to the fact that focuses are well distributed over the whole shape, and we can analyze the pixel distribution variability locally for each region of the shape. The best performance for both datasets has been obtained for values of α around 1. Regarding the number of focuses k, it depends on the size of the image, and its adjustment is a compromise between performance and dimensionality. Experimentally, we see that accuracy becomes stable for a certain number, and a higher number of focuses does not contribute to a significant improvement in the performance. We have set k equal to 16 for the MNIST dataset, and equal to 32 for the NicIcon dataset.

2.6 Conclusions

In this paper, a method for modeling the appearance deformations of a shape by learning the variability of the training set is described. It is developed on the top of two recently introduced adaptive shape descriptors based on the BSM. These descriptors are used as appearance features to build the statistical model. Then we describe two classification schemes to integrate the appearance models in shape recognition tasks. The experimental performance evaluation shows the ability of the appearance models to learn *structure* and *texture* variability, achieving a satisfactory performance in shape recognition. Additionally, results also show the capacity of both novel Adaptive Blurred Shape Model descriptors to capture the structure of the shape and deal with large deformations, outperforming the rigid grid-based approach of the original BSM. In this work, the BSM has been used as the basic descriptor to build the model. However, the non-rigid appearance model introduces a perfect framework that can be used with a large number of different appearance features, which may be selected depending on the application. Moreover, the methodology can be easily extended for a larger number of models, where the combination can be done at different levels or following different criteria.

Chapter 3

Deformable HOG-based Shape Descriptor[1]

Abstract

In this paper we deal with the problem of recognizing handwritten shapes. We present a new deformable feature extraction method that adapts to the shape to be described, dealing in this way with the variability introduced in the handwriting domain. It consists in a selection of regions that best define the shape to be described, followed by the computation of histograms of oriented gradients-based features over these points. Our results significantly outperform other descriptors in the literature for the task of hand-drawn shape recognition and handwritten word retrieval.

3.1 Introduction

A lot of effort has been dedicated in Computer Vision and Pattern Recognition to the problem of shape recognition. It is at the core of many different applications, such as object retrieval or sketch recognition. In the Document Analysis domain, the case of hand-drawn is a specially challenging problem since we have to deal with large variability coming from noise, distortions, inaccuracy in strokes and changes caused by different writing styles. The extraction of robust features is a critical point in this case. Thus, descriptors able to adapt to all this variability are necessary.

In the literature many different feature extraction methods have been proposed for shape description. We are interested in descriptors that have been applied to the recognition of shapes written or drawn by hand. Several generic shapes descriptors have been applied to this kind of shapes, and a general overview can be found in [114]. Among them, we can highlight the curvature scale space (CSS) descriptor [76] which successively blurs the shape contour by convolving it with a Gaussian kernel, the Shape Context [13], which selects n points from the contour of the shape and computes the distribution of the distance and angle between them, or the radon-based method proposed in [100], which uses DTW to match corresponding pairs

[1] This chapter corresponds to the publication "J. Almazán, A. Fornés and E. Valveny. Deformable HOG-based Shape Descriptor. In *International Conference on Document Analysis and Recognition*, 2013".

of radon histograms at every projecting angle. These descriptors are robust to deformations, however, wether they can only deal with some specific shapes or they are computationally expensive, they can not be applied to all the tasks in the "handwritten" domain. Specially conceived for the specific case of hand-drawn symbol recognition, the Blurred Shape Model (BSM) [29] has shown to obtain good results in hand-drawing applications. It is based on computing the spatial distribution of shape pixels in a set of pre-defined image sub-regions and is able to handle a certain degree of deformation. However, due to the rigidity of the model, large deformations cause large differences in the spatial information encoded by the BSM.

In order to overcome the rigidity of the BSM, the cmiBSM feature extraction method [4] was presented as an extension of the BSM improving its robustness against large deformations. It consists in substituting the fixed regular grid of the BSM by a more flexible grid. A region partition algorithm adapts a given number of points to the shape to be described, and then the "pixels density" is computed in each one of them by the accumulation of *shape pixels*, just as the BSM does. This approach showed a good performance for recognizing shapes in difficult problems such as writer independent symbol recognition. However, when dealing with fine details, *e.g.* recognizing skilled forgery signatures, it presented some difficulties. This is mainly due to the simplicity of the features extracted. We argue that the intensity of foreground pixels is insufficient to capture all the fine-grained details.

Another descriptor that has been recently applied to handwritten shapes [7] with excellent results is the well-known Histogram of Oriented Gradients (HOG) [26]. HOG takes the pixel gradient information as the basis to extract features, which has been shown to be able to deal with fine-grained details and to capture more information than other kind of features, such as the "pixels density". It consists in dividing the image in a rigid grid of cells and computing a histogram of gradients in each one of them. Therefore, apart from the basis features, HOG is similar to the BSM in the sense of using a grid and computing a histogram in each cell. Thus, we argue that the main issues of this descriptor with hand-drawn shapes, as it happens to the BSM, come from his rigidity: allowing some deformation will let us focus on the most discriminative areas, *i.e.*, those that best define the shape. Commonly, handwritten shapes are composed of regions without meaningful information, and on the other hand, regions where all the information is concentrated. Thus, descriptors should focus mainly on these meaningful regions. Therefore, in this paper we propose to combine the deformable grid scheme of the cmiBSM approach with HOG-based features. In this way we plan to improve the HOG descriptor in order to focus the description on the most discriminative regions of the shape.

The main contribution of this work is the extension of the HOG descriptor for the specific case of handwritting, combining gradient features and a flexible and adaptable grid. We use the *region partitioning* algorithm for the detection of shape regions where information is concentrated in combination with HOG, a feature extraction method able to capture fine and discriminative details. In this sense, we will show that gradient-based features performs better with hand-drawn symbols than density-based features encoded by the BSM, and that the flexibility of the deformable grid improves the results of the rigid grid that the HOG uses.

Finally, we will show that the new descriptor can solve one of the common problems (also related to the rigid grid) encountered when applying the HOG descriptor to images that have different aspect ratios. In order to compare two images using HOG, both should have the same size, otherwise, the dimension of the feature vector may result different. This makes a warping to a fixed image size necessary, which even deforms the shape contained or adds background space without meaningful information. This also provokes that corresponding HOG cells may not contain the same regions of the shape, so it will negatively affect the

matching process. However, the approach that we propose, as a side effect of combining the HOG descriptor with the deformable grid, is able to deal with changes in the aspect ratio of the images. That is, as a result of the region partition algorithm, focuses will be located in similar regions of the shape independently of the aspect ratio of the image.

The ability of our method for the description of handwritten shapes has been evaluated for two different, but related, tasks: hand-drawn symbol recognition and handwritten word retrieval. In the latter we consider the handwritten word as a shape to be described and retrieved from the dataset, so it is not related with the typical word spotting approach. Both tasks will test the feature extraction method against writer independent configurations and also against images with different scales and aspect ratios.

The rest of the paper is organized as follows: Section II describes the method proposed. The explanation of the experiments, including the datasets used and the experimental protocols is conducted in Section III. Then, Section IV is devoted to show performance results, as well as the comparison with other approaches. Finally, Section V concludes the paper and proposes a future work line.

3.2 Deformable HOG

The deformable HOG-based feature extraction approach is based on the computation of HOG features in a given set of $k \times k$ points, denoted as *focuses*, over the shape to be described. These focuses, which can also be seen as an adaptable mesh, are automatically positioned with the objective of being distributed along the shape pixels. Therefore, this approach can be divided in two sequential steps: a first step devoted to compute the location of the focuses following an *iterative region partitioning* algorithm [105] and a second step where regions centered over the focuses are *extracted and described* using HOG features.

3.2.1 HOG Features

HOG descriptor was first introduced by Dalal and Triggs [26], but we use Felzenszwalb *et al.* implementation [33], which includes some improvements over the original approach. It consists in first computing for every pixel in the image the orientation and the magnitude of the intensity gradient. Then, the image is divided in an uniform grid of cells and for each one of them a histogram of gradients is computed using "soft binning". Finally, a dimensionality reduction is performed, resulting in a 31-dimensional vector for each cell: 27 dimensions corresponding to different orientation channels (9 contrast insensitive and 18 contrast sensitive), and 4 dimensions capturing the overall gradient energy in square blocks of four cells around. An example of the HOG features extracted from two different words can be seen in Figure 3.1. As we can see, these gradient-based features contains enough discriminative and fine-grained information to be able to differentiate between both words.

3.2.2 Region Partitioning Procedure

The **region partitioning** procedure consists in subdiving the image into regions centered on the geometrical centroid of the corresponding region of the previous level. The location coordinates of the resulting geometrical centroids will be the points, denoted as *focuses*, where features will be following extracted. Next, we give a brief description of this procedure in order to introduce some notation. For further details, we refer the reader to [105], where this procedure was originally proposed, and [4], where was first used as an adaptable mesh for the extraction of features.

Figure 3.1: HOG features.

We denote the set of *shape pixels* of the binary image as S and their number as N. The region partitioning procedure will work by obtaining a series of subregions of the image at successive levels. Furthermore, we define as $R_i^l,\ i = \{1, 2, \ldots, 4^l\}$ the i-th rectangular region obtained in the iteration (or 'level') l of the partitioning algorithm, and as $F^l \in \mathbb{R}^2$ the set of geometrical centroids of the regions in R^l. For each level l, the *region partitioning* procedure estimates the geometric centroid of all regions R_i^l and then splits each region into four sub-regions using the geometric centroid. The new sub-regions generated will form the new set of regions R^{l+1}. We consider R^0 as the whole image, and F^0 to contain the geometrical centroid of this region (Figure 3.2(a)).

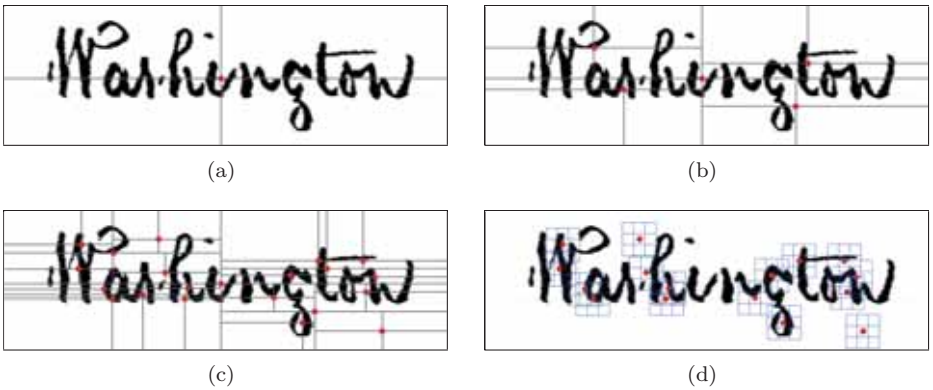


Figure 3.2: (a-c) Regions and focuses resulted in the *region partition* procedure for different levels with L equal to 0, 1 and 2. (d) HOG features extracted from level L equal to 2 using a 3×3 cells grid.

Considering a separate cartesian coordinates system for each region R_i^l, the geometrical centroid F_i^l is computed using equations

$$\mathbf{x_c} = \frac{\sum_{(x,y)\in S_i^l} \mathbf{x}}{\mathbf{N_i^l}},\ \mathbf{y_c} = \frac{\sum_{(x,y)\in S_i^l} \mathbf{y}}{\mathbf{N_i^l}}, \tag{3.1}$$

where N_i^l denotes the number of shape pixels set S_i^l in the processed region R_i^l, and $\mathbf{x}$, $\mathbf{y}$ are the pixel coordinates. This iterative procedure finishes when a termination level L is reached. Then, the final coordinates of the *focuses* will be only the geometrical centroids of the level L, *i.e.*, F^L. Thus, the number of focuses to represent the shape 4^L can be determined using

(a) (b)

Figure 3.3: (a) NicIcon dataset for shape recognition. (b) George Washington dataset for word retrieval.

this termination level L. These focuses can be seen as the representation of a deformable grid adapted to the shape to be described. Examples of the distribution of focuses for levels L equal to 0, 1 and 2 are shown in Figures 3.2(a), 3.2(b) and 3.2(c) respectively.

3.2.3 Feature Extraction

Once focuses locations have been calculated, the **feature extraction** is computed according to the coordinates of focuses in set F^L. For every focus f_i, $i = \{1, 2, \ldots, 4^L\}$ we extract a sub-image I_i centered on their (x, y) coordinates (Figure 3.2(d)). The size of the sub-images depends on the number $\mathbf{c}$ of HOG cells and the size in pixels of every cell. For our experiments we fix the cell size to 16 pixels and analyze the performance with different number of $\mathbf{c} \times \mathbf{c}$ cells grid. As we said, we follow [33] and use HOG histograms of 31 dimensions to represent each cell. The final vector descriptor v of a given image results from the concatenation of all the histograms from all the focuses. Thus, its dimension depends on the number of cells $\mathbf{c}$ and the termination level L, *i.e.*, the number of focuses 4^L: $v \in \mathbb{R}^d$, $d = 4^L \cdot \mathbf{c}^2 \cdot 31$.

3.3 Experiments

The new descriptor proposed has been experimentally evaluated for two different purposes in two different datasets: NicIcon dataset [112] for hand-drawn symbol recognition and the George Washington dataset [86, 87] for handwritten word retrieval. We compare its performance with the cmiBSM and the HOG descriptor (setting the cell size to 16 pixels) for both tasks.

The NicIcon dataset (Figure 3.3(a)) is composed of 26,163 handwritten symbols of 14 classes from 34 different writers with on-line and off-line data available. The dataset is divided in three subsets (training, validation and test) for two different settings: writer dependent and writer independent. Every symbol has been cropped and size-normalized in an image of 256×256 pixels. We have selected the off-line data with the writer independent configuration for the symbol recognition task and we have used two different classifiers: Nearest Neighbor-based and Support Vector Machine with an exponential χ^2 kernel, whose cost and gamma parameters have been experimentally validated. For comparison we report the classification accuracy.

The George Washington dataset (Figure 3.3(b)) is comprised of 20 pages and 4,846 words. We apply a pre-processing for noise removal and slant correction and use the groundtruth information to segment the words. For word retrieval purposes we use the following protocol: each word is considered once as a query and used to rank the rest of the words using cosine as a similarity distance between words. We report the mean Average Precision of all the

queries, which is a standard measure in retrieval systems and can be understood as the area below the precision-recall curve. For the original HOG descriptor we resize all the images to a fixed size of 180×270, which is the mean value of height and width respectively.

Concerning the selection of parameters, we have fixed them with the aim of reaching a trade off between performance and dimensionality. However, we will further explore their influence in an extensive analysis, showing that performance of the nrHOG can be considerably increased at a cost of increasing the dimension of the feature vector. As it was shown in [4], cmiBSM performance reaches a plateu at the termination level L equal to 5, so we set it to this value for our experiments. So, in order to have a comparable dimensionality, we set L in the nrHOG equal to 2, and we use a grid of 3×3 to compute HOG features in the focuses. Finally, both HOG and nrHOG use a size bin equal to 16 pixels.

3.4 Results and Discussion

In Table 3.1 we show the classification accuracy in the NicIcon dataset for the three methods compared: *cmiBSM*, *HOG* and the proposed approach *non-rigid HOG*, denoted as *nrHOG*. We can see that for both classifiers used (Nearest Neighbor-based and SVM with exponential χ^2 kernel) HOG-based approaches outperform the *cmiBSM* descriptor. This confirms the need of capturing fine-grained details using more informative and discriminative features. Moreover, we also observe that the incorporation of an addaptative grid in the grid-based HOG improves the performance for the classification of shapes.

Table 3.1: Results in the NicIcon dataset for the word symbol recognition task in the writer independent configuration

Method	NN accuracy (%)	SVM accuracy (%)
$\text{cmiBSM}_{\text{f+p}}$ [4]	89.42	90.62
HOG [33]	93.47	96.68
nrHOG	95.88	97.69

Then, we show in Table 3.2 the mean Average Precision for the word retrieval task over the George Washington dataset, where we extract a similar conclusion: HOG-based features are able to deal with fine details to discriminate between handwritten shapes, and its combination with a deformable mesh substituting the rigid grid leads to a significant performance improvement. In this task, where shapes are more complex and we have to deal with a larger number of classes, differences between descriptors are considerably larger. The cmiBSM is clearly not able to deal with the fine details to correctly differentiate words, and it is surpassed by HOG-based descriptors. The proposed nrHOG approach reports the best performance.

As we said in the introduction, the integration of the deformable mesh of focuses provides to the original HOG descriptor some invariance to changes in the aspect ratio. This can be specially appreciated in the results of the GW dataset, where we have big changes in aspect ratio for images of the same class, so the difference in performance between nrHOG and HOG is larger than in the NicIcon, which only contains squared images.

Like in [4], we could use the focus coordinates as a feature vector and perform an in-kernel fusion with the HOG features when pre-computing the exponential χ^2 kernel to improve the

Table 3.2: Results in the George Washington dataset for the word retrieval task

Method	mAP (%)
$\text{cmiBSM}_{\text{f+p}}$ [4]	8.51
HOG [33]	37.21
nrHOG	44.59

Figure 3.4: Qualitateve results comparing cmiBSM, HOG and nrHOG for the word retrieval task over the George Washington dataset.

performance. However, in this case the improvement is unsubstantial and we do not consider worthy the extra computational time that this fusion requires.

Finally, we show in Figure 3.4 some qualitative results comparing the three approaches for the word retrieval task over the George Washington dataset. There we can see that, even that HOG improves the results of the cmiBSM by retrieving words whose shape is more similar to the query ("October", "November" and "December" share most of the characters), it is not enough to be able to differentiate between handwritten words. For that, we need to focus the description over the discriminative regions as the nrHOG does, resulting in this way in a better performance.

3.4.1 Parameters Analysis

The *nrHOG* has two main parameters: the termination level L, which determines the number of *focuses* used to describe the shape, and the value $\mathbf{c}$ of the $\mathbf{c} \times \mathbf{c}$ grid used to extract HOG features around every focus. In Figure 3.5 we explore the effect of these parameters. There we can see that increasing their values leads performance to increase. However, for both L and c, higher values means a larger vector dimension, so their value adjustment will be a trade-off between performance and dimensionality. Considering that the size of the cell has been fixed to 16 pixels, the dimension of the resulting feature vector of the HOG descriptor is equal to 3,906. In the case of the nrHOG, the first configuration that outperforms the HOG with the minimum dimensionality has a feature vector with dimension equal to 1,116. The configuration of the nrHOG that has the best performance in Figure 3.5 results in a vector with dimension equal to 198,400.

3.5 Conclusion and Future Work

In this work we have shown how a combination of a deformable grid and a fine-grained feature extraction method based on histograms of gradients can be used to describe handwritten shapes and can be applied to shape recognition and retrieval. We have also shown its

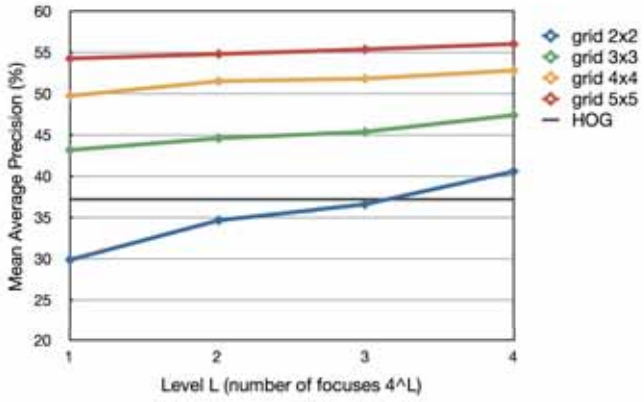


Figure 3.5: Influence of level L in the George Washington dataset for different number of grid cells. The size of cell is fixed to 16 pixels.

robustness against variability for different writing styles and different aspect ratios. This has resulted in a successful adaptation of the well-known HOG descriptor to the handwriting domain. We have obtained excellent results when comparing to other shape descriptors.

Part II

Word Spotting

Chapter 4

Segmentation-Free Word Spotting with Exemplar SVMs[1]

Abstract

In this paper we propose an unsupervised segmentation-free method for word spotting in document images. Documents are represented with a grid of HOG descriptors, and a sliding-window approach is used to locate the document regions that are most similar to the query. We use the Exemplar SVM framework to produce a better representation of the query in an unsupervised way. Then, we use a more discriminative representation based on Fisher Vector to rerank the best regions retrieved, and the most promising ones are used to expand the Exemplar SVM training set and improve the query representation. Finally, the document descriptors are precomputed and compressed with Product Quantization. This offers two advantages: first, a large number of documents can be kept in RAM memory at the same time. Second, the sliding window becomes significantly faster since distances between quantized HOG descriptors can be precomputed. Our results significantly outperform other segmentation-free methods in the literature, both in accuracy and in speed and memory usage.

4.1 Introduction

This paper addresses the problem of query-by-example word spotting: given a dataset of document images and a query word image, the goal is to identify and retrieve regions of those documents where the query word may be present. From decades ago this problem has attracted a lot of interest from the computer vision community [36, 38, 68, 87, 90], since making handwritten texts or ancient manuscripts available for indexing and browsing is of tremendous value. Interesting applications for word spotting are, for example, retrieving documents with a given word in company files, or searching online in cultural heritage collections stored in libraries all over the world. In this work, we specially focus on the unsupervised word-spotting problem, where no labeled data is available for training purposes.

[1]This chapter corresponds to the publication "J. Almazán, A. Gordo, A. Fornés and E. Valveny Segmentation-Free Word Spotting with Exemplar SVMs. *Pattern Recognition*, 2014".

This is a very common scenario since correctly labeling data is a very costly and time consuming task.

Traditionally, word-spotting systems have followed a well defined flow. First, an initial layout analysis is performed to segment word candidates. Then, the extracted candidates are represented as sequences of features [70, 89, 108]. Finally, by using a similarity measure – commonly a Dynamic Time Warping (DTW) or Hidden Markov Model (HMM)-based similarity –, candidates are compared to the query word and ranked according to this similarity. Examples of this framework are the works of Rath and Manmatha [87] and Rodríguez-Serrano and Perronnin [90]. One of the main drawbacks of these systems is that they need to perform a costly and error prone segmentation step to select candidate windows. Any error introduced in this step will negatively affect the following stages, and so it is desirable to avoid segmenting the image whenever possible. Unfortunately, since the comparison of candidate regions, represented by sequences, is based on costly approaches such as a DTW or a HMM, it is not feasible to perform this comparison exhaustively with a sliding-window approach over the whole document image.

Late works on word spotting have proposed methods that do not require a precise word segmentation, or, in some cases, no segmentation at all. The recent works of [36, 38] propose methods that relax the segmentation problem by requiring only a segmentation at the text line level. One of their drawbacks, however, is that they require a large amount of annotated data to train the Hidden Markov Models [36] or the Neural Networks [38] that they use. Additionally, the line segmentation still has to be very precise to properly encode the lines.

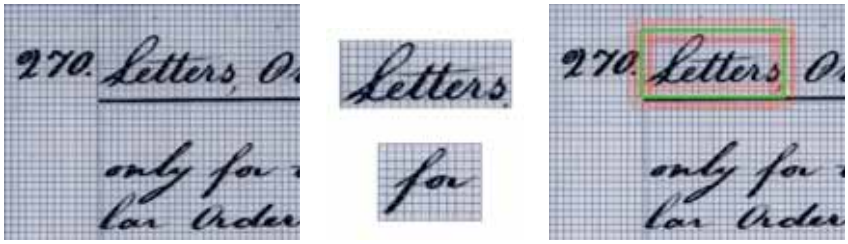
In [39], Gatos and Pratikakis perform a fast and very coarse segmentation of the page to detect salient text regions. Queries are represented with a descriptor based on the density of the image patches.Then, a sliding-window search is performed only over the salient regions of the documents using an expensive template-based matching.

The methods proposed by Leydier *et al.* [60] and Zhang and Tan [115] avoid segmentation by computing local keypoints over the document images. While [60] represents the document images with gradient-based features, [115] uses features based on the Heat Kernel Signature. The main drawback of these approaches is that they use a costly distance computation, which is not scalable to large datasets.

The work of Rusiñol *et al.* [95] avoids segmentation by representing regions with a fixed-length descriptor based on the well-known bag of visual words (BoW) framework [24]. In this case, comparison of regions is much faster since a dot-product or Euclidean distance can be used, making a sliding window over the whole image feasible. To further improve the system, unlabeled training data is used to learn a latent space through Latent Semantic Indexing (LSI), where the distance between word representations is more meaningful than in the original space. Rothacker *et al.* [94] exploits the use of the BoW representation to feed a HMM and avoids the segmentation step by means of a patch-based framework. Comparison of regions is slower than the BoW-based approach of [95], so it could not be directly applied in a large-scale scenario, but, thanks to the sequential encoding of the BoW features of the HMM, they obtain a more robust representation of the query.

Howe [44] uses a generative model for word appearance from a single positive sample, resulting in an example of a *one-shot learning* approach. The model consists of a set of nodes connected via springlike potentials, and arranged in a tree structure whose a priori minimum energy configuration conforms to the shape of the query word.

In this work we focus on this family of segmentation-free word-spotting approaches and we argue that previously described current methods can be improved in several ways. **First, they can be improved in the choice of low level features.** The features of [70, 89, 94, 108] produce sequence representations, which are usually slower to compare than fixed-length representations. The work of [39] uses a descriptor based on the patch density, which



(a) HOG grid. (b) (c)

Figure 4.1: a) Grid of HOG cells. Only one small part of the image is shown. b) Two random queries of the George Washington dataset. The windows adjust around the HOG cells. c) A query (in green) and some positive samples (in red) used to learn the Exemplar SVM. To avoid clutter, only some of the positive windows are shown.

is insufficient to capture all the fine-grained details. The bag of visual words approach of [95] fixes the size of the scanning window for efficiency reasons, which makes the accuracy of the method very dependent of the size of the query. We address these issues by using a sliding-window approach based on HOG descriptors [26], which have been shown to obtain excellent results when dealing with large variations in the difficult tasks of object detection and image retrieval. The document images are represented with a grid of HOG descriptors, where each cell has the same size in pixels. (*cf*. Fig 4.1(a)). In this case, we do not have a fixed window size; instead, the window size is adjusted to the query size, covering several HOG cells (*cf*. Fig 4.1(b)).

HOG descriptor provides a good trade-off between speed and memory requirements and discriminative power. However, more discriminative representations exist, but cannot be used in practice when dealing with large volumes of data – such as those of large-scale datasets or sliding-window setups. We propose to apply a common solution that consists of, first, using affordable features (*i.e*. HOG) to produce an initial ranking over all the dataset, and then, use more powerful and expensive representations to rerank only the most promising candidates returned by the first stage. Since only a small subset of the whole dataset is scanned, it is feasible to use more expensive features. In the case of word spotting, however, it is not clear which features would be better suited for this task. Therefore, we have included in the experiments a comparison of low-level features to represent the word images with the objective of analyzing their use on a reranking step. This reranking step can significantly improve the accuracy with a cost that does not depend on the number of documents on the dataset.

Second, spotting methods can be improved in the learning of a more discriminative space. In [95], LSI is used to learn a latent space where words and documents are more related. However, learning a semantic space with LSI may be too conditioned to the words used in the unsupervised training stage, and adapting to new, unseen words may be complicated. Instead, we propose to perform this unsupervised learning once the query has been observed, and adapt the learning to the particular query. For this task, we propose to use a similar approach to the Exemplar SVM framework of [67, 103]. Additionally, to further improve the representation of the query, we propose to combine reranking with a query expansion step, which uses the best candidates after the reranking step to construct a new, more informative representation of the query, and use it to query again the dataset to

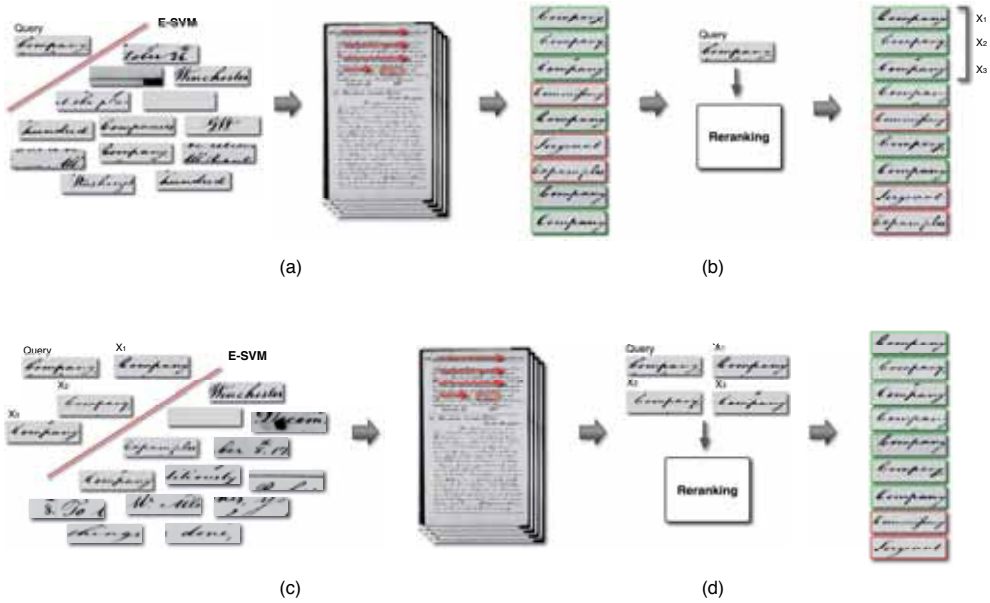


Figure 4.2: General scheme of the method proposed. (a) E-SVM training and sliding-window search. (b) First reranking of the best retrieved regions. (c) E-SVM retraining and sliding-window search applying query expansion with the first reranked regions. (d) Second reranking using the expanded training set.

further improve the results. As a consequence of this reranking step it may be necessary to train several exemplar models per query. We propose to use a fast solver based on Stochastic Gradient Descent (SGD) to considerably speed up the training process.

Finally, these methods can be improved in the cost of storing the descriptors of all the possible windows of all the dataset items. Assuming HOG descriptors of 31 dimensions represented with single-precision floats of 4 bytes each (*i.e.*, 124 bytes per HOG), and 50,000 cells per image, storing as few as 1,000 precomputed dataset images would require 5.8GB of RAM. Since documents will not fit in RAM memory when dealing with large collections, we are left with two unsatisfying options: either recomputing the descriptors of every document with every new query, or loading them sequentially from a hard disk or a solid-state drive (SSD). Any of these approaches would produce a huge performance drop in the speed at query time[2]. To address this problem, we propose to encode the HOG descriptors using Product Quantization (PQ) [48]. Encoding the descriptors with PQ would allow us to reduce the size of the descriptors and to preserve a much larger amount of images in RAM at the same time. As a side effect, computing the scores of the sliding window also becomes significantly faster.

In summary, we present a full system for efficient unsupervised segmentation-free word spotting that will be shown to outperform existing methods in two standard datasets. The use of HOG templates provides a very natural model, and its discriminative power is improved through the use of Exemplar SVMs with SGD solvers. The use of PQ drastically improves the efficiency of the system at test time. Finally, the use of more

[2]A similar point can be argued about the methods of [39], [95] or [94].

informative features in combination with reranking and query expansion improves the final accuracy of the method at a reduced cost. A general scheme of the method can be seen in Figure 5.3.

A preliminary version of our system was described in [7]. The system described here extends the work of [7] in the following ways: the comparison of different low-level features for word representation, the introduction of reranking and query expansion to improve performance, the analysis of different configurations of the PQ compression method, the use of a faster Stochastic Gradient Descent-based solver for learning the Exemplar SVM and new experiments with the full system integrating all these steps.

The rest of the paper is organized as follows. Section 4.2 describes the basic configuration of a HOG-based word-spotting approach. Section 4.3 extends it to make use of the Exemplar SVM framework, and Section 4.4 introduces the use of PQ to compress the HOG descriptors of the document. Then, reranking and query expansion are described in Section 4.5 and Section 4.6. Finally, Section 5.6 deals with the experimental validation and Section 5.7 concludes the paper.

4.2 Baseline System

Word spotting – and, particularly, handwritten word spotting – is a challenging problem for descriptors because they have to deal with many sources of variability such as deformations, different styles, *etc*. Moreover, in a large-scale and segmentation-free scenario, it is mandatory to use descriptors that are both fast to compute and compare, and that could be integrated in a sliding-window based search. Traditionally, features for word spotting have been based on sequences [70, 89, 108] and compared using methods such as Dynamic Time Warping or Hidden Markov Models. The rationale behind this is that these types of variable-length features would be able to better adapt to the word they represent and capture information independently of deformations caused by different styles, word length, *etc*. The main drawback of these methods is that comparing representations is slow, with usually a quadratic cost with respect to the length of the feature sequence, which makes them impractical in large-scale scenarios as well as in a sliding-window setup.

Interestingly, it has been shown that these variable-length features do not always lead to the best results and can be outperformed by fixed-length representations. The work of Perronnin and Rodríguez-Serrano [82] exploits the Fisher kernel framework [45] to construct the Fisher Vector of a HMM, leading to a fixed-length representation that outperforms standard HMM over variable-length features. Rusiñol *et al*. [95] represent document image regions with a descriptor based on the well-known bag of visual words framework [24] over densely extracted SIFT descriptors, obtaining a reasonable performance in segmentation-free word spotting. Unfortunately, although these features are superior to some sequence-based ones, they still remain impractical for a sliding-window setup since computing the descriptors for every window is usually quite slow. Rusiñol *et al*. [95] address this issue by fixing the window size and precomputing offline the descriptors of all possible windows of that size. This, however, makes the results very dependent on the size of the query, since very small or very large queries will not correctly adapt to the precomputed windows.

Because of these reasons here we advocated for the use of HOG descriptors [26], which have been shown to obtain excellent results in difficult tasks such as pedestrian and object detection and image retrieval (see, *e.g*., [33] or [103]). Although in general HOG descriptors cannot compete in terms of accuracy with other more powerful features, they are very fast to compute and compare, and are particularly suited for problems that require a sliding-window search such as this one, providing a very good trade-off between accuracy and

speed. Now, as we will show in Section 4.5, it is actually possible to use more powerful and discriminative features during a second step to rerank the best candidate words yielded by the sliding-window approach at a very reasonable cost, overcoming the main deficiency of HOG descriptors for this task.

In our system, the document images are divided in equal-sized cells (see Fig 4.1a) and represented with HOG histograms, which encode local gradients. We follow [33] and use HOG histograms of 31 dimensions (9 contrast insensitive orientation features, 18 contrast sensitive orientation features and 4 features to reflect the overall gradient energy around the cell) to represent each cell. Queries are represented analogously using cells of the same size in pixels. In this case, as opposed to [95], we do not have a fixed window size; instead, the window size depends on the query size, covering $H_q \times W_q$ HOG cells (*cf*. Fig 4.1(b)), leading to a descriptor of $H_q \times W_q \times 31$ dimensions. Note that the number of cells in a query, and therefore the final dimensionality of the descriptor, depends on the size of the query image. The score of a document region $\mathbf{x}$ of $H_q \times W_q$ cells with respect to the query $\mathbf{q}$ is then calculated as a 3D convolution, $\text{sim}(\mathbf{q}, \mathbf{x}) = \mathbf{q} * \mathbf{x}$. This could also be understood as calculating the dot-product between the concatenated HOGs of the query $\bar{\mathbf{q}} = \text{vec}(\mathbf{q})$ and the concatenated HOGs of the document region $\bar{\mathbf{x}} = \text{vec}(\mathbf{x})$, *i.e.*, $\text{sim}(\mathbf{q}, \mathbf{x}) = \bar{\mathbf{q}}^{\mathbf{T}}\bar{\mathbf{x}}$, but we remark that at test time we perform a convolution instead of concatenating the descriptors explicitly. Following this approach, we can compute the similarity of all the regions in the document image with respect to the query using a sliding window and rank the results. To avoid returning several windows over the same region, Non-Maximum Suppression (NMS) is performed to remove windows with an overlap over union larger than 20%.

We further modify the baseline in two ways. First, instead of using the HOG descriptors directly, we reduce their dimensionality down to 24 dimensions with PCA. We observed no loss in accuracy because of this, probably because of the "simplicity" of text patches. Second, instead of calculating the dot-product, we are interested in the cosine similarity, *i.e.*, calculating the dot-product between the L2 normalized descriptors. The cosine similarity is a typical choice in document retrieval, and we observed experimentally that L2 normalizing the vectors can indeed make a significant difference in the results. Note that the L2 normalization is performed at the region level, not at the cell level. Fortunately, we do not need to explicitly reconstruct the regions to normalize the descriptors, since $\text{sim\_cos}(\mathbf{q}, \mathbf{x}) = (\frac{\bar{\mathbf{q}}}{||\bar{\mathbf{q}}||})^T(\frac{\bar{\mathbf{x}}}{||\bar{\mathbf{x}}||}) = \frac{1}{||\bar{\mathbf{q}}||}\frac{1}{||\bar{\mathbf{x}}||}\bar{\mathbf{q}}^T\bar{\mathbf{x}} = \frac{1}{||\bar{\mathbf{q}}||}\frac{1}{||\bar{\mathbf{x}}||}\mathbf{q} * \mathbf{x}$ Therefore, we can calculate the $\text{sim}(\mathbf{q}, \mathbf{x})$ score with a convolution without explicitly concatenating the descriptors and later normalize it with the norms of the query and the document region. The norm of the query can in fact be ignored since it will be constant for all the document regions and therefore does not alter the ranking. As for the region patch, we can accumulate the squared values while performing the convolution to construct, *online*, the norm of the region patch without explicitly reconstructing it.

4.3 Exemplar Word Spotting (EWS)

In the previous section we introduced how HOG descriptors can be used in the framework of a sliding-window based approach for word spotting. There, we used a basic retrieval setting based on the cosine similarity. We note however, that the cosine similarity, despite being a reasonable option, may not be the optimal way to compare document regions, and that learning a better metric may yield significant improvements. In [95], this is achieved by learning, *offline*, a latent space through LSI, where the cosine similarity is an appropriate measure. However, this may be too conditioned to the words used in the unsupervised training stage, and adapting to new, unseen words may be complicated. Additionally, it is not clear how we could adapt this latent learning to our grid of HOGs framework.

Here we take a different approach and propose to learn, *at query time*, a new representation of the query, optimized to maximize the score of regions relevant to the query when using the dot-product. This new representation can be understood as *weighting* the dimensions of the region descriptors that are relevant to the query. We achieve this goal by means of the Exemplar SVM framework [67, 103]. Let us assume that we have access to a set $\mathcal{P}$ of positive regions that are relevant to the query. These are described as the concatenation of their PCA-compressed HOG descriptors, and are L2 normalized. Analogously, let us assume that we have access to a set $\mathcal{N}$ of negative regions that are not relevant to the query. In this case, we could find a new representation $\mathbf{w}$ designed to give a high positive score to relevant regions, and a high negative score to non-relevant regions when using the dot-product with L2 normalized regions. This can be casted as an optimization problem as

$$\operatorname{argmin}_{\mathbf{w}} \frac{1}{2}||\mathbf{w}||^2 + C_1 \sum_{\{x_p, y_p\} \in \mathcal{P}} L(y_p \mathbf{w}^T \mathbf{x}_p) + C_2 \sum_{\{x_n, y_n\} \in \mathcal{N}} L(y_n \mathbf{w}^T \mathbf{x}_n), \tag{4.1}$$

where $y_p = 1$, $y_n = -1$, $L(\mathbf{x}) = \max(0, 1 - \mathbf{x})$ is the hinge loss and C_1 and C_2 are the cost parameters of relevant and non-relevant regions. This is very similar to the standard SVM formulation and can be solved by standard solvers such as LIBLINEAR [31]. The classifier bias can be considered implicitly by augmenting the samples as $\mathbf{x} = [\mathbf{x}\ \textit{bias\_multiplier}]$, where *bias_multiplier* usually equals 1. Solving this optimization produces a weight vector $\mathbf{w}$, which can be seen as a new representation of the query designed to maximize the separation between relevant and non-relevant regions. As in the baseline system, we can rearrange the terms at test time so that $\text{sim}(\mathbf{w}, \frac{\mathbf{x}}{||\bar{\mathbf{x}}||}) = \frac{1}{||\bar{\mathbf{x}}||}\text{sim}(\mathbf{w}, \mathbf{x})$, where $\text{sim}(\mathbf{w}, \mathbf{x})$ can be calculated without reconstructing the region vectors and $||\bar{\mathbf{x}}||$ can be calculated online while performing the convolution. Note that, once learned, the classifier bias is constant for every dataset sample given a query, and therefore it is not needed to rank the words. However, considering the bias during the learning may lead to a better $\mathbf{w}$ model, and so it is important to include it during training even if afterwards it is not explicitly used.

Unfortunately, in most cases we will not have access to labeled data, and so $\mathcal{P}$ and $\mathcal{N}$ will be unavailable. To overcome this problem, $\mathcal{P}$ is constructed by deforming the query, similarly to what is done in [103]. In our case, we slightly shift the window around the query word to produce many almost identical, shifted positive samples (see Fig 4.1(c)). As a side effect, at test time, sliding windows that are not completely centered over a word will still produce a high score, making the approach more resilient to the sliding window granularity. To produce the negative set $\mathcal{N}$, we sample random regions over all the documents after filtering those with very low norm. Note that, since we do not have access to segmented words, we can not guarantee that a given negative region will contain a complete word or a word at all. This is different from unsupervised methods that perform word segmentation such as that of [90]: even if they do not use labeled data, they have access to the bounding boxes of training words. As in [103], positive samples could also appear in this randomly chosen negatives set.

In the most basic setup, this model needs to be learned only once per query independently of the number of documents on the dataset, and so the learning time becomes small compared to the complete retrieval time if the number of documents in the dataset is not small. Still, using a batch solver such as LIBLINEAR [31] as was done in [7] may require between one and two seconds per query, which is not negligible. This may be even worse when using query expansion (*cf*. Section 4.6), since under this setup it is needed to learn the exemplar model not just once but at least twice per query. Here we propose to use a Stochastic Gradient Descent (SGD) implementation that can very significantly reduce the training time while retaining good accuracy results. The key idea behind SGD is that the classifier is updated one sample at a time based on the (sub-)gradient of the objective function O (Equation (4.1)

in our case), *i.e.*, $\mathbf{w_{(t+1)}} = \mathbf{w_{(t)}} - \eta \frac{\nabla O}{\partial \mathbf{w_{(t)}}}$, where η is the learning rate or step rate. In our case, that leads to the following update:

$$\mathbf{w_{(t+1)}} = (1 - \lambda\eta)\mathbf{w_{(t)}} + \eta\delta_{it} y_i \mathbf{x}_i, \tag{4.2}$$

where $\delta_{it} = 1$ if $L(y_i {\mathbf{w_{(t)}}}^T \mathbf{x}_i) > 0$ and 0 otherwise, and λ is a regularization parameter. We consider the set $\mathcal{PN} = \mathcal{P} \cup \mathcal{N}$, and we randomly sample pairs $\{\mathbf{x}_i, y_i\} \in \mathcal{PN}$ and use them to update $\mathbf{w}$ using Equation (4.2). The initial weight vector $\mathbf{w_{(0)}}$ is initialized randomly from a Normal distribution of mean 0 and variance $1/\sqrt{d}$, where d is the dimensionality of $\mathbf{w}$. The process is repeated for several iterations until convergence. Instead of setting the C_1 and C_2 weights of equation (4.1) directly, we follow the approach of [1] and control the proportion of negatives to positives samples used at every iteration. As we will see experimentally, the SGD solver leads to results comparable to LIBLINEAR while being approximately 10 times faster.

4.4 Feature Compression

One important drawback of sliding-window based methods such as this or [39, 95] is the cost of recomputing the descriptors of every image with every new query. As we will see during the experimental evaluation (*cf*. Section 5.6), computing the HOG descriptors of an image can take between 50% and 90% of the total test time per document, and this has to be recomputed for every new query. A possible alternative could be precomputing and storing the HOG descriptors. However, this is usually not a feasible option because of the large amount of memory that would be necessary to maintain them. Assuming HOG descriptors of 31 dimensions represented with single-precision floats of 4 bytes each (*i.e.*, 124 bytes per HOG), and 45,000 cells per image, storing as few as 1,000 precomputed dataset images would require 5.2GB of RAM. Even if we compress the HOG descriptors with PCA down to 24 dimensions, we can barely fit 250 documents in 1GB of Memory. Since documents have to be kept in RAM to be rapidly accessed, storing large collections of documents would be extremely expensive or directly unfeasible.

In this section we propose to encode the HOG descriptors by means of Product Quantization (PQ) [48]. This technique has shown excellent results on approximate nearest neighbor tasks [49, 50], maintaining a high accuracy while drastically reducing the size of the signatures. After compression, we can store up to 24,000 images per GB of RAM, depending on the PQ configuration. As a side effect, because of the dimensionality reduction, the sliding-window comparisons will also be faster: without PQ, we can analyze approximately 5 documents per second, and that is excluding the time to compute the HOG descriptors of each page. After PQ, we can analyze approximately 70 document images per second, an almost 15 fold improvement in speed.

In the following section we will first give an overview of PQ, and then we will describe how PQ fits in our system, both in the exemplar training and the sliding-window stages.

4.4.1 Product Quantization

In vector quantization the goal is to find a finite number of reproduction values $\mathcal{C} = \{\mathbf{c}_1, \dots \mathbf{c}_k\}$ (usually through k-means), and then represent the vector with the the closest reproduction value, *i.e.*, given a vector $\mathbf{x} \in \mathbb{R}^D$ and a quantizer $\mathtt{q}$, then

$$\mathtt{q}(\mathbf{x}) = \operatorname{argmin} \mathbf{c}_i \in \mathcal{C} d(\mathbf{x}, \mathbf{c}_i), \tag{4.3}$$

where d is usually the Euclidean distance. Assuming k centroids, then quantized vectors can be encoded using as few as $\log_2 k$ bits by storing the indices of the centroids and not the centroids themselves, which are kept in a different table. Unfortunately, vector quantization is not possible when the dimensionality of the vectors is not trivially low: as noted in [48], to encode a descriptor of 128 dimensions using only 0.5 bits per dimension, we would need to compute 2^{64} centroids, which is not feasible.

PQ addresses this issue by quantizing groups of dimensions independently. Given a set of D-dimensional vectors, each vector is split sequentially into m groups of D/m dimensions, and then, a different quantizer is learned for every group of sub-vectors. Assuming k centroids per group, this leads to $m \times k$ centroids. Now, given vector $\mathbf{x}$ that we want to encode, we denote with $\mathbf{x}^j$ the j-th group of $\mathbf{x}$, and with $\mathbf{c}_{ji} \in \mathcal{C}_j$ the i-th centroid learned from group j, then

$$\mathfrak{q}_j(\mathbf{x}) = \operatorname{argmin} \mathbf{c}_{ji} \in \mathcal{C}_j \; d(\mathbf{x}^j, \mathbf{c}_{ji}), \tag{4.4}$$

and $\mathfrak{q}(\mathbf{x}) = \{\mathfrak{q}_1(\mathbf{x}), \mathfrak{q}_2(\mathbf{x}), \ldots, \mathfrak{q}_m(\mathbf{x})\}$. The final codification of vector $\mathbf{x}$ results from the concatenation of the indices of the m centroids. In this case, to produce a b bits code, each quantizer needs to compute only $2^{b/m}$ centroids, which is reasonable if m is chosen appropriately.

In our case, we use PQ to encode our PCA-HOG descriptors of 24 dimensions. In the experiments section we compare the performance when using different number m of groups of $k = 256$ centroids, *i.e.*, 8 bits to represent each one.

One advantage of PQ is that to compute the distance between a query $\mathbf{q}$ and a (quantized) document x from a dataset – represented by m indices to the centroids table –, it is not necessary to quantize the query or to explicitly decode the quantized document. We can, at query time, pre-calculate a look-up table $\ell(j, i) = d(\mathbf{q}^j, \mathbf{c}_{ji})$. Note that this table does not depend on the number of elements of the dataset, only on the number of groups m and centroids k, and so the cost of computing it is negligible if the number of documents is large. Once this look-up table has been constructed, we can calculate the distance between query $\mathbf{q}$ and a quantized document x as $\sum_{j=1}^{m} \ell(j, x_j)$, where x_j is the j-th index of x, without explicitly reconstructing the document.

To learn the Exemplar SVM, we need to create the sets $\mathcal{P}$ and $\mathcal{N}$. As seen in Figure 4.1(c), positive samples do not exactly align with the precomputed cells, and therefore we need to recompute the descriptors for those regions. Note that this is a small amount of descriptors compared to the whole image. The negative samples, however, can be chosen so that they align with the precomputed grid of HOGs, decoded, and then fed to the SVM training method. Similarly to [97], we decode the features and learn our classifier on the decoded data. When calculating the sliding window, we no longer have to compute the dot-product between the query and the HOG descriptor to obtain the score of a cell, and it is enough to perform m accesses to the look-up table to obtain that score. Intuitively, this could lead to comparisons between 24 (with $m = 1$) and 4 (with $m = 6$) times faster with respect to the 24-dimensional HOG descriptors. In practice, we have obtained, depending on the number m of groups, up to 15 fold speed-ups.

4.5 Reranking

The main advantage of the HOG-based method proposed in this paper is that it can be efficiently computed over a large dataset of images in combination with a sliding window-based search. This allows one to apply it over non-segmented documents, which is mandatory in many real scenarios where pre-segmenting the words without errors is not possible. One

drawback, though, is that HOG features are in clear disadvantage in a direct comparison in performance with other more complex and informative image representations that unfortunately cannot be applied to all the dataset due to their cost.

In this paper we propose to improve the effectivity of the HOG-based framework using reranking, a popular technique applied in image retrieval [11, 22]: it consists of applying a second ranking step that considers only the best windows retrieved by an initial efficient ranking step, and that uses more discriminative (and costly) features. These features cannot be used over the whole dataset due to their cost, but it is feasible to use them only on a small subset of windows. Common methods in image retrieval use *geometrical verification*, which applies strong spatial constrains, ensuring that both query image and images retrieved contain the same scene [22]. This can be ensured because different views of the same scene can be seen as an affine transformation of their points, and therefore techniques such as Ransac can be used to verify this transformation accurately. However, due to the variability and inconsistency in handwriting, transformations are not affine and we cannot apply this technique in our case.

We therefore analyze other representations that are appropriate for our problem. In the recent [62], the authors perform a similar analysis, showing that a bag of words representation using SIFT descriptors can outperform classic approaches such as DTW based on sequence features as well as graph-based or pseudo-structural descriptors. We build on that work, and analyze how the HOG features compare with respect to DTW using the popular Vinciarelli features [108][3] as well as an encoding based on the bag of words framework, the Fisher Vector (FV) [84].

The Vinciarelli features are extracted by computing local descriptors using a horizontal sliding-window approach over the image. At each region, the local window is divided in a 4×4 grid and the density of each region is computed, leading to a window descriptor of 16 dimensions. The descriptor of the word image is a sequence of such 16-dimensional descriptors, where the exact number depends on the length of the image to represent. Due to their variable length, methods such as DTW are necessary to compare these descriptors.

The Fisher Vector [84] can be seen as a bag of words that captures not only the visual word count but also higher order statistics. The FV was recently shown to be a state-of-the-art encoding method for several computer vision tasks such as image retrieval and classification [18].

To the best of our knowledge, the only work that applies unsupervised reranking in a word-spotting context is that of [102]. In this case, a bag of words was used to represent images and to perform hashing, which allowed a fast but noisy retrieval. On a second stage, spatial pyramids were used to improve the representation adding spatial information, which was missing on the indexing stage. Note that in this case the reranking uses the same features, and the spatial pyramid can be seen as a way to calibrate the score of each region of the word independently. In our case, we are not just adding spatial information but using more powerful and discriminative features.

4.6 Query Expansion

Although we focus on a completely unsupervised case, and therefore only one image representation of the query is usually available at first, this framework could be improved if several instances of the query word became available, since a more discriminative model could be learned. For example, we could make use of query expansion, a technique popular in instance-level image retrieval works [11, 21, 22].

[3]These features are more discriminative than the ones used in [62].

Query expansion is based on improving the representation of the query using retrieved and verified images from the dataset. In the work of Chum *et al*. [22] a number of the best ranked images from the original query are verified using strong spatial constrains, and the validated images are combined into a new query. They use BoW [24] as the image representation and they average together the query BoW vector and BoW vectors of the new images. Arandjelović and Zisserman[11] introduce a discriminative query expansion approach, where negative data is also taken into account and a classifier is trained. In a different work [10], they use multiple queries to retrieve images extracted from a text search in Google images and they propose different ways to combine either their representations or their retrieved results. Query expansion has also been applied to word spotting, although in combination with relevance feedback, by Rusiñol and Lladós [96]. Their system asks the user to cast several queries instead of a single one and combines the results by testing different strategies. To the best of our knowledge, there exist no word-spotting methods that perform query expansion in a completely unsupervised way.

Here, like in [22], we assume that the best results retrieved correspond to the query and can be used to improve its representation. However, due to the variability and inconsistency of handwriting text, we cannot apply the geometric constrains generally used in natural images. Instead, we use as a verification step the reranking process described in Section 4.5. Once results retrieved by the sliding-window search are reranked using more informative features, we define a set $\mathcal{X}$, composed by the k best windows retrieved and the original query. This is used as the set of positive examples of the query model. Although we have no absolute guarantees that the set $\mathcal{X}$ will contain only positive samples, we observed a significant improvement of the accuracy on the tested datasets.

In this paper we explore two different ways of combining and exploiting this new set of positive examples in the HOG-based Exemplar SVM framework:

Single-Exemplar: This approach consists of training a single Exemplar SVM with $\mathcal{X}$ as the set of positive images. We build set $\mathcal{P}$ of relevant examples by applying the shifting deformation of the window as described in Section 4.3 and showed in Figure 4.1(c) to every sample in $\mathcal{X}$, producing many almost identical, shifted windows. Training results in a single weight vector $\mathbf{w}$, which is used again to retrieve new windows.

Multi-exemplar: It consists of training one Exemplar SVM, as detailed in Section 4.3, for every sample in $\mathcal{X}$. It results in a set $\mathbf{W}$ that contains the weight vector $\mathbf{w}_i$ of every Exemplar SVM. Finally, retrieved ranked lists are combined by scoring each window by the average of the individual scores obtained from each Exemplar SVM: $\frac{1}{k}\sum_{i=0}^{k}\text{sim}(\mathbf{w}_i, \mathbf{x})$.

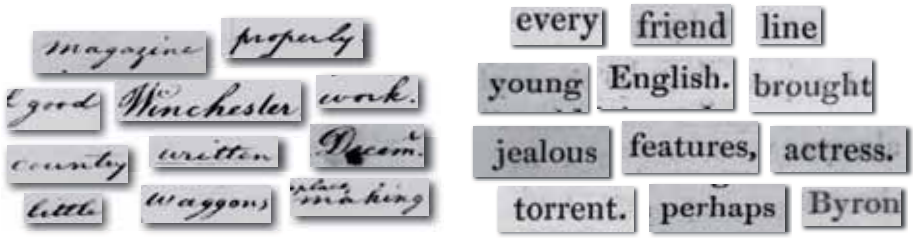
Finally, set $\mathcal{X}$ can also be used to improve the FV representation of the query and perform a second reranking process. We use as the new representation of the query the average FV of the representations extracted from the samples in set $\mathcal{X}$. Then, it is used to rerank the first regions retrieved by the *expanded model* of the query, as it is described in Section 4.5.

4.7 Experiments

4.7.1 Experimental Setup

Datasets and Performance Evaluation. We evaluate our approach on two public datasets: The George Washington (GW) dataset [86, 87] and the Lord Byron (LB) dataset [95][4]. These datasets are comprised of 20 pages each and contain approximately 5,000 words. George Washington (Figure 4.3(a)) contains handwritten text while Lord Byron

[4]We obtained the exact images and groundtruth after direct communication with the authors of [95].



(a) GW (b) LB

Figure 4.3: Examples of the words contained in the (a) GW and (b) LB datasets.

(Figure 4.3(b)) only contains typewritten text. We follow a similar protocol as used in [95]: each word is considered as a query and used to rank all the regions of every document in the dataset. However, as opposed to [95], the query image, if retrieved, is removed from the retrieved results and not considered in the performance evaluation. This is consistent with most other works on word spotting. For compatibility reasons, however, we will also report results without removing the query using our final system to ease the comparison with [95], since it is the work most related to ours. A region is classified as positive if its overlap over union with the annotated bounding box in the groundtruth is larger than 50%, and negative otherwise. For every document, we keep only the 1,000 regions with the highest score and perform NMS to avoid overlaps of more than 20%. Finally, we combine the retrieved regions of all the documents and rerank them according to their score. We report the mean Average Precision (mAP) as our main measure of accuracy. The mAP is a standard measure in retrieval systems and can be understood as the area below the precision-recall curve.

Parameters. To compute the HOG grid, we use cells of 12 pixels, which resulted in a reasonable trade-off in performance and time consumption. On both datasets, using this cell size produces approximately 45,000 cells for each document. For the sliding-window search we have also fixed the step size to one cell. An extended analysis on the effect of different cell sizes and step sizes can be found in [7]. When performing the unsupervised learning of PCA and PQ, we randomly sample 10,000 HOGs from all the documents, after filtering those with very low norm, and 1.5 million SIFTs for the unsupervised learning of the GMM for the FV representation. When learning the Exemplar SVM, we used the Stochastic Gradient Descent-based solver of the JSGD package [1] and fixed the step size η to 10^{-3} and the regularization parameter λ to 10^{-5} based on the results of a small subset of 300 queries. This provided reasonable results on both datasets, although better results could be obtained by fine-tuning this parameter individually for each dataset. We produce 121 positive samples (11 shifts in horizontal $\times$ 11 shifts in vertical) for each query (and for each expanded example in case that query expansion is used) and 7,744 negative samples (64 times the number of positive samples of the query). Increasing the number of negative samples led to a very slight improvement in the accuracy, but we did not consider it worth the extra cost during training. Note that [67, 103] propose to use several iterations of hard negative mining. However, we did not experience any gain in accuracy by doing so. Finally, we evaluate reranking using different k for selecting the *top-k* retrieved results to be reranked and query expansion using different number of examples to expand the query.

We used MATLAB's built-in profiler to measure the running times of the different sections of our pipeline on an Intel Xeon running at 2.67GHz using one single core. We noticed that the profiler added approximately a 10% overhead that we have not subtracted. There-

fore, the actual times may be slightly faster than reported. Note that, although we used MATLAB, the core sections (computing HOG and FV descriptors, training the SVM, calculating the scores with a sliding window, and the NMS) were implemented in C. When reporting the sliding-window times, these already include the time to perform the ranking and the NMS.

4.7.2 Word Spotting Results and Discussion

In [7], we already analyzed the influence of the parameters of the basic configuration of the proposed approach: the cell size of the HOG descriptor and the step size of the sliding-window search. Here, we will focus on evaluating the impact of the new contributions included in this paper. First, we will analyze the performance of different features for representing word images. For that, we will use a simplified version of the datasets where the document words have already been segmented using the ground truth annotations. Then, in a segmentation-free setting and relying on a sliding-window approach, we will analyze the influence of the SGD solver and the different configurations of PQ, reranking and query expansion. Finally, we will compare the whole system with recent word-spotting methods.

Comparison of Features. We begin our experiments by testing the proposed method on a simplified setting, where the document words have already been segmented using the ground truth annotations. In this setting, we test the effectivity of our HOG and HOG+SVM approaches, and compare it with a descriptor based on Vinciarelli features with a DTW distance, as well as with a FV representation. We also study the computational needs of each approach. The objective of this test is twofold. First, to show that, although the accuracy of the proposed HOG+SVM approach may be outperformed by the Vinciarelli features or the FV, the accuracy of the HOG+SVM is reasonably high. Second, to highlight that neither the Vinciarelli nor the FV approaches can be used in a segmentation-free approach due to their computational costs, but can still be used to rerank a short list of results produced by the HOG+SVM approach.

Under this setup, different words may have different numbers of HOG cells, which affects their dimensionality. To compare a query and a document word of different sizes, the document word is first slightly enlarged by a 10%, and then the query word is "searched" inside the document word using a sliding-window approach. The best window score is used as a measure between the query and the word.

To compute the Vinciarelli features, we experimented with several region sizes and used the one that yielded the best results, which gives it an slightly unfair advantage. To compute the FV we densely extract SIFT features from the images and use a Gaussian mixture model (GMM) of 16 Gaussians. To (weakly) capture the structure of the word image, we use a spatial pyramid of 2×6 leading to a final descriptor of approximately $25,000$ dimensions.

The results are shown in Table 4.1.

Dataset	HOG	HOG+SVM	Vinciarelli+DTW	FV
GW	40.24	49.19	56.25	64.90
LB	75.37	83.04	83.47	91.75

Table 4.1: Retrieval performance in mAP of different descriptors for segmented words.

We observe how, indeed, the powerful FV features obtain the best results on both datasets despite being a fixed-length feature. The Vinciarelli features with DTW outperform HOG and HOG+SVM on the GW dataset, suggesting that HOG is quite rigid for the type of

variations found in this dataset, although it still achieves reasonable results. On the LB dataset, where variations are much smaller, HOG+SVM performs similar to the Vinciarelli features, despite the unfair advantage of the Vinciarelli features.

Regarding the computational costs, there are two separate issues: computing the descriptor given a document window, and comparing the descriptors. With our setup, searching a query in a document page requires on average to compute and compare about 40,000 window descriptors. Because of this, computing and comparing window descriptors needs to be extremely fast to be feasible on a segmentation-free setting.

- Descriptor computation. On the HOG and HOG+SVM setup, the HOG cells of a page can be precomputed and stored offline. At test time, computing the descriptor of a window requires only to access the corresponding elements of the precomputed grid, and so the cost is negligible. A similar technique can be applied for the Vinciarelli features by precomputing an integral image of aggregated densities over a document page. At test time, the descriptor for a window can be directly computed based on the precomputed statistics. Unfortunately, such techniques cannot be applied to the FV formulation, and it needs to be calculated independently each time. With our optimized implementation, we can compute approximately 60 FVs per second. Computing the 40,000 descriptors per page would require approximately 700 seconds, which makes it unfeasible. Precomputing the descriptors is also not possible, since only one page would require approximately 4Gb of memory.
- Descriptor comparison. On the case of HOG, HOG+SVM, and FV, comparing descriptors is fast since it is based on dot-products of vectors of same size. The distances between a FV query and 1,000 FV words can be computed in less than 15 ms, and comparing the HOG descriptors is an order of magnitude faster due to their reduced dimensionality. However, comparing the Vinciarelli features is slower since it is based on DTW. Again, using our optimized DTW implementation in C, comparing one query against 1,000 words takes on average 350ms, a hundred times slower than comparing HOGs.

Because of these reasons, the underlying costs of Vinciarelli and FV makes them unfit for a segmentation-free task. However, it is possible to perform the segmentation-free search using the HOG+SVM approach with a reasonable success, and use the FV features to rerank only the best scored candidates.

Influence of SVM solvers. Here we compare the results of the Exemplar SVM over HOG descriptors using a batch solver as was done in [7] and using an SGD implementation based on [1]. In this case, as we did in the previous experiment, we do not use the annotations to segment the words and rely on the sliding-window approach. We do not use neither PQ compression nor reranking or query expansion for this experiment. Results are shown in Table 4.2. We compare the accuracy in mAP for both datasets and compare it with the cosine approach, that does not use any training. We also compare the training time on the GW dataset – training times on the LB dataset are extremely similar. We can observe how the performances of both LIBLINEAR and the SGD implementation are very similar, with differences that are not significative when compared with the cosine approach, *i.e.* the baseline system. However, in terms of speed, the SGD implementation is almost ten times faster than LIBLINEAR. Through the rest of the experiments, we will use the SGD implementation.

Influence of PQ. We study the influence of PQ using different number of subquantizers, from $m = 1$, *i.e.*, one subquantizer of 8 bits for 24 dimensions (compression ration of 1:96), to $m = 6$, *i.e.*, one subquantizer of 8 bits for 4 dimensions (compression ratio of 1:16). We also consider the average time in milliseconds needed to scan one page using a sliding

	GW		LB
	mAP	Time (ms/query)	mAP
Cosine	31.86	-	66.34
LIBLINEAR	38.28	1,090	78.01
SGD	38.16	124	77.91

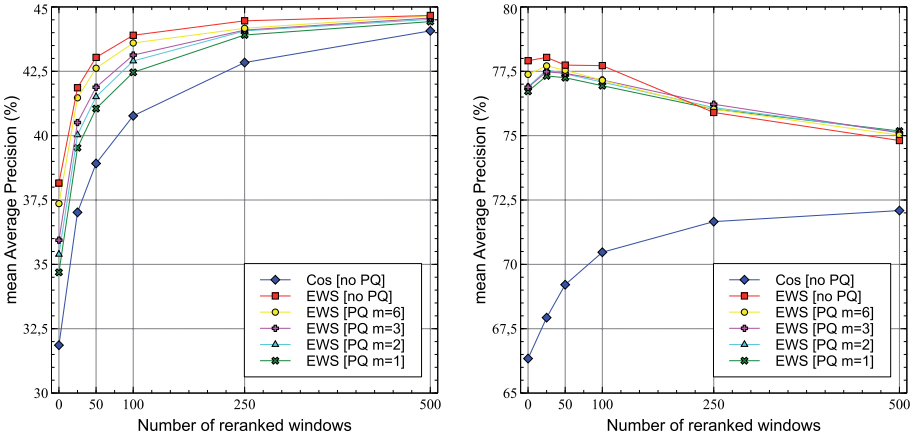
Table 4.2: Comparison of LIBLINEAR and SGD on terms of accuracy in mAP and training time in milliseconds.

window. Results are shown in Table 4.3. After PQ, the accuracy of the methods suffers a small drop, especially when we use a single quantizer for each HOG feature, *i.e.*, $m = 1$. However, as long as we use more quantizers per cell, this drop is reduced until it becomes insignificant. The difference between applying PQ with $m = 6$ and not applying PQ is less than 1% absolute. Moreover, the sliding-window times become between 3 and 15 times faster, depending on the configuration, and a much larger number of documents can fit in memory at the same time. We should also consider that when using PQ one does not need to compute the HOG descriptor of every document for every new query since they are precomputed, saving approximately 500ms per document and query.

	GW			LB
	mAP	ms/doc	docs/GB	mAP
Cosine [no PQ]	31.86	218	$\sim$ 0.25K	66.34
EWS [no PQ]	38.16	218	$\sim$ 0.25K	77.91
EWS [PQ m=6]	37.36	86	$\sim$ 4K	77.38
EWS [PQ m=3]	35.94	45	$\sim$ 8K	76.88
EWS [PQ m=2]	35.39	34	$\sim$ 12K	76.89
EWS [PQ m=1]	34.69	14	$\sim$ 24K	76.72

Table 4.3: Comparison of different numbers of quantizers used in PQ on terms of accuracy in mAP, time to perform a sliding-window search in milliseconds per document, and number of pages that fits in one Gigabyte of memory. Since time and space consumption is extremely similar for both datasets we only report numbers for GW.

Effect of Reranking. We study the effect in performance obtained by the reranking process using FV as representation as a function of the number of the first retrieved windows that have been reranked. Figures 4.4(a) and 4.4(b) show the accuracy in mAP for different configurations of the method for both GW and LB datasets. We plot the performance of the method using cosine similarity and EWS with different configurations of the PQ compression (without compression and compressing using from $m = 1$ to $m = 6$ quantizers). We see that, as the number of windows increases, all the configurations using Exemplar SVMs converge in mAP independently of the amount of compression used, showing that the FV representation can compensate the high compression used during the first ranking step. On GW, accuracy increases as the number of windows increases and reaches a plateau at approximately 45% of mAP using 500 windows. The effect in LB is different. When using the inferior cosine similarity, reranking does indeed improve the results significantly. However, when using the Exemplar model, reranking actually decreases the performance when reranking more than 25 windows. We believe the reason is that, for the rigidity of this dataset, the very



(a) GW (b) LB

Figure 4.4: Retrieval results in mAP for different configurations in the (a) George Washington and (b) Lord Byron datasets.

structured cell of HOG descriptors is more accurate than the FV representation, that only uses weak geometrical structure through the spatial pyramid. Note that although in Table 4.1 we showed that FV obtained better results than HOG, the difference was quite small in this dataset.

Regarding the computational cost of reranking, the time that is needed to extract the FV of a given window and compute the new score similarity with the query is of approximately 20 milliseconds. The cost of performing the actual reranking is negligible since it only involves sorting a few tens or hundreds of values.

Effect of Query Expansion. Finally, we study the influence in performance of the query expansion process, as a function of the number of windows used to expand the set of positive samples, for both combination models proposed, *single-exemplar* and *multi-exemplar*. Here we combine the use of EWS, PQ and reranking previous and posterior to query expansion. We set the the number m of quantizers used in PQ to 3 as a good trade-off between accuracy and time consumption. We also set the number of windows used for the first reranking to 100 on the GW dataset and 25 on the LB dataset. This offers a reasonable trade-off based on the results reported in Figure 4.4. For the second reranking, we use 100 and 250 windows on the GW dataset and 25 on the LB dataset.

We show the results in Table 4.4. We see that expanding the query with the first retrieved windows after reranking leads to improved results. Interestingly, only a small number of windows has to be used: 2 in the GW dataset and 1 in LB; otherwise the results do not improve as much and can even worsen in the case of LB. One of the reasons behind this is that, for many queries, the number of relevant items is very small. On both GW and LB datasets, about 15% of the queries have 2 or less relevant items. On those cases, including more results in the query expansion will inevitably add negative samples to the set $\mathcal{X}$, degrading the results.

Regarding the ways to combine the new samples with the query, *multi-exemplar* has a very slight edge over *single-exemplar*, which is consistent with the main idea of Exemplar SVMs. However, note that *multi-exemplar* has to perform as many Exemplar SVM trainings

	Method configuration		Expanded samples			
			0	1	2	4
GW	EWS RR2: 100	Single-exemplar	43.13	45.55	45.77	44.88
		Multi-exemplar	43.13	45.63	45.94	45.32
	EWS RR2: 250	Single-exemplar	44.10	46.33	46.53	45.62
		Multi-exemplar	44.10	46.35	46.58	45.84
LB	EWS RR2: 25	Single-exemplar	77.71	77.91	76.83	75.01
		Multi-exemplar	77.71	78.35	77.56	76.14

Table 4.4: Influence of the number of examples to expand the query for different number of windows reranked. The number of windows reranked in the reranking previous to query expansion has been fixed to 100 for GW and 25 for LB.

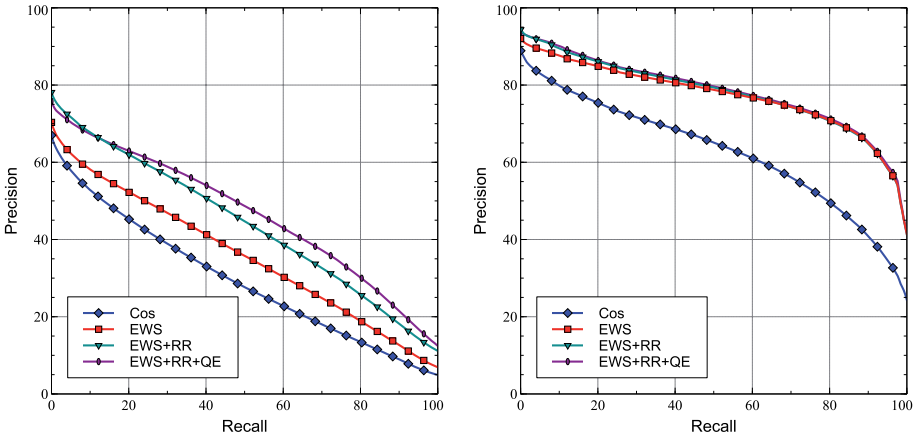
as elements the set of positive samples contains, contrary to *single-exemplar*, which only requires one training. This could make, for some situations, not worth the extra cost.

Finally, in Figure 4.5 we show precision-recall plots of the different approaches that we presented on both GW and LB datasets. We observe how indeed the Exemplar learning has a very large influence in the results on both datasets, showing the importance of learning more discriminative signatures. On GW, the reranking and query expansion also lead to a significant improvement due to the superiority of the FV representation and the variability of the data, while on LB, where the variability is much smaller, reranking and query expansion only bring minimal improvements.

Comparison with other methods. We first compare our results with the segmentation-based methods of [86] and [90]. The authors of [86] report results between 52% and 65% mAP on the GW dataset depending on the particular fold they evaluate with, while [90] reports a 54% mAP, which should be compared to our 46.58% using reranking and query expansion. Note however that these results are not completely comparable since i) they use different query/database partitions, with smaller database sets, which benefits the mAP metric, and ii) they work on already segmented words, *i.e.*, they are not segmentation-free.

Additionally, in order to be able to compare with other segmentation-free methods that reports experiments in these datasets [94, 95, 115], we run again our method using the same final configuration under their evaluation protocols. All these methods include the query in the retrieved results and therefore in the mAP computation. Moreover, [94] uses an overlap over union of 20% with the groundtruth to classify a window as positive, instead of the more traditional 50%, and [115] only considers as queries the words with more than 5 characters. Finally, the work of [44] also reports experiments in GW, but the results are not comparable since it performs a reranking step based on segmentation to avoid substring matching. We show the results in Table 4.5, where we can see that our baseline system already obtains quite reasonable results. Moreover, if we use our full system, including EWS, PQ, reranking and query expansion, we considerably outperform the work of [95] for both GW and LB datasets. With this full system, and using their respective protocols, we also outperform [94] and [115].

Finally, Figure 4.6(a) illustrates some typical failure cases, such as confusions with similar-shaped words, giving too much weight to artifacts in the query word, or retrieving substrings from longer words. We also observe how reranking and query expansion address the first two issues by constructing more discriminative queries that are more informative and more independent of the background. The third problem is one drawback of the



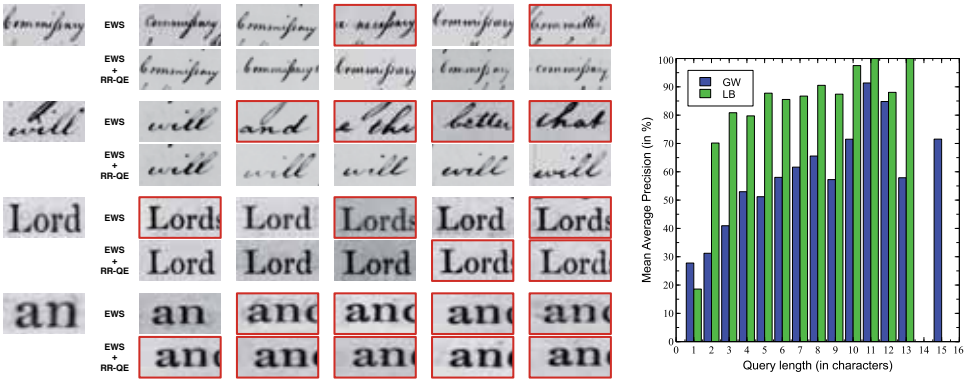
(a) GW (b) LB

Figure 4.5: Precision-recall curves for different configurations in the (a) George Washington and (b) Lord Byron datasets.

	GW	LB
Rusiñol et al. [95]	30.42	42.83
Cos noPQ	48.66	74.04
EWS+PQ	51.88	84.34
EWS+PQ+RR	57.46	84.51
EWS+PQ+RR+qe+RR2	59.13	84.04
Rothacker et al. [94] (overlap 20%)	61.10	–
EWS+PQ (overlap 20%)	59.51	–
EWS+PQ+RR+qe+RR2 (overlap 20%)	68.88	–
Zhang and Tan [115] (queries $>$ 5 characters)	62.47	–
EWS+PQ (queries $>$ 5 characters)	72.85	–
EWS+PQ+RR+qe+RR2 (queries $>$ 5 characters)	82.23	–

Table 4.5: Retrieval performance in mAP and comparison with state-of-the-art when query is included in the results. Methods have been set to the best parameters.

segmentation-free, sliding-window methods since, as opposed to DTW, they do not penalize matching a short word with a substring of a long word. As observed in Figure 4.6(b), this can very significantly reduce the mAP of short queries. However, this may be also seen as an interesting feature that could be exploited for sub-word searches.



(a) (b)

Figure 4.6: a) Failure cases. For every query we show a first row with the results retrieved by EWS and a second row with the results retrieved by EWS combined with reranking and query expansion. First query: words have a very similar shape. Second query: an artifact in the query leads to results with the same artifact. Third and fourth query: we detect the query word as a substring of a longer word. This is common when querying short words. Query expansion and reranking are able to alleviate some of the problems. b) Mean Average Precision as a function of the query length for the system combining EWS, PQ, reranking and query expansion.

4.8 Conclusions

In this paper we have shown how a combination of HOG descriptors and sliding windows can be used to perform segmentation-free, unsupervised word spotting, both on handwritten and machine-printed text. This method can be extended using Exemplar SVMs to represent the queries, improving the results at a minimum extra cost at query time. We have shown how the HOG descriptors can be aggressively compressed with Product Quantization with only a small loss in accuracy. Finally, we have shown that results can be improved by, first using more informative and discriminative features in a reranking step of the best windows retrieved, and second using some of these windows to expand the Exemplar SVM training set and improve the query representation. We have obtained excellent results when comparing to other segmentation-free methods in the literature. We have published the MATLAB code implementation for training and testing the Exemplar Word Spotting [5], as well as experiments with other datasets [35], in the hope that it would ease the comparison for new works on word spotting.

Finally, we would like to note that the variability of the writing style on the datasets that have been used in the experiments is very small. This variability is usually much higher in a multi-writer scenario. For example, two writers may write the same word with very different widths, and only one window size will not be able to correctly capture both of them. We

believe that, as presented, our method would have difficulties in this scenario. To overcome this problem, a possible solution would be to severely deform the query varying the stretch and the slant, and learn several Exemplar SVMs as in [67]. This would increase the query time, but not the memory required to store the datasets.

Chapter 5

Word Spotting and Recognition with Embedded Attributes[1]

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.

5.1 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.

[1] This chapter corresponds to the publication "J. Almazán, A. Gordo, A. Fornés and E. Valveny. Word Spotting and Recognition with Embedded Attributes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2014".

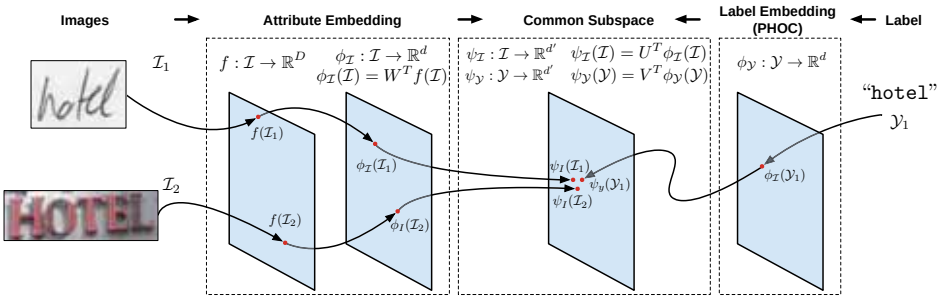


Figure 5.1: Overview of the proposed method. Images are first projected into an attributes space with the embedding function $\phi_{\mathcal{I}}$ after being encoded into a base feature representation with f. At the same time, labels strings such as "hotel" are embedded into a label space of the same 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.

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 [15, 69, 78, 79] could be used, but we consider that out of the scope of this work[2].

Traditionally, word spotting and recognition have focused on document images [30, 36, 38, 68, 87, 88, 89, 90, 113], 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, 73, 78, 79, 91, 110], 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 of very fine-grained, zero-shot classification, where we are interested in classifying a word

[2]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.

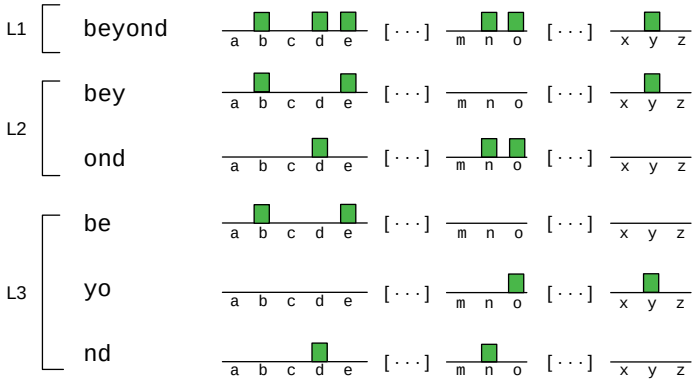


Figure 5.2: PHOC histogram of a word at levels 1, 2, and 3. The final PHOC histogram is the concatenation of these partial histograms.

image into (potentially) hundreds of thousands of classes, for which we may not have seen any training example. The examples on Figs. 5.9 and 5.10 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 [59, 64] 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 5.2). 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 5.1.

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 [48] could now be used to perform spotting in very large datasets. To the best of our knowledge, we 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 5.2, we review the related work in word spotting and recognition. In Section 5.3 we describe how to encode our images into a low-dimensional attribute representation. In Section 5.4 we describe the proposed common subspace learning. Section 5.5 suggests some practices to learn the attributes space and the common subspace when training data is scarce. Section 5.6 deals with the experimental validation of our approach. Finally, Section 5.7 concludes the paper.

This paper is an extended version of the work initially published in ICCV 2013 [8]. Novel contributions include a better base feature representation tailored for word images (Section 5.3), a more detailed formulation of the common subspace problem (Section 5.4), a bagging approach to learn with scarce data (Section 5.5), and evaluation on recognition tasks, as well as new datasets, including two popular benchmarks based on natural images (Section 5.6).

5.2 Related Work

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

5.2.1 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 [36, 38, 68, 87, 88, 89, 90], 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 [36, 38, 66, 87, 89, 90, 113].

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 [113]), 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.* [68] and Keaton *et al.* [52]. In [68], a distance between binary word images is defined based on the result of XORing the images. In [52], 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 [83] exploits the Fisher kernel framework [45] 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 [7, 39, 95]. Gatos *et al.* [39] perform a template matching of block-based image descriptors, Rusiñol *et al.* [95] use an aggregation of SIFT descriptors into a bag of

visual words to describe images, while Almazán *et al*. [7] use HOG descriptors [26] combined with an exemplar-SVM framework [67]. 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 [7, 95] 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 [30]. 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.

5.2.2 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 [110] or recognition in signs harvested from Google Images [72]. 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*. [110] 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 [78, 79] also address this problem of end-to-end word recognition. In [78] 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 [79] 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*. [15] 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*. [72, 73, 74]. In [72] 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 [74] 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, [73] 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*. [40] 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.

5.2.3 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 [32, 56, 93, 109]. 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 [36, 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 [2, 14, 91], 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 [91] 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[3]. 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 [84]) 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

[3]Both the conference version of this paper [8] and the work of Rodriguez and Perronnin [91] appeared simultaneously.

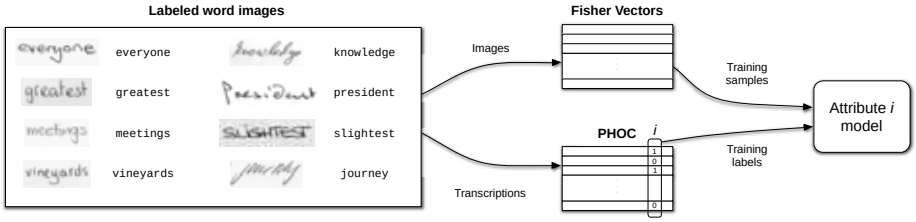


Figure 5.3: Training process for i-th attribute model. An SVM classifier is trained using the Fisher vector representation of the images and the i-th value of the PHOC representation as label.

that embeds our images into the common subspace can be decomposed into a matrix that projects the images into an attributes space.

5.3 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 [90], 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 [32], and have recently gained a lot of popularity for image retrieval and classification tasks [32, 56, 104, 106, 109]. Attributes have also shown ability to transfer information in zero-shot learning settings [32, 56, 93, 109] 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.

5.3.1 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[4], 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. 5.2). 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 5.4). 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 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.

5.3.2 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 5.3. To represent the images, we use Fisher vectors (FV) [84], a state-of-the-art encoding method [18] which works well with linear classifiers. The FV can be seen as a bag of visual words [25] that encodes not only word counts but also higher-order statistics. In a nutshell, at training time, low-level descriptors

[4]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.

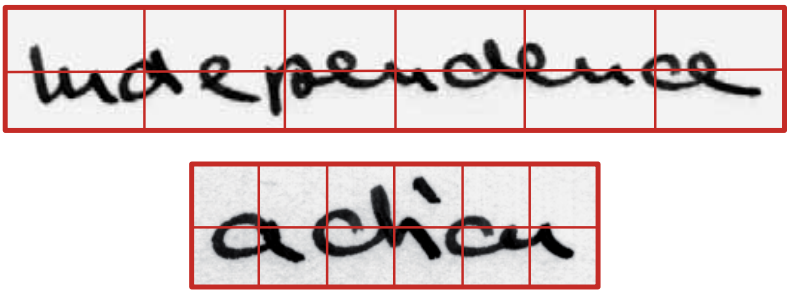


Figure 5.4: Spatial pyramids on word images. The sizes and contents of each spatial region are very dependent on the length of the word.

(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.

5.3.3 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 [57]. 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. 5.4.

A different approach that works well in combination with FVs was proposed by Sánchez et al [98]. 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

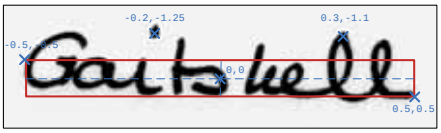


Figure 5.5: Word image and the automatically adjusted reference box that defines the coordinates system.

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 5.5 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 5.6 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 5.6 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 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.

5.4 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 [101], 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

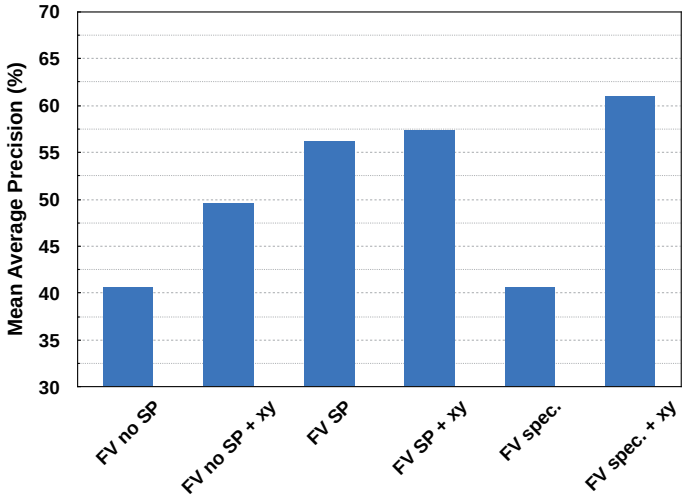


Figure 5.6: Results of the attributes classifiers for different Fisher vector configurations on the IAM dataset.

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 $A = \phi_{\mathcal{I}}(\mathcal{I}) \in \mathbb{R}^{d \times N}$ 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 $P \in \mathbb{R}^{d \times d}$, and to minimize the distance across all the samples and their labels,

$$\begin{aligned} &\underset{P}{\operatorname{argmin}} \sum_{i}^{N} \frac{1}{2} F(\mathcal{I}_i, \mathcal{Y}_i; P) + \frac{1}{2}\Omega(P) \quad = \\ &\underset{P}{\operatorname{argmin}} \frac{1}{2}||P^T A - B||_F^2 + \frac{1}{2}\Omega(P), \end{aligned} \tag{5.1}$$

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. 5.7). 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 $U, V \in \mathbb{R}^{d \times d'}$ 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,

$$\begin{aligned} &\underset{U,V}{\operatorname{argmin}} \sum_i^N \frac{1}{2}\hat{F}(\mathcal{I}_i, \mathcal{Y}_i; U, V) + \frac{1}{2}\Omega(U) + \frac{1}{2}\Omega(V) \quad = \\ &\underset{U,V}{\operatorname{argmin}} \frac{1}{2}||U^T A - V^T B||_F^2 + \frac{1}{2}\Omega(U) + \frac{1}{2}\Omega(V), \\ &\quad \text{s.t.} \\ &\qquad \psi_{\mathcal{I}}(\mathcal{I})\psi_{\mathcal{I}}(\mathcal{I})^T = I \\ &\qquad \psi_{\mathcal{Y}}(\mathcal{Y})\psi_{\mathcal{Y}}(\mathcal{Y})^T = I, \end{aligned} \tag{5.2}$$

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:

$$\begin{aligned} \lambda(AA^T + \alpha I)u_k &= AB^T v_k \\ \lambda(BB^T + \alpha I)v_k &= BA^T u_k, \end{aligned} \tag{5.3}$$

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:

$$AB^T(BB^T + \alpha I)^{-1}BA^T u_k = \lambda^2(AA^T + \alpha I)u_k \tag{5.4}$$

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 [43]. CCA is a tool to exploit information available from different data sources, used for example in retrieval [42] and clustering [16]. In [41], 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 [41] 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 (5.2) aims at minimizing the distance between the images and 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.* [111]. However, we found that

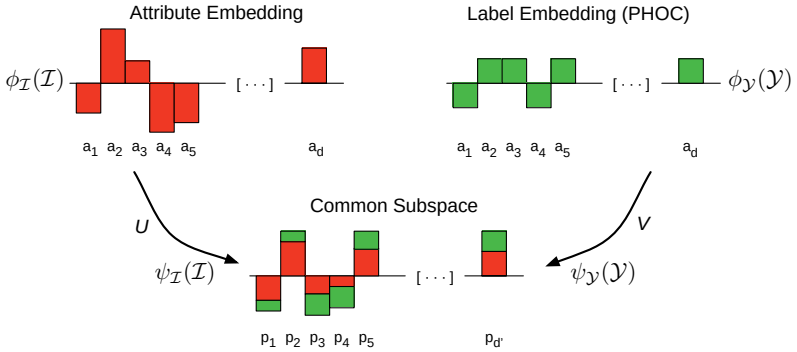


Figure 5.7: Projection of predicted attribute scores and attributes ground truth into a more correlated subspace with CSR.

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 [91], 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 [41]: we explicitly embed the data using a random Fourier feature (RFF) mapping [85], 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.

5.5 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.

Algorithm 1 Learn attribute model with bagging

Input: Training data $X \in \mathbb{R}^{D \times N}$
Input: Training labels $Y \in \{0,1\}^N$
Input: Number of folds F
Output: Model $W \in \mathbb{R}^D$
Output: Training data embedded onto the attribute space $A \in \mathbb{R}^N$

$W = \text{zeros}(1, D)$
$A = \text{zeros}(1, N)$
$count = \text{zeros}(1, N)$
$f = 1$
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

5.6 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.

5.6.1 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 [5] [71] 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.

The George Washington (GW) dataset[6] [87] 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 [3, 38] 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[7] [72] 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[8] [110] 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.

[5] http://www.iam.unibe.ch/fki/databases/iam-handwriting-database

[6] http://www.iam.unibe.ch/fki/databases/iam-historical-document-database/washington-database

[7] http://cvit.iiit.ac.in/projects/SceneTextUnderstanding/IIIT5K.hmtl

[8] http://vision.ucsd.edu/~kai/svt/

Table 5.1: 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

5.6.2 Implementation Details

We use Fisher vectors [84] 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*. [99] 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. 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 [40] or by using in-house datasets [15].

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 [47] 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.

5.6.3 Word Spotting

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 5.2, 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.* [3], 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 [38] 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.* [38] 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 [91]. 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 5.3.

Results

The results of our approach on the word spotting task are shown on Table 5.1. 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.

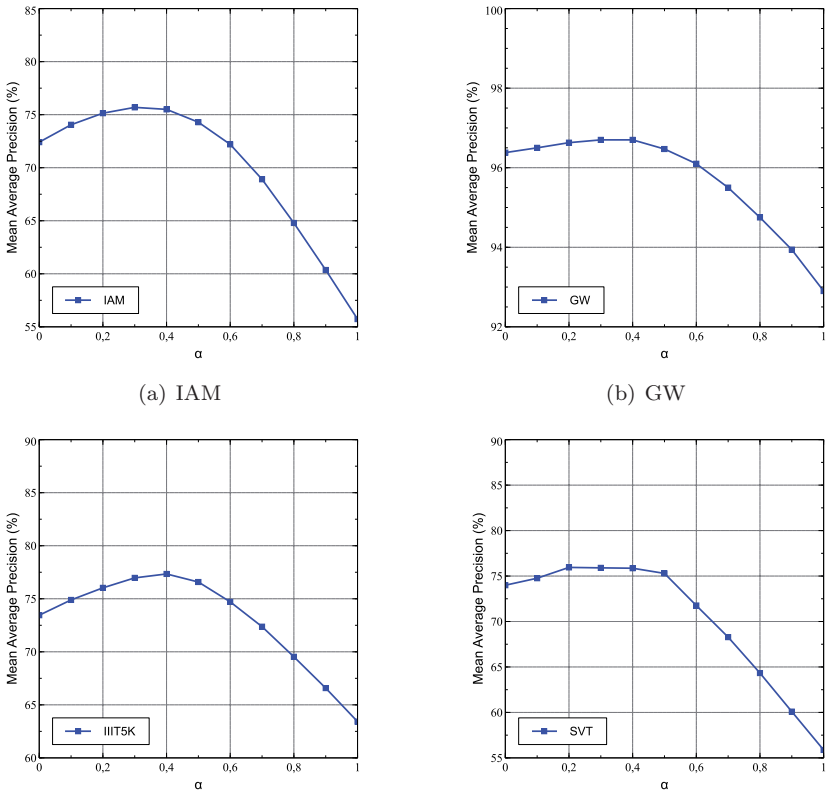
CSR vs KCSR. The kernelized version of CSR obtains results very similar to CSR. Our intuition is that, due to the higher dimensionality of Random Fourier Features, the method is more prone to overfit, and requires more training data to show its benefits. Indeed, in the IAM dataset, which contains larger 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 5.8 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 5.2), we first focus on QBE and compare our approach with the FV baseline and a DTW approach based on *Vinciarelli* [107] features. On GW, we report the results of [90] on DTW as well as their results with semi-continuous HMM (SC-HMM). Although the results of [90] are not exactly comparable, since partitions are different (we followed [3, 38] instead of [90]), 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.

Table 5.2: 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 [90]	50.00
			SC-HMM [90]	53.00
	Proposed (Platts)	48.09	Proposed (Platts)	93.04
	Proposed (KCSR)	**55.73**	Proposed (KCSR)	**92.90**
QBS	cHMM [36, 38]	36.00	cHMM [36, 38]	60.00
			Aldavert *et al.* [3]	76.20 (56.54*)
	Frinken *et al.* [38]	78.00	Frinken *et al.* [38]	84.00
	Proposed (KCSR)	**80.64**	Proposed (KCSR)	**93.93 (91.11*)**



(a) IAM (b) GW

(c) IIIT5K (d) SVT

Figure 5.8: Hybrid spotting results with KCSR as a function of the weight α assigned to the visual part of the query.

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 [83], 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 [38] and [3], as well as with the character HMM approach of [36] as evaluated in [38]. 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 [3] (76.2 vs 56.54), showing a much nicer adaptation to unseen words.

We also compare our approach on natural images, see Table 5.3. We compare ourselves with the recent label-embedding method of [91], 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 5.3: Word spotting comparison with the state-of-the-art in IIIT5K dataset for the QBE task.

Method	Top-1 acc.
FV	40.70
DTW [91]	37.00
Rodríguez and Perronnin [91]	43.70
Proposed (KCSR)	**72.28**

Qualitative results. We finally show qualitative results with samples from the IAM and IIIT5K datasets on Figure 5.9. 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").

5.6.4 Word Recognition

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 [30]. 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

Figure 5.9: Qualitative results on word spotting on the IAM and IIIT5K. Relevant words to the query are outlined in green.

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.

Results

The results on word recognition on IAM are on Table 5.4, where we compare with the state-of-the-art approach of [30], 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 [30], our results are competitive without performing any costly preprocessing on the images and with much faster recognition speeds.

Table 5.4: Recognition error rates on the IAM dataset.

Method	WER	CER
España-Bosquera *et al.* [30]	**15.50**	**6.90**
Proposed (KCSR)	20.01	11.27

Table 5.5 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 [15] approach. However, [15] 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 5.10 shows some qualitative results on image recognition with samples from the IAM, IIIT5K and SVT datasets. Common errors include very similar-looking words

Table 5.5: Recognition results on the IIIT5K and SVT dataset. Accuracy measured as precision at 1.

Dataset	Method	$\|y\| = 50$	$\|y\| = 1000$
IIIT5K	Mishra *et al.* [72]	64.10	57.50
	Rodríguez and P. [91]	76.10	57.40
	Proposed (KCSR)	**88.57**	**75.60**
SVT	ABBY [40]	35.00	-
	Mishra *et al.* [73]	73.26	-
	Goel *et al.* [40]	77.28	-
	PhotoOCR [15]	**90.39**	-
	Proposed (KCSR)	87.01	-

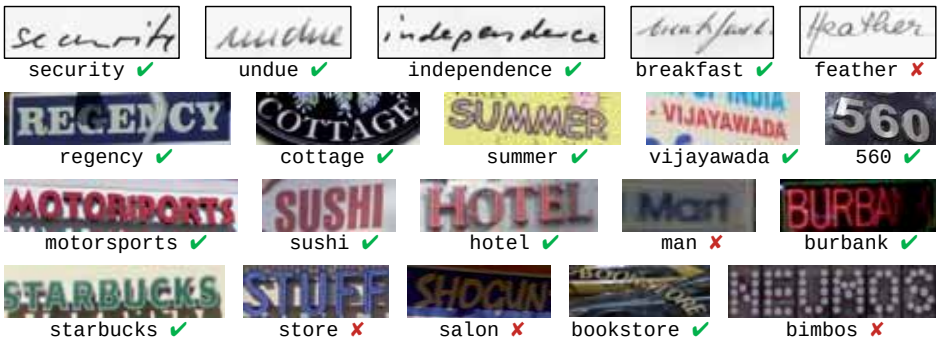


Figure 5.10: Qualitative results on word recognition on the IAM, IIIT5K, and SVT datasets.

("Heather" and "feather"), low-resolution images ("Mart" and "man"), or unorthodox font styles ("Neumos" vs "bimbos").

5.6.5 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 [38] takes a few milliseconds to process a single line in the IAM for the QBS task. PhotoOCR [15] 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 [79] takes 35 seconds to locate and recognize the words in an image, and the same task takes around 8 seconds with the method in [74].

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 [15] 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.

5.7 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 [15]. We want to study how we could learn on individual segmented characters (using existing datasets such as Char74K [27]) 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.

Chapter 6

Conclusions

Representing handwritten shapes has become a very important problem in the Document Image Analysis field since it is a core aspect of many tasks. It is also a very challenging problem and therefore many effort has been put in the last decades to solve it. One of the consequences of working with handwritten shapes is that representations needs to be *robust to deformations*, *discriminative*, and, due to the recent increase of digitized documents, *efficient*. Most available methods do not satisfy these conditions: they propose complex variable-length representations that are costly to match or simple representations that have difficulties to adapt to large variability. Still, a lot of research remains to be done in order to achieve this objective.

Through this dissertation we have first studied the general problem of learning to represent handwritten shapes aimed at matching and recognition tasks. Motivated by the conditions mentioned in the previous paragraph, we first proposed a shape descriptor based on a deformable grid. This non-rigid grid is able to deal with large deformations by adapting to the shape and then represent it by extracting density-based features in the final locations of the grid's cells. The result is a fixed-length representation that can be efficiently compressed, indexed or used to feed a classifier. In chapter 2, we proposed and compared two different deformation processes for this non-rigid grid and showed how this deformation efficiently adapts to the handwritten shapes and considerably improves the performance of the rigid version. Then we proposed to use this novel descriptor to learn statistical models, based on the Active Appearance Model, that jointly learns the variability in structure and texture of a given shape class. Experiments showed the ability of this model to learn *structure* and *texture* variability, achieving a satisfactory performance in handwritten digit and handwritten symbol recognition. Finally, we proposed to extend the well-known HOG descriptor, which is based on a rigid grid, for the specific case of handwritten shapes by integrating it with our deformable grid. We showed how this combination improved the results of the original descriptor and other variable-length representations in symbol recognition and word retrieval tasks.

Motivated by the results obtained in word retrieval with gradient-based features, we then focused on the specific task of learning to represent words for spotting and recognition. Most methods proposed for word spotting have followed a common pipeline: segment word candidates, represent this candidates with variable-length representations and rank them according to computationally expensive distances with the query, *e.g.*, Dynamic Time Warping or Hidden Markov Models. In chapter 4 we explored the use of HOG descriptors and sliding windows in word spotting and we showed how they can be used to perform

segmentation-free, unsupervised word spotting, both on handwritten and machine-printed text. We also showed how can be extended using Exemplar SVMs to represent the queries and how the HOG descriptors can be aggressively compressed with Product Quantization with only a small loss in accuracy but a high reduction of computational cost and memory space. Finally, we showed that results can be improved by, first using more informative and discriminative features in a reranking step of the best windows retrieved, and second using some of these windows to expand the Exemplar SVM training set and improve the query representation. We obtained excellent results in two public datasets when comparing to other segmentation-free methods in the literature.

However, we noted that the variability of the writing style on these datasets is very small compared to a multi-writer scenario, and some preliminary experiments showed us that the method in chapter 4, as presented, would have some difficulties. Through chapter 5 we explored this problem and proposed an attribute-based approach 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 style. Besides the handwriting domain, we noted that this method can be also directly applied to the problem of text in natural images, showing that it is also independent of the font style, illumination or capture angle. This attributes representation leads to a low-dimensional, 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 finally showed results in four public datasets of documents and natural images that outperform state-of-the-art approaches, and showed that the proposed attribute-based representation is well-suited for word searches, whether they are images or strings, in handwritten and natural images.

6.1 Continuation Lines

Finally, we provide some possible continuation lines for the methods proposed in this thesis, which extend those already introduced in the closing sections of the different papers.

Non-rigid descriptor:

In chapters 2 and 3, we have shown the ability of the proposed non-rigid descriptor to capture the structure of the shape and deal with large deformations, and how different features, namely density and gradient-based, can be indistinctly integrated. Then, in chapter 4, we have analyzed the performance of different features for the problem of word spotting, showing a superiority for representations encoding SIFT descriptors, concretely Fisher Vector. We believe that this analysis could be extended for different problems, and therefore design specific representations for specific tasks. In the case of Fisher Vector, it could be efficiently integrated following an integral image approach, where Fisher Vectors are precomputed at different bins and then aggregated. Moreover, these features can be combined at different levels of the region partitioning procedure. In the experiments carried out we only used the lowest lever of the pyramid. However, we believe that extracting features at different levels, where coarser features are extracted at the first levels, and more fine-grained features at deeper levels, will produce a better representation of the shape.

Word spotting with Exemplar SVMs:

In chapter 3, we have shown how the proposed nrHOG descriptor improves the performance of the original HOG descriptor when dealing with handwritten words. However, for efficiency

reasons, HOG was finally used in the word spotting method proposed in chapter 4, since the rigid grid seemingly integrates with the sliding window framework. The major problem in the combination of the nrHOG descriptor and the sliding window-based search comes with the fact that the centroids of the deformable grid should be computed for every possible window, which is not feasible in most of the cases. However, this problem could be addressed by assuming that the deformation obtained in a query word should be similar to those obtained in words of the same class appearing in the dataset. In this case, we could compute the deformation of the grid only once in the query, and fix this grid in the sliding window search.

Another problem that we noted is that, as presented, the Exemplar Word Spotting method would have difficulties when dealing with datasets where the variability is much higher, as for example, a multi-writer scenario: two writers may write the same word with very different widths, and only one window size will not be able to correctly capture both of them. To overcome this problem, a possible solution would be to severely deform the query varying the stretch and the slant, and applying elastic deformations, and then learn several Exemplar SVMs as in [67]. This would increase the query time, but not the memory required to store the datasets.

Words with Embedded Attributes:

In preliminary experiments we noted the importance of having a large and representative training set in order to produce better attributes. Having a reduced dataset makes that some character attributes does not have enough positive examples or, in some cases, any example at all. We alleviated this problem by proposing the bagging approach (Section 5.5) for learning with scarce data, which considerably improved the results. However, we noted that in some cases, where the training set is very limited, it may not be enough .

Producing large training sets is usually a tedious and expensive task. Therefore, and similarly to Jaderberg *et al.* [46], we believe that using synthetically generated training data would be a possible solution in order to have larger and more representative sets. As in [46], different algorithms could be used to generate both realistic and diverse word image representations of text strings. In this way, besides totally random generation of words, we could also focus on generating more training examples for those characters attributes that most need of them. In [46] they only focus on the scene text problem, but a similar approach, using handwriting-based fonts, could be also applied to the problem of handwritten document images. We believe that this would definitely produce better attribute models, and therefore, considerably increase the results.

Publications

The following publications are a consequence of the research carried out during the elaboration of this thesis and give an idea of the progression that has been achieved.

Journals

- J. Almazán, A. Gordo, A. Fornés and E. Valveny. Word Spotting and Recognition with Embedded Attributes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2014.
- J. Almazán, A. Gordo, A. Fornés and E. Valveny Segmentation-Free Word Spotting with Exemplar SVMs. *Pattern Recognition*, 2014.
- J. Almazán, A. Fornés and E. Valveny, A non-rigid appearance model for shape description and recognition. *Pattern Recognition*, 2012.

International Conferences and Workshops

- D. Fernández, J. Almazán, N. Cirera, A. Fornés and J. Lladós. BH2M: the Barcelona Historical Handwritten Marriages database. In *International Conference on Patter Recognition*, 2014.
- P. Riba, J. Almazán, A. Fornés, D. Fernández-Mota, E. Valveny and J. Lladós. e-Crowds: a mobile platform for browsing and searching in historical demography-related manuscripts. In *International Conference on Frontiers in Handwriting Recognition*, 2014.
- J. Almazán, A. Gordo, A. Fornés and E. Valveny. Word Spotting with Corrected Attributes. In *International Conference on Computer Vision*, 2013.
- J. Almazán, A. Fornés and E. Valveny. Deformable HOG-based Shape Descriptor. In *International Conference on Document Analysis and Recognition*, 2013.
- D. Karatzas, F. Shafait, S. Uchida, M. Iwamura, S. Robles, J. Mas, D. Fernández, J. Almazán, L.P. de las Heras. ICDAR 2013 Robust Reading Competition. In *International Conference on Document Analysis and Recognition*, 2013.
- J. Almazán, A. Gordo, A. Fornés and E. Valveny. Efficient Exemplar Word Spotting. In *British Machine Vision Conference*, 2012.
- J. Almazán, D. Fernández, A. Fornés, J. Lladós and E. Valveny. A Coarse-to-Fine Approach for Handwritten Word Spotting in Large Scale Historical Documents Collection. In *International Conference on Frontiers in Handwriting Recognition*, 2012.

- A. Fornés, V. Frinken, A. Fischer, J. Almazán, G. Jackson and H Bunke. A Keyword Spotting Approach Using Blurred Shape Model-Based Descriptors. In *Workshop on Historical Document Imaging and Processing*, 2011.
- J. Almazán, A. Fornés and E. Valveny. A Non-Rigid Feature Extraction Method for Shape Recognition. In *International Conference on Document Analysis and Recognition*, 2011.
- J. Almazán, A. Fornés and E. Valveny. Deforming the Blurred Shape Model for Shape Description and Recognition. In *Iberian Conference on Pattern Recognition and Image Analysis*, 2011.

Bibliography

[1] Z. Akata, F. Perronnin, Z. Harchaoui, and C. Schmid. Good practice in large-scale learning for image classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2013. `http://lear.inrialpes.fr/src/jsgd/`.

[2] Z. Akata, F. Perronnin, Z. Harchaoui, and C. Schmid. Label-embedding for attribute-based classification. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2013.

[3] D. Aldavert, M. Rusiñol, R. Toledo, and J. Lladós. Integrating Visual and Textual Cues for Query-by-String Word Spotting. In *International Conference on Document Analysis and Recognition*, 2013.

[4] J. Almazán, A. Fornés, and E. Valveny. A non-rigid feature extraction method for shape recognition. In *International Conference on Document Analysis and Recognition*, 2011.

[5] J. Almazán and A. Gordo. EWS: Exemplar Word Spotting library. `http://almazan.github.io/ews`.

[6] J. Almazán and A. Gordo. WATTS: Words with Attributes library. `http://almazan.github.io/watts`.

[7] J. Almazán, A. Gordo, A. Fornés, and E. Valveny. Efficient exemplar word spotting. In *British Machine Vision Conference*, 2012.

[8] J. Almazán, A. Gordo, A. Fornés, and E. Valveny. Handwritten word spotting with corrected attributes. In *International Conference on Computer Vision*, 2013.

[9] J. Almazán, E. Valveny, and A. Fornés. Deforming the blurred shape model for shape description and recognition. In *Pattern Recognition and Image Analysis, Lecture Notes in Computer Science, 2011*, volume 6669, pages 1–8, 2011.

[10] R. Arandjelović and A. Zisserman. Multiple queries for large scale specific object retrieval. In *British Machine Vision Conference*, 2012.

[11] R. Arandjelović and A. Zisserman. Three things everyone should know to improve object retrieval. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[12] S. Barrat and S. Tabbone. A bayesian network for combining descriptors: application to symbol recognition. *International Journal on Document Analysis and Recognition*, 13:65–75, 2010.

[13] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2002.

[14] S. Bengio, J. Weston, and D. Grangier. Label embedding trees for lagre multi-class tasks. In *Advances in Neural Information Processing Systems*, 2010.

[15] A. Bissacco, M. Cummins, Y. Netzer, and H. Neven. PhotoOCR: Reading Text in Uncontrolled Conditions. In *International Conference on Computer Vision*, 2013.

[16] M. B. Blaschko and C. H. Lampert. Correlational spectral clustering. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[17] H. Bunke. Attributed programmed graph grammars and their application to schematic diagram interpretation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1982.

[18] K. Chatfield, V. Lempitsky, A. Vedaldi, and A. Zisserman. The devil is in the details: an evaluation of recent feature encoding methods. In *British Machine Vision Conference*, 2011.

[19] K. Cheung, D. Yeung, and R. Chin. A bayesian framework for deformable pattern recognition with application to handwritten character recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(2):1382–1387, 1998.

[20] H. Chui and A. Rangarajan. A new point matching algorithm for non-rigid registration. *Computer Vision and Image Understanding*, 89:114–141, 2003.

[21] O. Chum, A. Mikulík, M. Perdoch, and J. Matas. Total recall II: Query expansion revisited. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

[22] O. Chum, J. Philbin, J. Sivic, M. Isard, and A. Zisserman. Total recall: Automatic query expansion with a generative feature model for object retrieval. In *European Conference on Computer Vision*, 2007.

[23] T. Cootes, G. Edwards, and C. Taylor. Active appearance models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(6):681–685, 2001.

[24] 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, European Conference on Computer Vision*, 2004.

[25] 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, European Conference on Computer Vision*, 2004.

[26] N. Dalal and B. Triggs. Histograms of Oriented Gradients for Human Detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2005.

[27] T. de Campos, B. Babu, and M. Varma. Character recognition in natural images. In *International Conference on Computer Vision Theory and Applications*, 2009.

[28] M. Douze, A. Ramisa, and C. Schmid. Combining attributes and fisher vectors for efficient image retrieval. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 745–752, 2011.

[29] S. Escalera, A. Fornés, O. Pujol, P. Radeva, and J. Lladós. Blurred shape model for binary and grey-level symbol recognition. *Pattern Recognition Letters*, 2009.

[30] S. España-Bosquera, M. Castro-Bleda, J. Gorbe-Moya, and F. Zamora-Martinez. Improving offline handwritten text recognition with hybrid HMM/ANN models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2011.

[31] R. Fan, K. Chang, C. Hsieh, X. Wang, and C. Lin. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 2008.

[32] A. Farhadi, I. Endres, D. Hoiem, and D. Forsyth. Describing objects by their attributes. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[33] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramaman. Object detection with discriminatively trained part based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2010.

[34] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, 2010.

[35] D. Fernández-Mota, J. Almazán, N. Cirera, A. Fornés, and J. Lladós. BH2M: the Barcelona Historical Handwritten Marriages database. In *International Conference on Pattern Recognition*, 2014.

[36] A. Fischer, A. Keller, V. Frinken, and H. Bunke. HMM-based word spotting in handwritten documents using subword models. In *International Conference on Pattern Recognition*, 2010.

[37] A. Fischer, A. Keller, V. Frinken, and H. Bunke. Lexicon-free handwritten word spotting using character HMMs. *PRL*, 2012.

[38] 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*, 2012.

[39] B. Gatos and I. Pratikakis. Segmentation-free word spotting in historical printed documents. In *International Conference on Document Analysis and Recognition*, 2009.

[40] V. Goel, A. Mishra, K. Alahari, and C. V. Jawahar. Whole is Greater than Sum of Parts: Recognizing Scene Text Words. In *International Conference on Document Analysis and Recognition*, 2013.

[41] Y. Gong, S. Lazebnik, A. Gordo, and F. Perronnin. Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2012.

[42] D. R. Hardoon, S. Szedmak, and J. Shawe-Taylor. Canonical correlation analysis; an overview with application to learning methods. Technical report, Department of Computer Science, Royal Holloway, University of London, 2003.

[43] H. Hotelling. Relations between two sets of variants. *Biometrika*, 1936.

[44] N. R. Howe. Part-Structured Inkball Models for One-Shot Handwritten Word Spotting. In *International Conference on Document Analysis and Recognition*, 2013.

[45] T. Jaakkola and D. Haussler. Exploiting generative models in discriminative classifiers. In *Advances in Neural Information Processing Systems*, 1999.

[46] M. Jaderberg, K. Simonyan, A. Vedaldi, and A. Zisserman. Synthetic Data and Artificial Neural Networks for Natural Scene Text Recognition. In *Advances in Neural Information Processing Systems*, 2014.

[47] H. Jégou and O. Chum. Negative evidences and co-occurrences in image retrieval: the benefit of pca and whitening. In *European Conference on Computer Vision*, 2012.

[48] H. Jégou, M. Douze, and C. Schmid. Product quantization for nearest neighbor search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2011.

[49] H. Jégou, M. Douze, C. Schmid, and P. Pérez. Aggregating local descriptors into a compact image representation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

[50] H. Jégou, R. Tavenard, M. Douze, and L. Amsaleg. Searching in one billion vectors: re-rank with source coding. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2011.

[51] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 1(4):321–331, 1988.

[52] P. Keaton, H. Greenspan, and R. Goodman. Keyword spotting for cursive document retrieval. In *Document Image Analysis*, 1997.

[53] D. Keysers, T. Deselaers, and C. Gollan. Deformation models for image recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(8):1422–1435, 2007.

[54] A. Khotanzad and Y. Hong. Invariant image recognition by zernike moments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1990.

[55] S. Kuo and O. Agazzi. Keyword spotting in poorly printed documents using pseudo 2-d hidden markov models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(8):842–848, 1994.

[56] C. H. Lampert, H. Nickisch, and S. Harmeling. Learning to detect unseen object classes by between-class attribute transfer. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[57] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: spatial pyramid matching for recognizing natural scene categories. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2006.

[58] Y. Lecun and C. Cortes. The mnist database of handwritten digits., `http://yann.lecun.com/exdb/mnist/`.

[59] C. Leslie, E. Eskin, and W. Noble. The spectrum kernel: A string kernel for SVM protein classification. In *Pacific Symposium on Biocomputing*, 2002.

[60] Y. Leydier, A. Ouji, F. Lebourgeois, and H. Emptoz. Towards an omnilingual word retrieval system for ancient manuscripts. *Pattern Recognition*, 42, 2009.

[61] J. LLadós, E. Martí, and J. Villanueva. Symbol recognition by error-tolerant subgraph matching between region adjacency graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2001.

[62] J. Lladós, M. Rusiñol, A. Fornés, D. Fernández, and A. Dutta. On the influence of word representations for handwritten word spotting in historical documents. *International Journal of Pattern Recognition and Artificial Intelligence*, 2012.

[63] J. LLadós, E. Valveny, G. Sánchez, and E. Martí. Symbol recognition: Current advances and perspectives. *Lecture Notes in Computer Science*, 2390:104–127, 2002.

[64] H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, and C. Watkins. Text classification using string kernels. *Journal of Machine Learning Research*, 2002.

[65] D. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[66] S. Lu, L. Li, and C. L. Tan. Document image retrieval through word shape coding. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2008.

[67] T. Malisiewicz, A. Gupta, and A. Efros. Ensemble of Exemplar-SVMs for object detection and beyond. In *International Conference on Computer Vision*, 2011.

[68] R. Manmatha, C. Han, and E. M. Riseman. Word spotting: A new approach to indexing handwriting. In *IEEE Conference on Computer Vision and Pattern Recognition*, 1996.

[69] R. Manmatha and J. Rothfeder. A scale space approach for automatically segmenting words from historical handwritten documents. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2005.

[70] U.-V. Marti and H. Bunke. Using a statistical language model to improve the performance of an HMM-based cursive handwriting recognition systems. *International Journal of Pattern Recognition and Artificial Intelligence*, 2001.

[71] U.-V. Marti and H. Bunke. The IAM-database: An english sentence database for off-line handwriting recognition. *International Journal on Document Analysis and Recognition*, 2002.

[72] A. Mishra, K. Alahari, and C. V. Jawahar. Scene text recognition using higher order language priors. In *British Machine Vision Conference*, 2012.

[73] A. Mishra, K. Alahari, and C. V. Jawahar. Top-down and bottom-up cues for scene text recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[74] A. Mishra, K. Alahari, and C. V. Jawahar. Image Retrieval using Textual Cues. In *International Conference on Computer Vision*, 2013.

[75] A. K. Mishra, P. W. Fieguth, and D. A. Clausi. Decoupled active contour (dac) for boundary detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(2):310–324, 2011.

[76] F. Mokhtarian and A. Mackworth. Scale-based description and recognition of planar curves and two-dimensional shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1986.

[77] J. C. Nascimento and J. S. Marques. Adaptive snakes using the em algorithm. *IEEE Transactions on Image Processing*, 14(11):1678–1686, 2005.

[78] L. Neumann and J. Matas. Real-Time Scene Text Localization and Recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[79] L. Neumann and J. Matas. Scene Text Localization and Recognition with Oriented Stroke Detection. In *International Conference on Computer Vision*, 2013.

[80] N. Paragios and R. Deriche. Geodesic active regions and level set methods for supervided texture segmentation. *International Journal of Computer Vision*, 46(3):223–247, 2002.

[81] F. Perronin, J. L. Dugelay, and K. Rose. Iterative decoding of two-dimensional hidden markov models. In *International Conference on Acoustics, Speech and Signal Processing*, volume 3, pages 329–332, 2003.

[82] F. Perronnin and J. Rodríguez-Serrano. Fisher kernels for handwritten word-spotting. In *International Conference on Document Analysis and Recognition*, 2009.

[83] F. Perronnin and J. A. Rodríguez-Serrano. Fisher kernels for handwritten word-spotting. In *International Conference on Document Analysis and Recognition*, 2009.

[84] F. Perronnin, J. Sánchez, and T. Mensink. Improving the Fisher Kernel for large-scale image classification. In *European Conference on Computer Vision*, 2010.

[85] A. Rahimi and B. Recht. Randomm features for large-scale kernel machines. In *Advances in Neural Information Processing Systems*, 2007.

[86] T. Rath and R. Manmatha. Word image matching using dynamic time warping. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2003.

[87] T. Rath and R. Manmatha. Word spotting for historical documents. *International Journal on Document Analysis and Recognition*, 2007.

[88] T. Rath, R. Manmatha, and V. Lavrenko. A search engine for historical manuscript images. In *SIGIR*, 2004.

[89] J. Rodríguez-Serrano and F. Perronnin. Local gradient histogram features for word spotting in unconstrained handwritten documents. In *International Conference on Frontiers in Handwriting Recognition*, 2008.

[90] J. Rodríguez-Serrano and F. Perronnin. A model-based sequence similarity with application to handwritten word-spotting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2012.

[91] J. A. Rodríguez-Serrano and F. Perronnin. Label embedding for text recognition. In *British Machine Vision Conference*, 2013.

[92] K. Rohr, H. S. Stiehl, R. Sprengel, T. M. Buzug, J. Weese, and M. H. Kuhn. Landmark-based elastic registration using approximating thin-plate splines. *IEEE Transactions on Medical Imaging*, 20(6):526–534, 2001.

[93] M. Rohrbach, M. Stark, and B. Schiele. Evaluating knowledge transfer and zero-shot learning in a large-scale setting. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

[94] L. Rothacker, M. Rusiñol, and G. A. Fink. Bag-of-Features HMMs for Segmentation-free Word Spotting in Handwritten Documents. In *International Conference on Document Analysis and Recognition*, 2013.

[95] M. Rusiñol, D. Aldavert, R. Toledo, and J. Lladós. Browsing heterogeneous document collections by a segmentation-free word spotting method. In *International Conference on Document Analysis and Recognition*, 2011.

[96] M. Rusiñol and J. Lladós. The Role of the Users in Handwritten Word Spotting Applications: Query Fusion and Relevance Feedback. In *International Conference on Document Analysis and Recognition*, 2012.

[97] J. Sánchez and F. Perronnin. High-dimensional signautre compression for large-scale image classification. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

[98] J. Sánchez, F. Perronnin, and T. de Campos. Modeling the spatial layout of images beyond spatial pyramids. *PRL*, 2012.

[99] J. Sánchez, F. Perronnin, T. Mensink, and J. Verbeek. Image classification with the Fisher vector: Theory and practice. *International Journal of Computer Vision*, 2013.

[100] K. C. Santosh. Character Recognition based on DTWRadon. In *International Conference on Document Analysis and Recognition*, 2011.

[101] W. J. Scheirer, N. Kumar, P. N. Belhumeur, and T. E. Boult. Multi-attribute spaces: Calibration for attribute fusion and similarity search. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[102] R. Shekhar and C. Jawahar. Word Image Retrieval using Bag of Visual Words. In *Document Analysis Systems*, 2012.

[103] A. Shrivastava, T. Malisiewicz, A. Gupta, and A. A. Efros. Data-driven visual similarity for cross-domain image matching. *ACM Transaction of Graphics (TOG) (Proceedings of ACM SIGGRAPH ASIA)*, 2011.

[104] B. Siddiquie, R. S. Feris, and L. S. Davis. Image Ranking and Retrieval Based on Multi-Attribute Queries. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

[105] P. Sidiropoulos, S. Vrochidis, and I. Kompatsiaris. Content-based binary image retrieval using the adaptative hierarchical density histogram. *Pattern Recognition*, 2010.

[106] L. Torresani, M. Szummer, and A. Fitzgibbon. Efficient object category recognition using classemes. In *European Conference on Computer Vision*, 2010.

[107] A. Vinciarelli and S. Bengio. Offline cursive word recognition using continuous density hidden markov models trained with PCA or ICA features. In *International Conference on Pattern Recognition*, 2002.

[108] A. Vinciarelli, S. Bengio, and H. Bunke. Offline recognition of unconstrained handwritten texts using HMMs and statistical language models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2004.

[109] C. Wah and S. Belongie. Attribute-Based Detection of Unfamiliar Classes with Humans in the Loop. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2013.

[110] K. Wang, B. Babenko, and S. Belongie. End-to-end Scene Text Recognition. In *International Conference on Computer Vision*, 2011.

[111] J. Weston, S. Bengio, and N. Usunier. Large scale image annotation: Learning to rank with joint word-image embedding. In *European Conference on Machine Learning*, 2010.

[112] D. Willems, R. Niels, M. van Gerven, and L. Vuurpijl. Iconic and multi-stroke gesture recognition. *Pattern Recognition*, 2009.

[113] I. Yalniz and R. Manmatha. An efficient framework for searching text in noisy documents. In *Document Analysis Systems*, 2012.

[114] D. Zhang and G. Lu. Review of shape representation and description techniques. *Pattern Recognition*, 2004.

[115] X. Zhang and C. L. Tan. Segmentation-free Keyword Spotting for Handwritten Documents based on Heat Kernel Signature. In *International Conference on Document Analysis and Recognition*, 2013.