Document Image Representation, Classification and Retrieval in Large-Scale Domains

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

Bellaterra, November 18, 2012

Director	**Dr. Ernest Valveny** Dept. Ciències de la Computació & Centre de Visió per Computador Universitat Autònoma de Barcelona
Co-director	**Dr. Florent Perronnin** Textual Visual and Pattern Analysis Group Xerox Research Centre Europe
Thesis committee	**Dr. Hervé Jégou** INRIA Rennes, France **Dr. Andrew D. Bagdanov** University of Florence Florence, Italy **Dr. Dimosthenis Karatzas** Universitat Autònoma de Barcelona Barcelona, Spain

This document was typeset by the author using LaTeX 2$_\varepsilon$.

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

Copyright © 2012 by Albert Gordo. All rights reserved. No part of this publication may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopy, recording, or any information storage and retrieval system, without permission in writing from the author.

ISBN:

Printed by

Acknowledgements

Oh I am a lonely painter
I live in a box of paints
I'm frightened by the devil
And I'm drawn to those ones that ain't afraid

Joni Mitchell - A Case of You

I would like to start by thanking my thesis directors, Ernest Valveny and Florent Perronnin, for the excellent supervision they provided me during the last four years. For being patient when I was lacking focus, and for being focused when I was lacking patience. For letting me drift very far, but never *too* far. For the shiny toolbox full of ideas. Although obvious, I find it essential to remark that this thesis would not have been possible without them.

Through the last four years I have had the pleasure to meet many amazing people who have helped, directly or indirectly, to shape this work. All the friends and colleagues at CVC in Barcelona, but also all the people I met in Grenoble during my stays at XRCE. Although some of those friendships were short-lived – for such is the fate of an intern's life –, their imprint was not. Thank you for all the great discussions and insights, for the tricks of the trade and the lessons in history, but also for the wonderful times that we had far from work, and specially for bearing with me all those times I might not have been the greatest person to be with. I learned a lot from all of you, and I could not be happier about that.

I would like to profusely thank Andrés Marzal and the late Gloria Martínez. To put it succinctly, they changed my life; I would simply not be writing this today if it was not for them. Back in the day, I was terribly disheartened with what, I was being taught, was computer science and computer engineering. And then, Andrés and Gloria appeared, and showed me the beauty and the magic behind the curtains, from greedy algorithms to Gödel's incompleteness. And then it all was made clear. Thank you for putting these endearing stairs where I was only seeing an endless wall.

Finally, I would like to thank my parents and siblings for the endless love and support they have provided me through the last twenty-eight years. For being who they are, and for making me the person I am.

Abstract

Despite the "paperless office" ideal that started in the decade of the seventies, businesses still strive against an increasing amount of paper documentation. Although many businesses are making an effort in transforming some of the internal documentation into a digital form with no intrinsic need for paper, the communication with other businesses and clients in a pure digital form is a much more complex problem due to the lack of adopted standards. Companies receive huge amounts of paper documentation that need to be analyzed and processed, mostly in a manual way. A solution for this task consists in, first, automatically scanning the incoming documents. Then, document images can be analyzed and information can be extracted from the data. Documents can also be automatically dispatched to the appropriate workflows, used to retrieve similar documents in the dataset to transfer information, *etc*.

Due to the nature of this "digital mailroom", we need document representation methods to be *general*, *i.e.*, able to cope with very different types of documents. We need the methods to be *sound*, *i.e.*, able to cope with unexpected types of documents, noise, *etc*. And, we need to methods to be *scalable*, *i.e.*, able to cope with thousands or millions of documents that need to be processed, stored, and consulted. Unfortunately, current techniques of document representation, classification and retrieval are not apt for this digital mailroom framework, since they do not fulfil some or all of these requirements.

Through this thesis we focus on the problem of document representation aimed at classification and retrieval tasks under this digital mailroom framework. Specifically, on the first part of this thesis, we first present a novel document representation based on runlength histograms that achieves state-of-the-art results on public and in-house datasets of different nature and quality on classification and retrieval tasks. This representation is later modified to cope with more complex documents such as multiple-page documents, or documents that contain more sources of information such as extracted OCR text. Then, on the second part of this thesis, we focus on the scalability requirements, particularly for retrieval tasks, where all the documents need to be available in RAM memory for the retrieval to be efficient. We propose a novel binarization method which we dubbed PCAE, as well as two general asymmetric distances between binary embeddings that can significantly improve the retrieval results at a minimal extra computational cost. Finally, we note the importance of supervised learning when performing large-scale retrieval, and study several approaches that can significantly boost the results at no extra cost at query time.

Resumen

A pesar del ideal de "oficina sin papeles" nacida en la década de los setenta, la mayoría de empresas siguen todavía luchando contra una ingente cantidad de documentación en papel. Aunque muchas empresas están haciendo un esfuerzo en la transformación de parte de su documentación interna a un formato digital sin necesidad de pasar por el papel, la comunicación con otras empresas y clientes en un formato puramente digital es un problema mucho más complejo debido a la escasa adopción de estándares. Las empresas reciben una gran cantidad de documentación en papel que necesita ser analizada y procesada, en su mayoría de forma manual. Una solución para esta tarea consiste en, en primer lugar, el escaneo automático de los documentos entrantes. A continuación, las imágenes de los documentos puede ser analizadas y la información puede ser extraida a partir de los datos. Los documentos también pueden ser automáticamente enviados a los flujos de trabajo adecuados, usados para buscar documentos similares en bases de datos para transferir información, *etc*.

Debido a la naturaleza de esta "sala de correo" digital, es necesario que los métodos de representación de documentos sean *generales*, es decir, adecuados para representar correctamente tipos muy diferentes de documentos. Es necesario que los métodos sean *robustos*, es decir, capaces de representar nuevos tipos de documentos, imágenes con ruido, *etc*. Y, por último, es necesario que los métodos sean *escalables*, es decir, capaces de funcionar cuando miles o millones de documentos necesitan ser tratados, almacenados y consultados. Desafortunadamente, las técnicas actuales de representación, clasificación y búsqueda de documentos no son aptos para esta sala de correo digital, ya que no cumplen con algunos o ninguno de estos requisitos.

En esta tesis nos centramos en el problema de la representación de documentos enfocada a la clasificación y búsqueda en el marco de la sala de correo digital. En particular, en la primera parte de esta tesis primero presentamos un descriptor de documentos basado en un histograma de "runlengths" a multiples escalas. Este descriptor supera en resultados a otros métodos del estado-del-arte en bases de datos públicas y propias de diferente naturaleza y condición en tareas de clasificación y búsqueda de documentos. Más tarde modificamos esta representación para hacer frente a documentos más complejos, tales como documentos de varias páginas o documentos que contienen más fuentes de información como texto extraído por OCR. En la segunda parte de esta tesis nos centramos en el requisito de escalabilidad, sobre todo para las tareas de búsqueda, en el que todos los documentos deben estar disponibles en la memoria RAM para que la búsqueda pueda ser eficiente. Proponemos un nuevo método de binarización que llamamos PCAE, así como dos distancias asimétricas generales para descriptores binarios que pueden mejorar significativamente los resultados de la búsqueda con un mínimo coste computacional adicional. Por último, señalamos la importancia del aprendizaje supervisado cuando se realizan búsquedas en grandes bases de datos y estudiamos varios enfoques que pueden aumentar significativamente la precisión de los resultados sin coste adicional en tiempo de consulta.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Will the office change all that much? Listen to George E. Pake, who heads Xerox Corp.'s Palo Alto (Calif.) Research Center, a new think tank already having a significant impact on the copier giant's strategies for going after the office systems market: "There is absolutely no question that there will be a revolution in the office over the next 20 years. What we are doing will change the office like the jet plane revolutionized travel and the way that TV has altered family life."
Pake says that in 1995 his office will be completely different; there will be TV-display terminal with keyboard sitting on his desk. "I'll be able to call up documents from my files on the screen, or by pressing a button," he says. "I can get my mail or any messages. I don't know how much hard copy [printed paper] I'll want in this world."

The Office of the Future – Bloomberg BusinessWeek, June 30, 1,975.

In the last 20 years, working from a screen has become the norm in offices; however, companies still receive and produce large volumes of paper documents every day. The trend is the same around the world: A steep increase in paper received by companies. In the UK, it is estimated that 73% of all documents received by companies are paper documents (62% from mail and 11% from faxes). Electronic documents still represent less than a quarter of the bulk of documents (16% as emails, 8% as web forms). On average, companies today receive 3 million items per year and the cost of manual processing of incoming mail is estimated at 0.15 pounds to 0.25 pounds per item.

Implementing a Digital Mailroom [5] (White Paper) – Datafinity, July 2,009

George Pake was indeed right about the revolution that would come in the following decades. The flourishment of computers and word processing during those years was slow but steady, and by 1,995 we certainly had a "TV-display terminal with keyboard" on the desk, and were calling up documents from files on the screen and getting mail or messages. However, Pake was wrong about something: the paperless office is, still today, a myth, and offices still dive in endless stacks of physical documents.

The advent of word processing freed us from creating documents directly on paper; now they can be created in a digital form. However, the distribution of those documents is, almost 40 years later, still done mostly in a hard format. What has changed is the time

and place at which a piece of information is converted into a piece of paper [5]. Even if some effort has been put inside companies to ease the communication of documentation in digital form, paper is still the predominant format. This digital communication is even harder when dealing with other companies, clients, *etc*. Unless a standard on document information representation was defined *and* adopted offices will remain well stocked of paper.

In this scenario, where companies still receive incoming documents in paper format, one step towards the ideal paperless office would be to digitize and understand the contents of the incoming documents with minimal human intervention. Then, the relevant information could be extracted from the data and decisions taken accordingly. This naturally leads to the digital mailroom paradigm.

1.1 The Digital Mailroom Paradigm

In the digital mailroom paradigm, incoming documents are first digitized (in case they were not digital in the first place), analyzed, indexed, sent to the correct workflow or workflows, and archived, all in a centralized way. By digitizing the incoming mail process and indexing on the fly, companies gain more control of their mail process internally, as well as combining the already electronic mail formats (email, fax) into the same workflows. An example of the digital mailroom can be seen in Figure 1.1. Let us review the main stages based on [5]:

1. **Document Acquisition:** When incoming documents arrive to the mailroom, they may do so through different channels. Documents that arrive through digital channels such as email may not need any particular processing. Similarly, on most fax machines, documents arriving through fax can usually be saved in a digital image format instead of being printed. However, documents arriving in paper will need to be manually scanned, or, at least, fed to a scanning device. After this acquisition stage, all documents can be treated with the same procedures, independently of what channel was used to send them. Note that at this point we have not yet performed any kind of data analysis, and we are just interested in obtaining a digital copy of the original document, independently of the input channel.
2. **Document Understanding:** Once the documents have been digitized and are available in a digital form, we would be interested in automatically understanding their contents. Understanding a document would ease a significant amount of tasks, from delivering the document to the right workflow, to extract the relevant information, annotate the documents (for example, with their relevance and priority), *etc*.
3. **Workflow:** Once the document has been analyzed, that information can be used to send the document to the correct workflow, where information will be extracted from the data, either manually or by a computer, and decisions taken accordingly. Preserving the extracted information in a database could be extremely useful and make further consultation of the physical document unnecessary.
4. **Archiving:** Once the information has been extracted from the documents, we may be interested in preserving a digital image of the document instead of the actual, physical paper. This digital image can be crossreferenced with the information on the databases, and the image could be enriched embedding the relevant information on it. This would, theoretically, remove the need to keep a physical paper copy of the document. As noted in [5], however, different legislations may force the company to keep the physical copies of the documents.

Of these four steps, document understanding is the core of the digital mailroom, as well as being, by far, the most challenging step. The huge variability in the documents domain

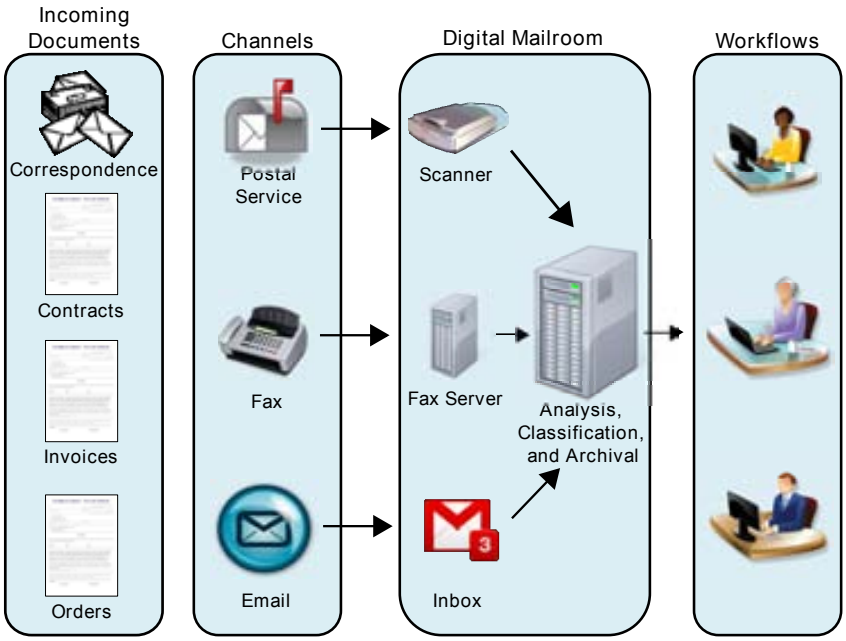


Figure 1.1: The digital mailroom

makes this an extremely difficult task. As a consequence of this, research has been focused on solving small sub-problems of document understanding. For example, obtaining the layout of a page for Manhattan and non-Manhattan layouts [6], classifying forms [108] – since, once classified, we can use templates to extract the relevant information –, distinguishing between regions of text, images, or tables [67], parsing tables [32], *etc.* However, most of these methods have restringent constrains that are usually not met in the wild. For an excellent survey on document analysis and understanding, please refer to [87]. Part of the article is dedicated to discussing which problems of document analysis are solved and which are not. An unsettling yet encouraging conclusion is that *most everything that is not trivial remains to be effectively solved.* In the 10 years since the publication of the article, we have made huge advances in many of the proposed problems. However, the conclusion remains the same: *most everything that is not trivial has yet to be effectively solved.* It is therefore unlikely that a system able to understand a wide range of document images will appear in the foreseeable future, and so it may be more rewarding to focus on less ambitious goals that will still improve the management of documents.

One such goal could be the classification and retrieval of documents. Classifying the document into a set of predefined categories is usually a significantly easier task than understanding the contents. Classified documents could be sent to the correct workflows without completely understanding the documents, in the same way that a human does not usually need *all* the document information to know where it should be delivered. In most cases, there are a series of cues based on visual aspect, layout structure, textual information, *etc.*, that are used to make the classification decision, and computers could classify documents

based on those cues without needing to understand the contents.

Correctly classifying the documents also leads to a better document understanding. If we know which particular form we are dealing with, we can more easily extract the information. Extracting tables on a document without any extra information may be extremely difficult, but if we expect the document to have a table, and we know what types of tables those documents usually have, then the task becomes significantly easier. The classifier may also decide that it is not clear what kind of document it is dealing with, and request human feedback, asking precise questions that will help it to correctly categorize it. Although we are not aware of the application of this "humans in the loop" framework for fine-grained document recognition, this approach has been recently applied in other computer vision tasks [18, 85].

In some cases, we can also be interested in retrieving similar documents instead of (or to help) classifying. For example, retrieving the closest documents in the archives may help to transfer metadata information and make a more informed decision, either by a machine or supervised by a person. It may also lead to realizing that it is a new kind of document that needs special attention. As opposed to document understanding, document classification and retrieval are tasks that can be tackled with reasonable success with our current tools, while at the same time there is significant room for improvement.

1.2 Objectives of this Thesis

In the previous section we noted how the classification and retrieval of documents is particularly important in this digital mailroom scenario, both for its direct applications – such as selecting the right workflow – as well as the indirect ones – such as finding a right template to extract the information from the document. As a consequence of this, there has been a significant amount of works dealing with the problem of representing, classifying and retrieving documents. Some of the key points that are desirable for those methods are:

1. **Generality.** Ideally, we should be able to apply the same pipeline to classify very different types of documents, from handwritten letters to forms or invoices.
2. **Soundness.** Methods should be resilient to small artifacts such as minor skew variations, noise, *etc*. Even if a drop in accuracy can be expected, a document should not be immediately rejected only because it is noisy.
3. **Scalability.** Describing and classifying or retrieving documents should be a very fast operation. This is very important since large companies deal with millions of documents yearly. As a consequence, they need to perform queries on datasets that may contain up to hundreds of millions of documents, and methods that are not fast enough will not be suitable. Additionally, for some tasks such as retrieval to be efficient, documents need to be kept in RAM memory at the same time. Unless the memory footprint of the documents is really small, this will not be feasible. Therefore, it is important to research methods that allow to aggressively compress the documents with minimal loss.

Unfortunately, as we will see through chapter 2, current methods in the literature do not fulfill these desiderata. Furthermore, we find important to note that the available literature usually understands document representation in a very restringent way, *i.e.*, single-page documents using some particular set of features. However, documents in general, and particularly in the digital mailroom, have some traits that cannot be accurately captured in such a way. For example, some documents may contain not one but several pages, and we need to find a discriminative representation for such type of documents. In some other cases, we may

need to combine different sources of information – such as visual and textual – in the same representation to improve its quality. Moreover, some of those sources may not always be available for all the documents, since they may be difficult / slow / expensive to extract for every document.

This dissertation aims to address these issues, and examines the problem of administrative document representation, classification and retrieval. In particular, the goal of this thesis is to research better document representation techniques that can cope with the aforementioned issues: generality, soundness, and scalability. We also aim to represent more complex documents, such as multiple-page documents, and to explore ways to enrich the representation of documents when very discriminative information (such as textual contents, spotting of particular words, checkboxes, *etc.*) may be available during a training stage, but will in general not be available at testing time due to the associated cost of obtaining it.

For some of our goals – particularly, the scalability of the methods –, we will need to go beyond "document representation", and research some more general techniques such as binary embeddings and asymmetric distances, that can, nonetheless, be applied to our document representation problems in a very natural way.

Through the next sections we will first describe the organization of this thesis and then we will summarize our main contributions.

1.3 Organization of this Thesis

The thesis is divided in two parts. The first part addresses the problem of document representation. Indeed, having a good representation of the documents is of paramount importance to perform other tasks such as classification, clustering, retrieval, *etc.* The second part addresses the particular problem of retrieval in large-scale scenarios. Note that both parts, although different in nature, are very interrelated. When designing a document representation, many decisions will be influenced by the fact that the descriptors will be used to represent documents in large datasets. For example, we expect the descriptors to be fast to compute – so computing the descriptor does not delay the pipeline –, reasonably small – so they can fit in memory –, *etc.*

In particular, the chapters of Part 1 deal with the following topics:

- In chapter 2, we deal with the problem of finding a generic, fast to compute, discriminative, single page representation. We first review the existing document representation literature, since many document representations have been proposed. Most of these representations are not state-of-the-art anymore, or suffer from some drawbacks that render them inappropriate for our goals. Then, we propose a runlength representation of the documents. This representation is fast to compute, compact, and does not require any layout analysis. Evaluation on two public and one in-house dataset shows how this representation obtains results beyond the current state-of-the-art at a very reasonable cost.
- In chapter 3, we explore ways to represent documents containing multiple unordered pages in a compact way. Although this is a common problem in the digital mailroom, extremely few publications exist on the topic. We propose to use a bag of pages, akin to the bags of words or visual words used in text and image representation. This bag of pages is later improved using the Fisher Vector framework. The end result is a compact signature that represents the multipage document and that clearly outperforms our baselines on classification and retrieval tasks. Furthermore, we show that, if labeled documents are available for training purposes, we can exploit this information to learn even more discriminative signatures.

- In chapter 4, we deal with the problem of mixing multiple sources of information or views, for example, visual and textual information. These are common choices since they contain very discriminative information, but other sources could be used, such as a layout graph representation of the document, tags, *etc*. Combining views is a standard procedure in computer vision tasks and several techniques exist, from early fusion of the features to late fusion of the scores or Multiple Kernel Learning. However, these methods assume that all the views will be available during testing time, which may not always be possible. Indeed, some views may be too *expensive* to be obtained at test time for every document. An example of this may be the textual contents of a page obtained with an OCR application, which may be too slow for our pipeline or be economically expensive because of fees. Nevertheless, we may be able to collect, *offline*, the expensive views of some documents.

  This chapter addresses this particular scenario, where some expensive views may be available for some samples for training purposes, but will not, in general, be available at testing time. We propose to use the expensive views, available only at training time, to improve the representation of the *cheap* views ,*i.e.*, the views that are available both at training and testing time. In particular, we propose to use Canonical Correlation Analysis to learn, offline, a common subspace between the different views. At testing time, the available cheap features can be projected into the common subspace and classified. Since we used the costly views to learn the subspace, the cheap views are coached into a subspace with more information. We observed significant improvement in classification and retrieval tasks, particularly when the number of labeled samples was small or non-existent.

The chapters of Part 2 deal with the problem of large-scale retrieval. This is a very general task, and can be applied to other domains such as natural images. In fact, for convenience reasons, most of the experimental validation of the methods of Part 2 will be performed on datasets based on natural images. We underline, however, that the methods are very generic and can be applied directly on documents; we do in fact devote a section of chapter 5 to study the effect of the proposed methods on documents, replicating some of the experiments of chapters 2 to 4 under large-scale conditions. In particular, Part 2 deals with the following problems:

- In chapter 5 we show how to compress the documents to make them suitable for large scale scenarios, particularly for retrieval in large datasets. Although the descriptors we have proposed are quite compact, efficient retrieval requires to store all the dataset in RAM memory. Otherwise, accessing the hard disks produces a huge performance drop. A common approach to this problem is binarization, where objects are represented with short binary codes and, in most cases, compared using the Hamming distance. Many binarization techniques have been proposed in the past few years. Examples of these methods are Locality Sensitive Hashing (LSH) , Spectral Hashing (SH), Kernel LSH (KLSH), Locality Sensitive Binary Codes (LSBC), or Semi-Supervised Hashing (SSH). Each of these binarization methods has certain properties and theoretical guarantees, making them particularly useful in certain scenarios. In this chapter, we first review and analyze some of the most common binarization techniques. Then, based on this analysis, we show how to use asymmetric distances to match binarized descriptors with a non-binarized query. Indeed, binarizing the query does not offer any significant space improvement but considerably degrades the quality of the descriptor. Although the use of asymmetric distances in this retrieval context is not new, they have always been paired with a particular embedding technique. Here, we show two different asymmetric distances that can be used with many popular embedding methods such as LSH, LSBC, or SH, significantly improving their accuracy.

- In chapter 6 we deal with the problem of leveraging category-level annotated data to improve instance-level retrieval. When performing instance-level retrieval, it is not rare to obtain some results that are semantically inconsistent, *i.e.*, they are not only not from the same instance, they are not even from the same type of category. Through this chapter we explore how to use large amounts of category-level annotated data to learn a more semantic representation of the images. Our intuition is that, when retrieving images that are more semantically consistent, we will also retrieve more images of the same instance among the top results. We tested four different approaches and observed that instance-level retrieval results can indeed be improved when a large amount of class-level labeled information is available.

1.4 Contributions of this Thesis

Finally, we summarize the main contributions of this thesis:

1. We present a novel descriptor for single-page document images based on runlength histograms. This representation is fast to compute, compact, and does not require any layout analysis. Evaluation on two public and one in-house dataset shows how this representation obtains results beyond the current state-of-the-art at a very reasonable cost.
2. We propose a bag of pages approach to represent documents containing multiple unordered pages. We also propose a supervised method to create more discriminative signatures when labeled data is available for training purposes.
3. We show how Canonical Correlation Analysis can be used to enrich the document signatures with supplementary information (such as textual information) to learn more discriminative signatures at train time, even if the supplementary information will not be available at test time.
4. We propose two generic asymmetric distances between a non-binary query and a binary dataset item which significantly improve several binary encoding methods and yield state-of-the-art results at a very reasonable cost. We also propose a very simple binarization method which we dubbed PCAE. When paired with the asymmetric distances, the performance of PCAE is on par with other more complex binarization methods.
5. We explore the use of *category-level* labeled data to improve the task of *instance-level* retrieval. In particular, we overview some methods that have been used for similar tasks such as metric learning, attributes representation, or Canonical Correlation Analysis, and propose a method that has not been used in this context and that significantly outperforms the baseline results.

Part I

Document Representation

Chapter 2

Single-Page Document Representation[1]

Document (noun)
2 a: a writing conveying information.

Merriam-Webster Online

Document: *noun*.
A piece of written, printed, or electronic matter that provides information or evidence or that serves as an official record.

Oxford Dictionaries Online

2.1 Introduction

The Oxford dictionary gives one acceptation of the noun document as *a piece of written, printed, or electronic matter that provides information or evidence or that serves as an official record*. This is a quite broad definition: letters or invoices are documents, but so can be, for example, pictures or videos. This broadness is not surprising since the etymology of document is Latin *documentum* (lesson, proof), derived from *docere*, to teach. Merriam-Webster gives a more constrained definition: *a writing conveying information*. This is more in line with the meaning usually accepted by the document analysis community.

However, even inside the community, the definition of document is not perfectly standardized. Nagy proposed in [87] a possible taxonomy of documents, which we show in Figure 2.1. There is a separation between "mostly graphic" documents, such as drawings, diagrams, sheet music, and maps, and "mostly text", such as newspapers, magazines, or plain text. Note however that this separation can be unclear sometimes, and the limits between a mostly graphic and a mostly text document can be fuzzy.

To be able to represent these documents, some features need to be extracted from the documents. These features may depend on the particular task we want to perform. For

[1] Parts of this chapter submitted to *Pattern Recognition*.

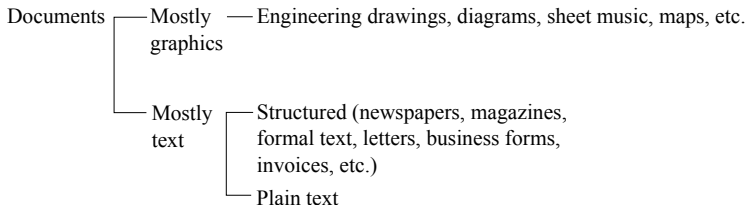


Figure 2.1: Document taxonomy as defined by Nagy in [87]

example, the features that will help us categorize a document or retrieve similar documents from a dataset can be completely different to the features we will need to extract if we want to perform document understanding, patent retrieval, table parsing, *etc*. The types of features we may want to extract may also be influenced by the type of document, *i.e*., features for mostly graphic documents are traditionally different from features for mostly text documents. Note that this is against the *generality* principle that we discussed in the introduction, and so it would be important to find representations that can represent both branches of documents. Achieving this may be extremely difficult when aiming at tasks such as document understanding, where understanding an ID card is a completely different task from understanding a table in a text document. However, when aiming at other tasks such as classification and retrieval, which are the core of the digital mailroom, it may be possible to find general features apt for all types of documents. Indeed, we may not be interested in understanding the contents of the ID card or the table, only in distinguishing them or being able to find similar documents in a dataset.

Based on Nagy's taxonomy, Chen and Blostein review in [23] the most relevant types of document features when aiming at classification and retrieval tasks:

- **Visual features**. These features describe the overall aspect of the document or regions of it, and are intended to capture document styles that can be differentiated "at a glance". To do so, they generally use techniques based on texture analysis, morphological operations, frequency of patches, *etc*. In general, visual features do not encode structural information of the page in an explicit way. However, capturing some structure is extremely important to correctly represent many types of documents. A common workaround consists in using techniques such as spatial pyramids [78] to compute the descriptors. In that case visual features also manage to capture some basic structural information. Because visual features can take into account the overall aspect of the page, they can be useful both for mostly-graphics and mostly-text documents. Visual features are *usually* fast to compute, and, as we will see in the following section, methods using visual features tend to produce feature vectors of fixed size. This makes them a typical choice for representing documents.
- **Structural features**. The visual features we just described capture how the document looks like "at a glance". Using a spatial pyramid we were able to capture some basic structural information of the document. However, they do not lead to easily capture the *relations* between the structures. Structural features try to address precisely this. Structural features are related to the disposition and relationships between structures on a page – and are, therefore, restricted to structured documents. These structures or blocks could contain relatively high level information (this box is a *title*, here is a *footnote*, this is a *stamp*...) usually called *logical* layout, medium level information (this block looks like *text*, this other does not), or just low level information (this block has this size and is at this position), referred to as *physical* layout.

Oftentimes, we find techniques were visual and structural features are used together. For example, we can calculate a descriptor based on visual features *only* over the blocks obtained after a physical layout extraction. Or, we can obtain a logical layout which includes, among others, visual characteristics of the blocks. There is little consensus on whether such methods should be labeled as "visual" or "structural". One trend (which we will adopt) is to label as "visual" the methods that represent the image with visual features even if they use some structural information to decide where to extract the features, and to label as "structural" those who use a graph-like representation of the document, even if this graph is enriched with visual information.

Representations based on high level, enriched structural features contain more information than representations based solely on visual features and should, at least intuitively, perform better on structured documents. However they have some problems. First, obtaining accurate layout information is extremely complex, error prone, and time consuming. Different types of documents (*e.g.* forms and invoices) have very different types of layouts that can not usually be obtained with the same algorithms. Structural features are very sensitive to noise, and noisy documents – such as those coming through a fax – may produce very inaccurate layouts. Furthermore, handling, learning, and comparing the graph-like structures requires, in general, more resources than when dealing directly with fixed-length feature vectors, both in terms of memory and CPU. In general, even if structural features are powerful, their use should be limited to very controled domains, and their use in the digital mailroom is not recommended because of the potentially bad quality of the input documents (*e.g.*, fax documents) as well as the memory and CPU requirements.

- **Textual features**. Sometimes, visual and structural features are not enough to correctly represent documents. This is particularly true in the case of the digital mailroom, where one of the goals is to send documents to the correct workflow. Sometimes, the same type of document (*e.g.*, the very same form type) may be sent to different workflows according to its textual contents.

  Textual features are usually based on the frequency of representative keywords on the documents using frameworks such as the bag of words. These keywords can be obtained after applying an Optical Character Recognition (OCR) to the document, but sometimes this is not a feasible option. For example, the results of an OCR over degraded or handwritten documents will most likely not have enough quality for an accurate search. In situations like these, word spotting techniques are usually more useful. Word spotting does not require a transcription of the text image, and can identify keywords just based on their graphical features.

As noted in the introduction, our main goal is to produce document representations that are general, sound, and scalable. It is therefore not surprising that these constrains will affect our choices in the types of features that we can use. For example, graph representations are usually considered more flexible than feature vectors, and could, at least theoretically, obtain better results. However, these representations are far from general. For example, they can only represent structured documents, which is only a subset of the documents we may find in the digital mailroom. Moreover, the graphs used to represent forms are usually different in nature to the graphs used to represent other documents such as invoices. Furthermore, graph representations are usually slow to compute and compare, and therefore are not scalable. Similarly, text descriptors have a very high discriminative power for many tasks, but are usually expensive to compute, both in computational and economic resources. Furthermore, documents may not contain text at all, or text may be handwritten, making the problem significantly more complex. Through the next section we will review some popular

document representation methods based on these features, and discuss how they cope with our requirements.

2.2 Prior Work on Document Representation

Let us review the most popular methods for document representation in the literature according to the main types of feature they use: visual, structural, and textual.

2.2.1 Visual features

In [28], Cullen *et al.* propose a feature vector of 80 dimensions based on texture analysis of the image. This feature vector concatenates several different features: densities at "interest points", a histogram of connected components size, a vertical projection histogram, and the density of connected components computed in each cell of a 5×4 grid over the document. Finally, the Euclidean distance is used to compare the feature vectors. Experiments are carried on in one small in-house dataset. One important drawback of the method is that most of the information it encodes is global, and only 20 of the 80 dimensions are devoted to encode some structural information. This would be insufficient to encode documents for fine-grained classification and retrieval tasks. Moreover, extracting the connected components, although not terribly slow, is not a fast operation. Although this would not be an issue in small-scale scenarios, it can be one in large-scale scenarios.

In [54], Heroux *et al.* propose a multi-scale density decomposition of the page, which is used to produce a fixed-length descriptor that captures some important structural information thanks to the spatial pyramid. Feature vectors can be constructed efficiently using integral images and compared with fast operations such as the dot-product. Unfortunately, density values are very sensitive to noise, and so these descriptors may be unreliable when encoding noisy documents.

In [8], Bagdanov and Worring present a representation based on describing the density changes after multiple morphology operations. These representations are later compared using the Euclidean distance. Although this approach is more resilient to noise, computing several morphological operations on the whole document is computationally expensive.

In [107], Sarkar proposes a method where images are described as a list of salient Viola-Jones-based features. The list includes the feature type, position and size of every feature that has fired in the document. Because of that, different images will be represented with lists of different lengths. Then, a Latent Conditional Independence model is used to classify the documents, obtaining state-of-the-art results in one public dataset. One drawback of the method is that the image descriptors do not have a fixed size, and so they are more difficult to store and compress than feature vectors. This will make its use problematic for large-scale tasks. Furthermore, as presented, this representation is designed to be used only in classification tasks. It is not clear how this representation could be used for other tasks such as retrieval or clustering.

In [113], a feature vector is constructed based on image features such as percentages of text and non-text, column structures, density of content area, or connected components features. Then the feature vectors are classified using decision trees and self-organizing maps. This approach mixes some visual and structural features, and requires to perform some slow and unreliable tasks: layout analysis, text and non-text separation, *etc.*

2.2.2 Structural features:

Structural features may contain more information than visual features alone, but unfortunately they usually rely on the results of a physical layout analysis, which is complex and error-prone. In general, given our requirements, structural features are discouraged.

In many cases, layout graphs are extracted from the images, and the classification and retrieval is based on finding suitable distances between graphs. Such distances can be, for example, the Minimum Weight Edge Cover (used by Keysers *et al.* in [66]) or the Earth's Mover Distance (used by Rubner *et al.* in [104] and Beusekom *et al.* in [124]). In [80], Liang *et al.* propose to represent the pages with a fully connected attributed relational graph and define a distance between such graphs. All these distances can be used directly for classification and retrieval tasks using a k-NN framework. They can also be used with large-margin classifiers such as an SVM in a direct way, by first representing the graphs as a function of some d representative samples, obtaining a d-dimensional representation, and then learning an SVM in that space. In [48], the SVM scores of the embedded samples are used to construct an attributes-based (*e.g.* [76, 126]) representation of the documents, which outperformed the direct distance embedding in semantic retrieval tasks. In [7], Bagdanov and Worring propose to use training documents with their associated graphs to learn representative First Order Gaussian Graphs for each possible class in the dataset. Then, given a new document graph, the probability of it being generated by each of the representative graphs can be computed and used for classification tasks. Again, it is not clear how this representation could be used for other tasks such as retrieval. In [51], the layout of the page is flattened into a sequence of blocks and compared with an approximate cyclic Dynamic Time Warping. The result is a rotation invariant distance measure which is evaluated in an in-house dataset for retrieval tasks. In [108], Saund propose an alternative approach for form description where it is not necessary to extract the whole layout of the page. Instead, only the junctures need to be extracted. Based on them, a graph lattice is used to represent the images. Although it does not need to perform a complete layout extraction, obtaining the form junctures is not an easy task either.

In some cases the documents layout can be represented with trees, which are usually less computationally demanding than graphs. A very typical choice is the X-Y tree [88], where documents are represented in a hierarchical way, grouping blocks in alternative horizontal and vertical projections. One such example is the work of [19], where the X-Y tree is modified to segment and represent journal pages. The modified tree makes cuts not only over white spaces but also over vertical and horizontal lines, which is useful for some types of documents such as forms. In [84], Marinai *et al.* propose to construct an X-Y tree on the layout of the page and to use a tree edit distance to classify the documents. Chen *et al.* follow a similar approach in [24]. Tree grammars and query expansion are then used to compensate for possible segmentation errors at an extra computational cost. In a similar way, Perea and Lopez [93] also construct a tree-based representation of the form layout. Then the syntactic representations of the documents are used to infer a tree automaton for each one of the classes involved in the task.

Note that these distances usually have a quadratic cost with respect to the number of nodes of the graphs / trees. Therefore, even if perfect layout graphs were extracted, the distance computation would still be unfeasible in large-scale cases.

In [79], a coarse physical layout is extracted by means of runlength smoothing [134], and then line and word image patches are extracted. The whole document is represented as a sequence of word patches described with their length, and documents are compared using sequence matching by means of dynamic programming. This method is devised to find documents that share some sequences of text, but seems inappropriate for general document representation.

Finally, some works perform the discrimination of layouts based on sets of rules. These rules can be hardcoded – as in [44], where rules are used to classify blocks in a document – or automatically learned from a training set – as Esposito *et al.* do in [40] or [39].

2.2.3 Textual features:

Classification using textual features is a task closely related to the text categorization problem in information retrieval [111]. Since its relation with computer vision is limited, we will only give a very brief overview. After extracting and recognizing the characters and words of an image, the indexing of documents is mostly based on weighted features (terms) that appear in a minimum number of documents (see, for example, the *tf-idf* schema [106]). These weights are usually computed by either statistical or probabilistic techniques [105, 136]. These terms can be put in an large inverted file for efficiency reasons (see *e.g.* [133]). By doing so, retrieval can be performed by searching only a small subset of the whole dataset. Another option consists in explicitly constructing a bag-of-words feature vector, that can be used for many tasks such as clustering, classification, *etc.* One of the main problems that arise here is that the OCR results will not be perfect, particularly on noisy images. In [57], a probabilistic method is proposed to overcome this problem, while in [117] the OCR is completely skipped and the classification relies on word spotting.

2.3 The Runlength Histogram

Through the previous section we reviewed the most typical features used to describe document images when aiming at classification and retrieval tasks, as well as many methods in the literature that use these features for such tasks. Unfortunately, we noticed that most of these methods would not be useful for our digital mailroom, since they are not general, sound, or scalable. We already mentioned that structural features can only work for structured documents, which is only a subset of the documents of the digital mailroom. Similarly, although text features can be extremely powerful, not all documents contain textual information, and, even if it exists, it may be difficult or expensive to extract. Therefore, we should focus on pure visual features that do not require a layout analysis to represent the documents.

A typical choice consists in extracting statistical information from the connected components of the image. Unfortunately, extracting the connected components is not a fast operation, and can take up to a few seconds if the images are large. Furthermore, connected components analysis is particularly sensitive to noisy documents, since it can significantly deform the components, merge large sections of the document, *etc.* Encoding the density of regions of the document is another typical option, but, once again, it can be quite sensitive to noise. Furthermore, we only have one measure per region, which may not be good enough to encode complex documents. A third option, which we will explore through this chapter, is the use of runlength histograms.

In a nutshell, a run is a sequence of pixels of the same value going in the same direction, and the runlength is the length of those runs. As an example, Figure 2.2 shows a region of an image where two runs have been highlighted: a vertical black run of length 7 and an horizontal white run of length 16. Given a region, we can compute the length of the runs of black and white pixels and compute a histogram of the lengths. These runs are encoding some (very) local structure of the region, and so offer more information than just computing the density. We can compute the runlengths in several directions using the white and black pixels independently to obtain more information. The use of runlengths is convenient since they are

fast to compute, and, if their lengths are encoded in a logarithmic way, are quite resilient to small variations, noise, *etc*. Because of this, runlengths are not new in the document analysis community. In [67], runlength features are used to help classifying document zones as text or non text. In [118], runlength histograms are used to detect the frames of double-paged document images. In [119], textures are described by means of runlengths. However, we are not aware of any use of runlength histograms for a whole page representation.

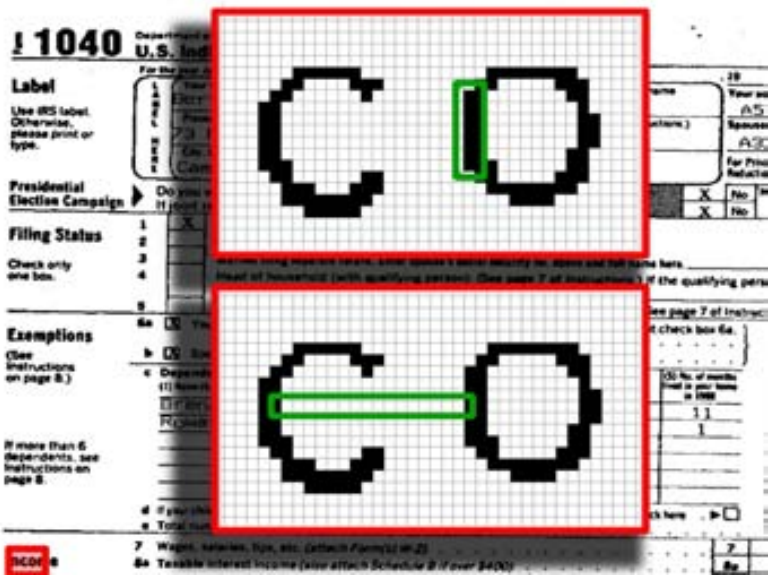


Figure 2.2: Examples of pixel runs. A vertical black run of length 7 (top) and an horizontal white run of length 16 (bottom). Detail from a small region on the bottom-left corner.

These runlength histograms can be computed in different regions of the image, maybe in different scales, to capture some important global structure of the document. Finally, we obtain a single feature vector that represents a whole document independently of its type, which is fast to compute, encodes more information than a density decomposition, and, as we will see in the experimental section, can obtain state-of-the-art results on two public datasets.

Through the next section we describe how to construct the runlength histogram.

2.3.1 Computing the Runlength Histogram

The document encoding is based on the following steps:

Step 1. Normalization (optional): Several steps can be performed to normalize images, such as centering, cropping, re-scaling, skew correction, *etc*. Throughout our experiments, we have only performed a re-scaling of the images, since it has a very noticeable effect in the final results.

Step 2. Pixel quantization: The runlength encoding requires a small number of levels to be efficient. In our case, we deal with binary images, *i.e.*, with two levels.

Step 3. Region extraction: The image can be partitioned into different sub-regions at different scales using spatial pyramids [78]. This is a standard technique to add some

basic structural information to the representation. These regions will later be described independently and finally concatenated. For example, we can see in Fig. 2.3 the splits corresponding to 1×1 (whole image), 2×2, and 4×4 partitions, producing a total of 21 regions.

Note that computing the descriptors at different scales (*e.g.* 1×1 and 2×2) can yield important benefits over computing it only over the small regions (*e.g.*, 2×2): if we apply a non-linear transformation to the histograms – such as the square root that we apply, *cf.* step 6 – then the 1×1 region histogram can no longer be expressed as a linear combination of the 2×2 region histograms and therefore brings some extra information when using a linear classifier.

Step 4. Runlength encoding of the regions: A *run* is a sequence of pixels of the same value (*cf.* Fig 2.2). The length of the run is the number of pixels such a sequence contains, and the runlength histogram is a histogram of the lengths of the runs. Following [67] we propose to quantize the length of the runs in a logarithmic scale as follows:

$$[1], [2], [3-4], [5-8], [9-16], \ldots, [k-]. \tag{2.1}$$

For example, the quantization $[1], [2], [3-4], [5-8], [9-16], [17-32], [33-64], [65-128], [129-]$ contains 9 quantization intervals. The number of intervals used should be adjusted depending on the size of the resized image (see step 1) and the size of the regions (see step 3), although, as we will see in the experimental evaluation of section 2.4.1, their influence is limited. When using 9 intervals, and if dealing with black and white images, this setup yields mini-histograms of length $2 \times 9 = 18$. We compute mini-histograms in horizontal, vertical, diagonal and anti-diagonal directions and concatenate them. Assuming again binary images, this yields a region descriptor of length $18 \times 4 = 72$. Note that this logarithmic-scale quantization of the runlengths makes the representation much less sensitive to noise, overcoming one of the main problems of density representations.

Step 5. Global image representation: To represent the document image, we simply concatenate the runlength histograms of all the regions. Following the previous example with 9 quantization intervals and two levels, that would yield an histogram of $72 \times 21 = 1,512$ dimensions.

Step 6. Normalization: The histogram can later be normalized. Several approaches can be considered:

- Normalize each mini-histogram independently.
- Normalize each region independently.
- Normalize the final histogram as a whole.

Experimental results show little difference between these normalization approaches. In our case, we will perform an L1 normalization over the whole histograms and then square root each of its elements. As noted in [98], the dot-product on L1-normalized, square-rooted vectors corresponds to an explicit embedding of the Bhattacharyya similarity, and so it is particularly suited for discrete probabilistic distributions such as our L1-normalized vectors. One standard explanation (*e.g.* [61, 25]) is that the square root helps to reduce the influence of large values in the histogram similarly to other kernels such as χ^2, which is a desirable property. During preliminary experiments we confirmed this square root normalization to improve the accuracy of the system. Also, on vectors with only positive values, L1 normalization plus square rooting is strictly equivalent to square rooting plus L2 normalization. And, on L2-normalized vectors, Euclidean distance is proportional to the dot-product, which means that the Euclidean distance is also an appropriate distance measure between our runlength descriptors. This is convenient if we want to perform other tasks with our descriptors

Figure 2.3: Original image and the 1×1, 2×2 and 4×4 partitions.

such as clustering. Finally, although other measures typically used on histograms such as χ^2 or histogram intersection could be used, the Euclidean distance seems to be particularly appropriate since it is approximately preserved if we perform a PCA dimensionality reduction.

Step 7. Dimensionality reduction: Note that the quantization of the lengths of the runs we proposed is likely to produce lots of zeros in the final histograms, particularly in the sections corresponding to the smaller sub-regions. In fact, in our experiments, we noticed that approximately 35% of our histogram values are equal to zero. This, along with the fact that using multi-scale histograms may cause correlations between dimensions, suggests that the use of PCA could be beneficial and lead to better results, or, at least, be applied without significant loss.

2.4 Experiments

To test the validity of the runlength descriptor, we will perform experiments in one in-house dataset as well as in two public datasets. First, we will use the in-house dataset to analyse the effect of the parameters of the descriptor: the size of the resized image, the number of regions, and the number of quantization intervals. Then, we will use the best configuration of the in-house dataset to represent the images of two public datasets and compare with the best published results.

2.4.1 Analysis of the Parameters

Through this section, we will study the effect of the parameters in the descriptor. To do so, we will use an in-house dataset coming from real world data. We will refer to this dataset as IH1 (in-house 1). This dataset contains 11,252 images of 14 different categories. These categories include different types of invoices, contracts, IDs, coupons, and others. Some categories are very unbalanced: while several categories have 1,000 documents or more, some others contain less than 200 documents. Because of this, the standard accuracy measure will be the mean class accuracy, *i.e.*, calculating the accuracy of each class independently and then averaging these scores. This measure is more significant than the document accuracy when classes are not balanced.

We are interested in understanding the effect of three parameters:

- **Image size.** The size of the image is an important factor in the quality of the descriptor. If the resolution is too small, then it will not be possible to capture all the necessary details of the image. On the other hand, if the image is too large, we may

obtain descriptors that capture excessively well the images and do not generalize. Also, although the size of the image does not affect the size of the final descriptor, it does affect the computation time. Through these preliminary experiments we will resize the images to 100,000, 500,000 and 1,000,000 pixels while preserving the aspect ratio. For reference purposes, the images on the datasets we will use range from $1,500 \times 2,500$ (3,750,000) to $2,500 \times 3,500$ (6,250,000) pixels.

- **Number of regions.** The use of the regions is a simple way to encode some basic structural information. These regions can be computed in a pyramidal way (as in the previous example, using a 1×1 plus a 2×2 and a 4×4 pyramid, producing 21 regions), but we can also compute only the highest level (for example, a 4×4 split producing 16 regions), or other combinations. Note that the number of regions directly affects the size of the histogram: doubling the number of regions will double the size of the final signature. We will test the following region partitions:
  - 1×1
  - 2×2
  - 4×4
  - 5×5
  - 6×6
  - 1×1 plus 2×2
  - 1×1 plus 2×2 plus 4×4
  - 1×1 plus 2×2 plus 4×4 plus 6×6.
- **Quantization intervals.** The number of quantization intervals also has an impact on the final signature size. However, choosing too few or too many intervals may degrade the quality of the descriptor. We will test the following quantizations:
  - $[1], [2], [3-4], [5-8], [9-16], \ldots, [65-]$.
  - $[1], [2], [3-4], [5-8], [9-16], \ldots, [129-]$.
  - $[1], [2], [3-4], [5-8], [9-16], \ldots, [257-]$.

We split the dataset evenly in a train and a test set. Each set contains half of the documents of each of the classes. The classification is performed with a linear SVM with an Stochastic Gradient Descent (SGD) solver inspired in Leon Bottou's implementation[2]. For the sake of simplicity, and since in this section we are interested in the effect of the descriptor parameters and not so much in the absolute classification score, the parameters of the SVM (the regularization factor λ and the number of iterations) have been validated over the test set. We believe this does not affect the analysis of the parameters of the descriptor. Experiments are repeated 5 times with different train/test partitions and the results have been averaged. In subsequent sections, when using public datasets and comparing the runlengths to other methods, the parameters of the SVM will be selected on a validation set.

To present the results, first we fix the quantization intervals and show the results as a function of the number of regions and the image size. The results can be seen in Figures 2.4 to 2.6. Then we fix the images size and show the results as a function of the number of regions and the quantization intervals. The results can be seen in Figures 2.7 to 2.9. We can outline the following points:

[2] http://leon.bottou.org/projects/sgd

Influence of the image size. If we focus on Figures 2.4 to 2.6 , we can observe how, in general, sizes of 500,000 and 1,000,000 work better than 100,000. This is particularly true when using few regions (1×1, 2×2, *etc.*). One possible explanation is that, if we are not going to enforce some structure through the regions, we need a high quality image to compensate. On the other hand, when we enforce a structure, the resolution plays a lesser role. Still, 100,000 is the size with the worst results in almost all cases. The differences between 500,000 and 1,000,000 are small and quite variable, and it is hard to discern a consistent pattern.

Influence of the quantization intervals. Figures 2.7 to 2.9 show the influence of the quantization under different scenarios. We can observe how the results do not follow a clear pattern. However, the results are so similar that any difference is hardly relevant: in the most extreme case, the difference is still less than 1% absolute. These differences get even smaller as the size of the images increases.

Influence of the number of regions. As shown in Figs. 2.4 to 2.9, adding some basic structure is quite important: using regions of 1×1, 2×2, or 1×1 plus 2×2 always obtains the worst results. 4×4 or 1×1 plus 2×2 plus 4×4 seem to obtain good results at a reasonable cost. Further increasing the number of regions slightly increases the accuracy, but also increases significantly the size of the final signature. Using spaial pyramid can also be important since the larger regions are less sensitive to a shifting of the document than smaller regions.

It is interesting to note how robust to these parameters the descriptor is: the difference between the best and the worst possible configurations we used is barely a 4% absolute with scores that are already above 90%. Therefore, and although we advocate to validate these parameters when using the descriptor in a real life scenario, we will fix the parameters and will use the same through all this dissertation unless explicitly noted. We will fix the image size to 500,000 pixels, the regions to a 1×1 plus 2×2 plus 4×4 pyramid, and the quantizaiton intervals to $[1], [2], [3-4], [5-8], [9-16], \ldots, [129-]$, producing a feature vector of 1,512 dimensions. The accuracy with this configuration is very close to the best accuracy, and all the parameters seem to offer a very reasonable trade-off in size, speed, and accuracy.

2.4.2 Evaluation on Public Datasets

For our experiments, we report results on two publicly available datasets, the NIST Structured Forms dataset [2], and the Medical Article Records Groundtruth (MARG) dataset [1]. The NIST dataset consists of 5,590 binary documents from 20 different classes of tax forms. The MARG dataset consists of 1,553 documents, first pages of medical journals, and is divided in 9 different layout types.

Figure 2.10 shows a few sample documents from different classes of both datasets. It is interesting to note how, albeit their very different nature, in both datasets the distinction between classes is based on the structural content of the document.

We report results on classification and retrieval tasks. As noted in the previous section, we will use our default 1,512-dimensional runlength descriptors to represent the document images. We will first show how the uncompressed descriptors can obtain state-of-the-art results on these datasets, and then we will show how descriptors can be compressed with PCA with minimum loss, as suggested when we introduced the descriptor. For learning the PCA transformation of NIST (resp. MARG), we will use a subset of 1,000 documents randomly drawn from NIST (resp. MARG).

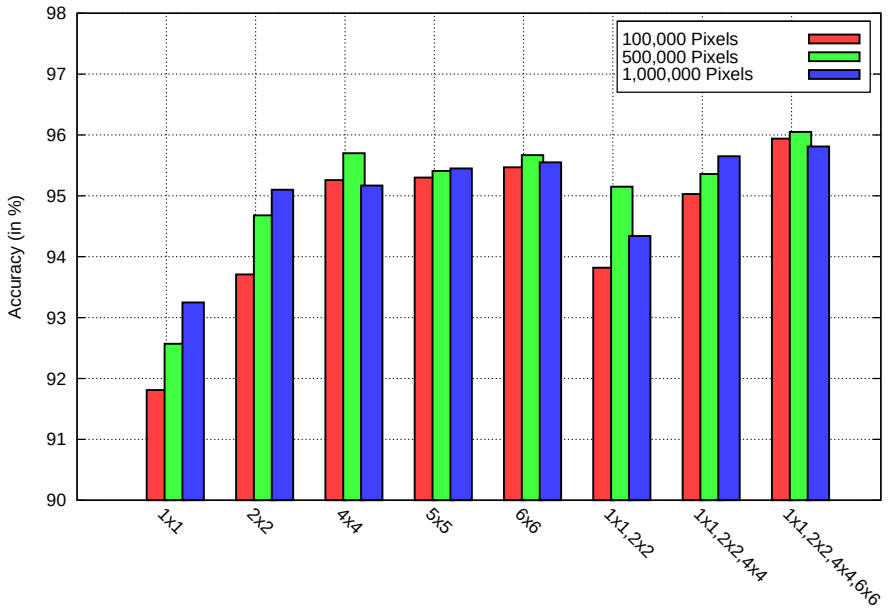


Figure 2.4: Accuracy as a function of the regions and the image size. The quantization intervals are fixed to [1], [2], [3-4], [5-8],. . . , [65-].

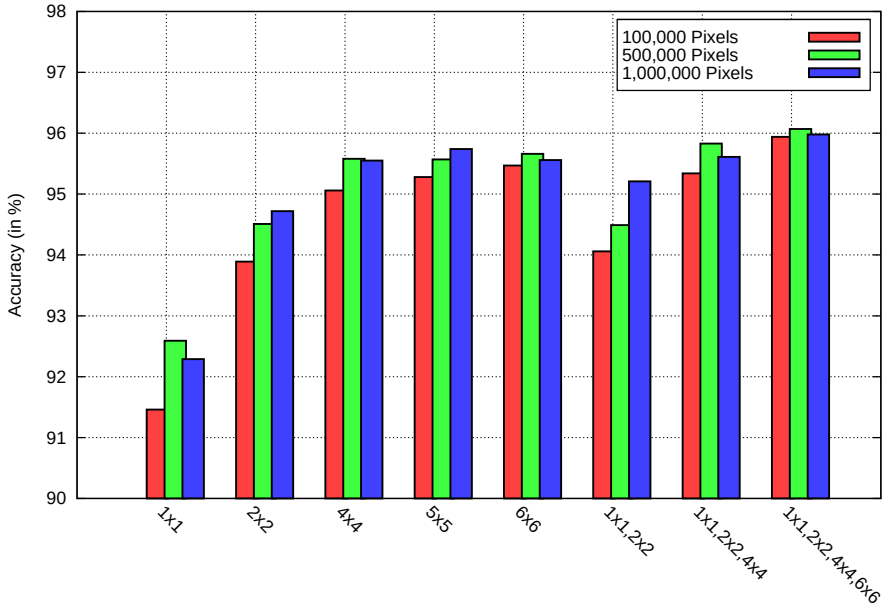


Figure 2.5: Accuracy as a function of the regions and the image size. The quantization intervals are fixed to [1], [2], [3-4], [5-8],. . . , [129-].

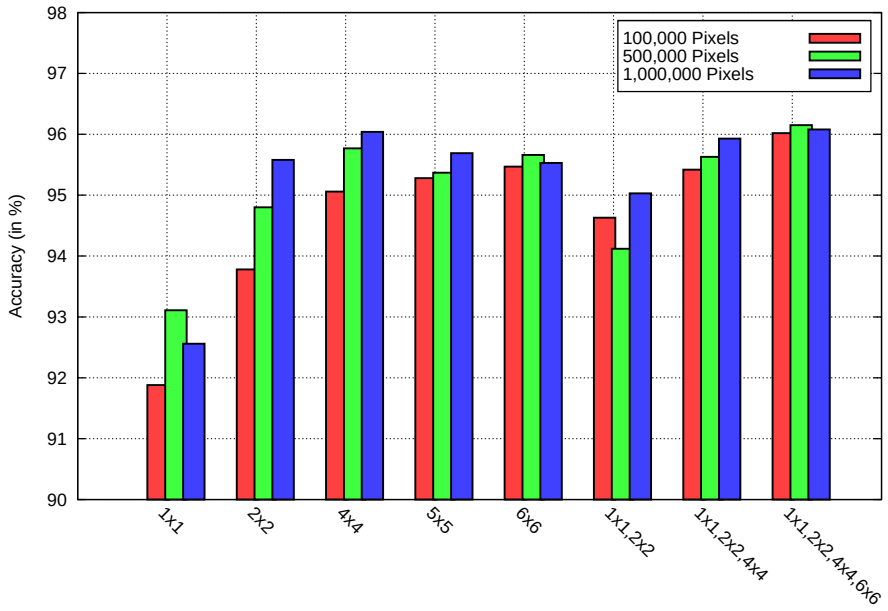


Figure 2.6: Accuracy as a function of the regions and the image size. The quantization intervals are fixed to [1], [2], [3-4], [5-8],. . . , [257-].

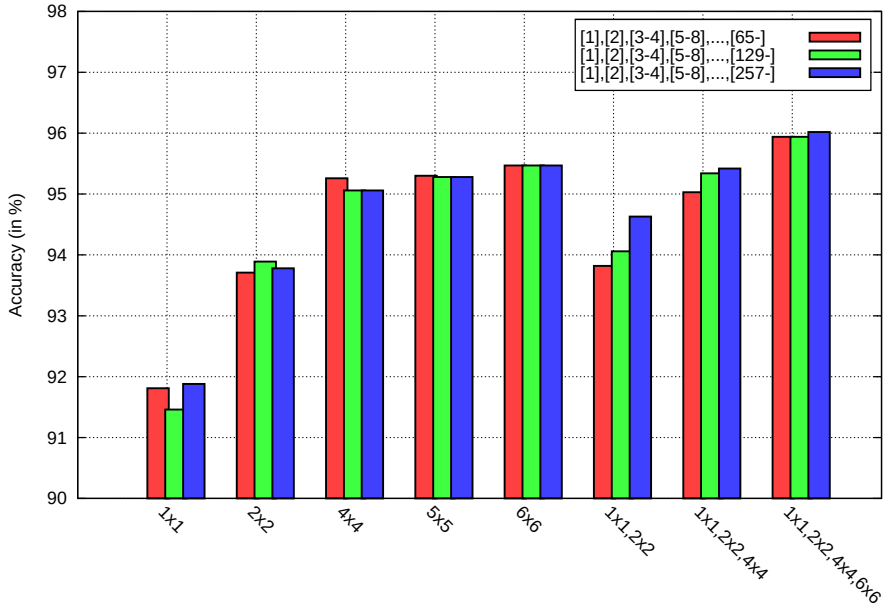


Figure 2.7: Accuracy as a function of the regions and the quantization intervals. The image size is fixed to 100,000 pixels.

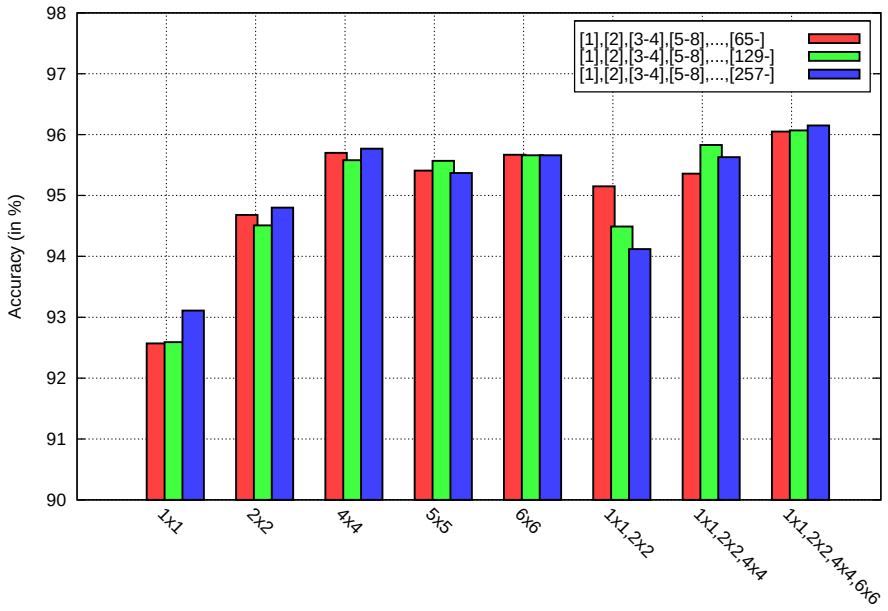


Figure 2.8: Accuracy as a function of the regions and the quantization intervals. The image size is fixed to 500,000 pixels.

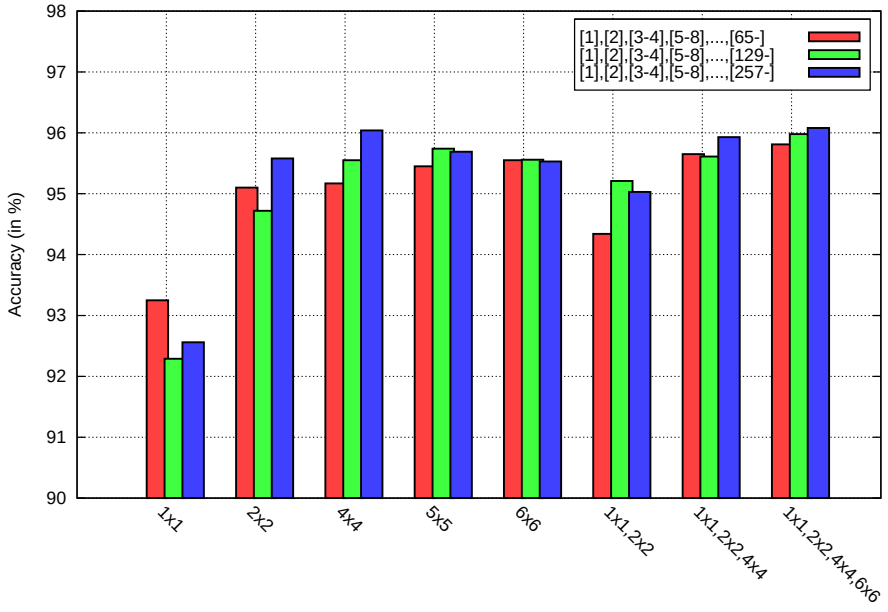


Figure 2.9: Accuracy as a function of the regions and the quantization intervals. The image size is fixed to 1,000,000 pixels.

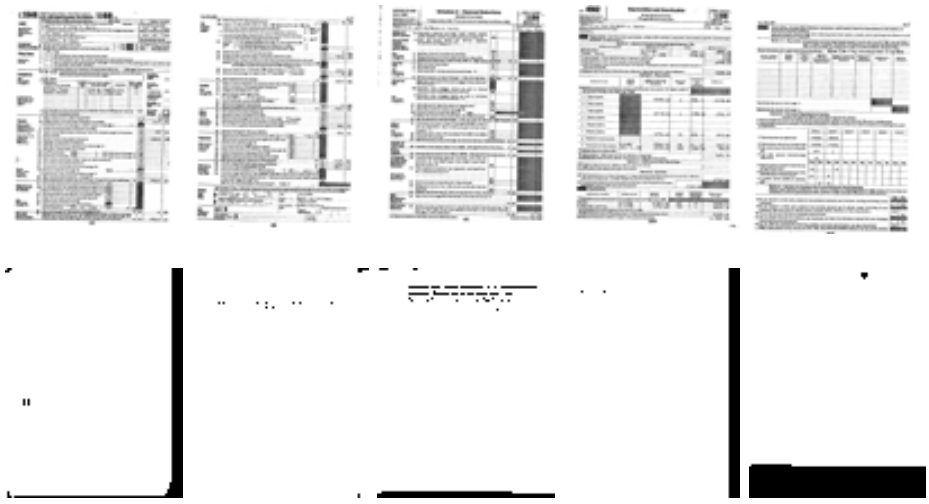


Figure 2.10: Samples of different classes drawn from NIST and MARG datasets. Top: NIST. Bottom: MARG.

Classification

Let us first describe the evaluation protocols we used on both document datasets:

NIST: To the best of our knowledge, the best reported results on the NIST dataset where those of [107], based on Viola-Jones-like features: a 99.82% classification accuracy with a 1-NN classifier. The protocol of [107] is as follows: the training set consists of 10 randomly drawn documents from each class (200 documents in total). The testing set comprises all 5,590 documents of the dataset, *including* the 200 documents in the training set. Note that since they use a nearest neighbor classifier, the 200 samples that appear in the training set will always be correctly classified, which makes the reported result an slight upper bound of the real accuracy.

We follow a very similar approach, but repeat the experiment 10 times with different training partitions and average the results, while the experiments of [107] were performed only with one fold. Although we report results following the optimistic evaluation of [107] for fair comparison purposes, we would like to remark that we also performed experiments without including the training samples in the test set and observed no significant decrease in the accuracy results.

MARG: A layout-based classification benchmark over the MARG dataset has been published in [124, 123]. They report results using different layout distance methods such as the Minimum Weight Edge Cover (EC), Assignment (ASS), or the Earth Mover Distance (EMD). They also explore the influence of the block distance used (overlap, Manhattan / Euclidean distance, *etc.*). We follow their procedure and use a 1-NN classifier with a leave-one-out evaluation protocol.

Note that, in both cases, we are using the Euclidean distance to compare the descriptors, and the only learning we performed is the unsupervised learning of the PCA projections.

Table 2.1 shows results of the uncompressed descriptor of 1,512 dimensions as well as the results after PCA compression. As previously hinted, we can significantly reduce the dimensionality of the vectors with minimum loss. On NIST, we can reduce the descriptors

down to 8 dimensions and still obtain a 99.99 classification accuracy. On MARG, we can reduce them down to 64 dimensions and still obtain results very close to the uncompressed baseline.

Table 2.1: Classification accuracy (in %) as a function of the number of dimensions on the NIST and MARG datasets.

dimensions	8	16	32	64	128	1,512
NIST	99.99	100	100	100	100	100
MARG	74.31	89.76	92.79	94.46	94.66	94.78

Table 2.2: Comparison of accuracy results (in %) on the NIST dataset.

NIST	
Method	Acc (%)
[107] Viola-Jones-based features	99.82
[113] Decision tree	99.70
[113] SOM	96.85
[93] Decision tree	98.82
[54] Density decomposition (reimp.)	**100**
Ours Uncompressed	**100**
Ours PCA 8D	**99.99**

Table 2.3: Comparison of accuracy results (in %) on the MARG dataset.

MARG	
Method	Acc (%)
[124] EC (Overlap + Manhattan)	92.6
[124] EC (Overlap)	91.8
[124] ASS (Overlap)	77.1
[124] EMD (Overlap)	79.8
[54] Density decomposition (reimp.)	91.18
Ours Uncompressed	**94.78**
Ours PCA 64D	**94.46**

Tables 2.2 and 2.3 compare our results with the reported state-of-the-art results published in [107] and [124], as well as the results on the NIST dataset published in [113] and [93][3]. We also compare our results to our implementation of the density decomposition method of

[3]Note that [113] and [93] use different evaluation protocols that make use of more training data, and so the comparison of the results should be exercised with caution.

[54]. The parameters of this density decomposition descriptor have been validated on the test set, which gives it an unfair advantage.

We can observe how, in both datasets, the uncompressed baseline outperforms the state-of-the-art reported methods (99.82% vs 100% on NIST and 92.6 vs 94.78% on MARG). In fact, reducing the signatures down to 8 dimensions on NIST and 64 on MARG still provides results superior to the state-of-the-art. The density decomposition method obtains excellent results on NIST, but its results drop when using a slightly more challenging dataset as MARG, even with the unfair advantage of having the parameters validated on the test set.

Furthermore, [107] reports times of "[...] only a few seconds [per page] with an unoptimized Java implementation [...]". With our non-optimized C++ code, it takes approximately 130ms to compute the descriptor of a NIST form. After compressing the NIST signatures down to 64 dimensions with PCA, it takes less than 100ms to compare the 5,590 documents against the 200 training samples, using a single CPU of a 3.16GHz Intel Xeon X5460 with 32GB of RAM. In the case of MARG, [124] reports times of 54s for the EC distance calculation and 62s for the ASS distance calculation using an Opteron CPU of 2.4GHz. Using descriptors of 64 dimensions as in NIST, we can compare the 1,553 documents in a leave-one-out strategy in less than 250ms.

Retrieval

In this section we evaluate our descriptor in retrieval tasks. We will follow a leave-one-out strategy: we will query each dataset item in turn, and we will rank all the remaining documents. As in the classification experiments, we will use the Euclidean distance for the uncompressed and the PCA compressed descriptors. The reported result will be the mean Average Precision (mAP) of all the queries. The Average Precision can be interpreted as the area beneath the Precision-Recall curve, and it is a standard measure in retrieval systems. We will follow the same procedure for NIST, MARG, and the IH1 dataset. In the case of IH1, however, we will not use all the documents in the dataset. To avoid the unbalance problem, we will sample 200 documents per class and perform the retrieval on that subset of the dataset.

The retrieval results can be seen in the first three rows of Table 2.4. One may note how the descriptor still achieves very high scores on the NIST dataset. One may also note the low retrieval results obtained on the MARG and IH1 datasets, compared to the high results obtained in classification.

Let us focus on MARG. A plausible explanation to these results lies in the way MARG is groundtruthed. The criteria used to define the nine categories are the position and shape of some information as the title, authors, affiliation, abstract, *etc.* (see Fig 2.12). However, the number of columns is *not* taken into account. Indeed, documents of any given layout category exist in one, two, or, sometimes, even three columns format. This may lead to bad retrieval results since the number of columns has a much higher visual footprint than the exact position of small fields such as the title or the author. When retrieving documents of, *e.g.*, one column, it is reasonable to expect the two-column documents of that category to be badly ranked, since one-column documents of other categories will be ranked first, thus significantly dropping the retrieval results as we have experienced. We can observe this in Figure 2.11, where we show the precision at k as a function of k for four random MARG queries. We can observe how the first results are very accurate, but then the precision begins to drop as the visual layout of the relevant documents becomes more and more dissimilar from the query.

This raises an interesting point. In some cases, the visual differences between pages do not match their semantic differences, at least not in a direct way. When performing

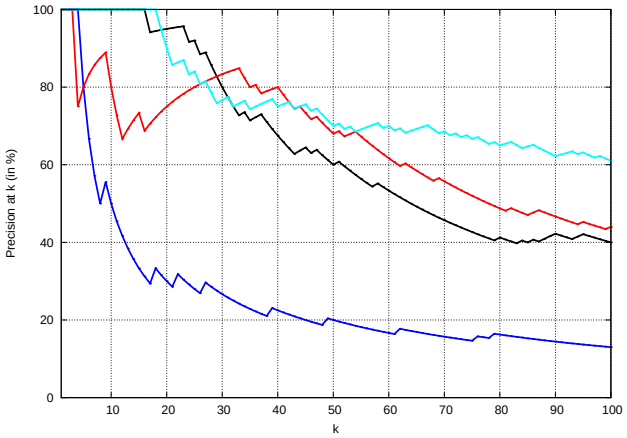


Figure 2.11: Precision at k as a function of k of four random MARG queries.

classification, this is not a big problem, since we perform some supervised learning, either directly – such as with an SVM classifier, learning what particular visual aspects make the documents different – or indirectly – such as with a k-NN classifier, where we expect to have at least one relevant document similar enough in the training set. However, when we perform unsupervised retrieval as in this case, we have not learned anything about the documents, and this may lead to a long tail of bad results. Unfortunately, this is not something only relevant to visual features, and may as well be a problem with structural or textual-based descriptors. In general, unless documents have a very rigid structure (such as the forms of NIST), unsupervised retrieval will not produce meaningful results beyond the very first retrieved items, as we can see in the MARG and IH1 results. Although out of the scope of this chapter, we would like to note that frameworks such as Metric Learning (see, *e.g.*, [129, 9]) would significantly help in this scenario if supervised training data is available. In a nutshell, we can define a similarity measure between signatures $s(q, d) = q^T W d$, where q is a query and d is a document in the dataset. Note that this reduces to the dot-product when $W = I$. Then, the goal is to learn W to enforce that $s(q, d^+) > s(q, d^-)$, where d^+ is relevant to q and d^- is not[4]. This can be learned efficiently, for example, using a large-margin framework and Stochastic Gradient Descent. As a sample of the potential improvements, see the Table 2.5, where a metric learning approach based on [9] was applied to the MARG dataset descibed with our runlength descriptors. In chapter 6 we will revisit metric learning as well as other supervised approaches to highlight the importance of learning when performing retrieval.

Finally, although the mean average precision is a good measure to evaluate a retrieval system, in most typical scenarios we are only interested in the quality of the top results, *i.e.*, the precision at k, where k is usually small, *e.g.* 5 or 10. This evaluation metric would significantly palliate the negative influence of semantically relevant document whose visual appeareance significantly differs from the query.

Table 2.6 shows Precision at 5 results on the MARG and IH1 datasets. We can observe how the results are significantly better than the mean average precision, suggesting that the first retrieved items are mostly correct and it is the long tail that is causing the results to drop in the case of using mAP. Also, as before, these results could also be significantly

[4]Alternatively, we can define a Mahalanobis-based distance as $d(q, d) = (q - d)^T W (q - d)$, and enforce $d(q, d^+) < d(q, d^-)$.

Table 2.4: Mean average precision (in %) as a function of the number of dimensions on the NIST, MARG, and IH1 datasets.

dimensions	8	16	32	64	128	1,512
NIST	99.99	100	100	100	100	100
MARG	27.52	30.09	31.55	31.95	32.01	31.97
IH1	62.01	68.71	68.66	67.90	67.45	66.82

Table 2.5: Mean average precision (in %) as a function of the number of dimensions on the MARG dataset, without and with metric learning.

dimensions	8	16	32	64	128	1,512
MARG (unsupervised)	27.52	30.09	31.55	31.95	32.01	31.97
MARG (Metric Learning)	87.08	86.99	89.91	86.49	86.09	-

improved when combined with supervised metric learning.

Table 2.6: Precision at 5 (in %) as a function of the number of dimensions on the NIST, MARG and IH1 datasets.

dimensions	8	16	32	64	128	1,512
NIST	100.0	100.0	100.0	100.0	100.0	100.0
MARG	64.97	77.63	82.09	84.11	84.73	84.70
IH1	85.88	90.87	91.58	91.68	91.53	91.16

2.5 Conclusions

Through this chapter we have introduced a novel document image representation based on runlength histograms. Given a region, we compute a histogram of the lengths of the runs of black and white pixels in several directions. This is fast to compute, encodes more information than just a density decomposition, and is more resilient to noise. We can compute the histograms using a spatial pyramid decomposition to add some important global structure to the representation. This descriptor is extremely general since it does not depend on the type of document and is fast to compute. Although this descriptor has some parameters that need to be adjusted (*e.g.*, the image size or the quantization intervals), we showed experimentally in one in-house dataset that the influence of the parameters is relatively small. We validated the best parameters on this in-house dataset and used those parameters through all other experiments in the chapter, even when dealing with different datasets. Even without fine-tuning the descriptor to each dataset, we obtained state-of-the-art results in two public datasets of different nature and quality on classification and retrieval tasks, showing

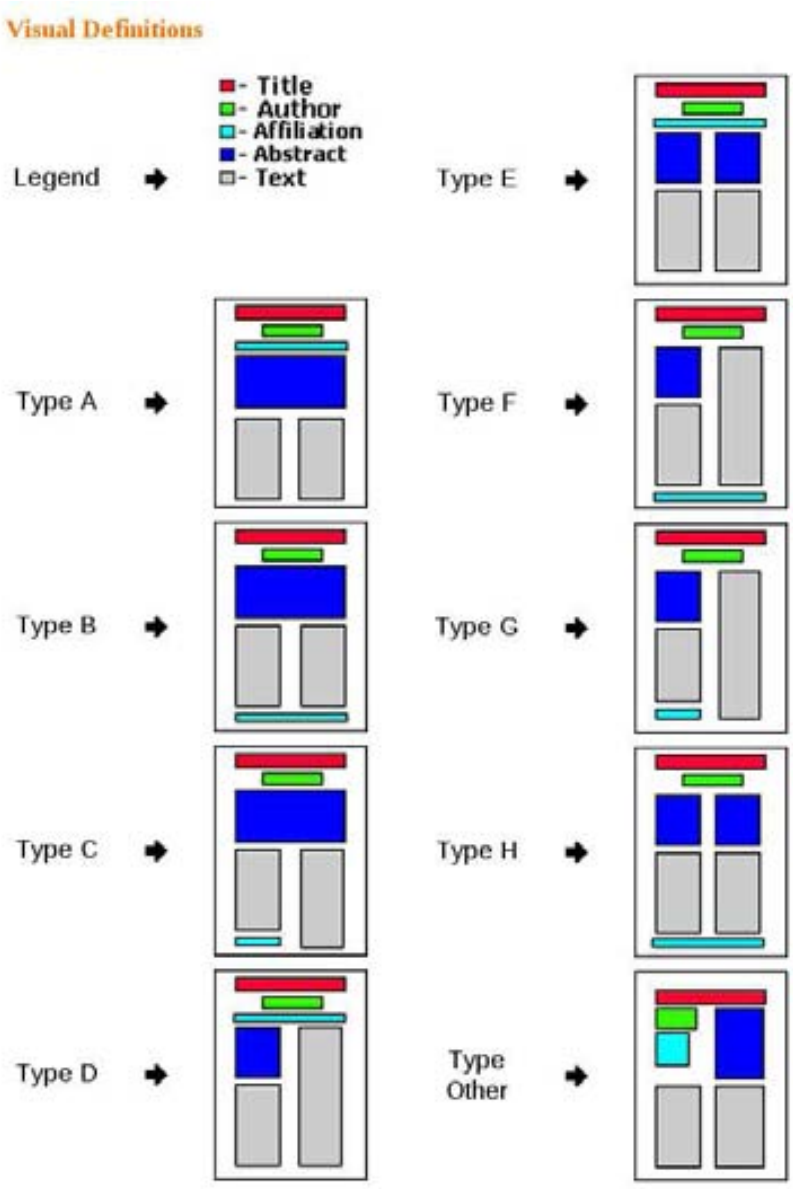


Figure 2.12: Layout categories in MARG, obtained from `http://marg.nlm.nih.gov/gtdefinition.asp`. The number of columns is *not* relevant to decide the category of a document.

the generality and soundness of the descriptor. The results were not only better in accuracy: the cost of computing the descriptor, as well as the retrieval and classification times, were significantly better than the ones from other state-of-the-art methods.

We observed that some retrieval tasks obtained bad results when relevant items are very different from the query, and we showed how it can be improved through supervised learning when labeled data is available. We will come back to this point in chapter 6, which deals with supervised learning for ranking purposes. We note however that this is not a problem of our particular descriptor, and it is shared by most if not all document representations in the literature. Not only that, but applying these supervised learning techniques is particularly easy in our fixed-length feature vectors, while learning to rank on other typical representations such as graphs can become an extremely difficult task.

Finally we note that, although this descriptor has obtained very good results in our datasets, it is restricted to documents that contain one single page. However, in the digital mailroom it is very common to find documents spanning through multiple pages. For example, an ID card can be found together with a two-pages form and a handwritten letter, and all these pages belong to one single document that needs to be represented as a whole. The next chapter deals precisely with this problem.

Chapter 3

Multiple-Page Document Representation[1]

One challenge that has burdened high-volume document processors has been the need to distinguish individual documents from one another. This is especially dificult when dealing with groups of multi-page documents, such as those frequently found in loan processing applications, or in the typical mailroom. While it is usually a simple matter for a human to say, "this is where a loan application document ends, and here is where the credit report begins," that determination has been much more dificult for an automated system. Yet by necessity of sheer volume, processors have been forced to feed massive amounts of unsorted pages into document scanners, arranged by little more than the order they arrive.

Classification and Separation [4] (White Paper) – KOFAX

3.1 Introduction

In chapter 2 we have focused on representing documents that consist of one single page. However, there are many scenarios where documents may contain multiple pages. One classic example are applications, where we may find pages of types as different as IDs, forms, cover letters, *etc.*, all in the same document. At the same time, some of those types may span several pages in themselves or be repeated, *i.e.*, 2-page forms, many IDs, *etc.*

One practical problem involving these documents consists in separating all the different subcategories in a document: these pages are IDs, this one is a form, *etc.* This is a very important step before trying to understand the data and is a task in high demand. Unfortunately, this is an extremely difficult problem. In fact, most current solutions involve a manual labelling of the document pages, either with separators sheets at boundary pages, barcodes, or both (see Fig 3.1). Then, after scanning, those barcodes or separators can be analyzed to correctly separate the data. Both [3] and [4] offer some interesting insights about how these basic approaches are used in practice by the industry. Unfortunately, these methods are error prone and costly: according to [109], the error rate for this process can

[1] Parts of this chapter published in *A. Gordo and F. Perronnin. A Bag-of-Pages Approach to Unordered Multi-Page Document Classification. In ICPR, 2010.*

be as high as 8%, and the cost of sorting and inserting the separator sheets can be as high as 50% of the document preparation cost.

Figure 3.1: Example of separator sheets with barcodes

Surprisingly, methods that make use of computer vision and machine learning techniques to fully automate document separation are still scarce. In [26], pages are clustered according to whether they belong to the same type of subdocument or not, effectively separating the subdocuments. To do so, they take into account the structure of the pages (particularly the page numbers) as well as the text to define a similarity measure used in the clustering. In [109], pages are first represented as bag of words using the output of an OCR. Then, the document is represented as a sequence of pages that has to be segmented into subdocuments. Markov Chains and SVMs are then used to learn and perform the segmentation. Note that both methods require the pages to be correctly ordered, which is not always the case after scanning.

A different but very related problem is that of multiple-page document classification. Given a document consisting of several subdocuments of potentially several pages each, we have to decide which is the category of the main document. This is particularly useful in a mailroom, where, for example, depending on the category, the document will be sent to different workflows. Note that both problems are very interrelated. Being able to correctly separate the subdocuments would help us to categorize the whole document. Also, if we know the category of the document, we may use this information to guide the segmentation.

As before, computer vision approaches to this problem are very scarce. The most similar scenario is that of [45], where documents spanning multiple ordered pages are modeled and classified using a Hidden Markov Model over an OCR text representation.

Here, we tackle the more general problem of unordered multiple-page document representation and classification. Documents contain a variable number of unordered pages, and the goal is to correctly represent and classify the documents as a whole. Those pages may belong to different subcategories, but those subcategories may not be well defined, or may not even be defined at all. Also, documents of the same category may contain a different number of pages. We will also assume that we have a set of labeled documents for training purposes. Those documents are labeled at whole document level, but not at page level. Despite its importance, we are unaware of any previous work dealing with this problem except that of [49], upon which this work stands.

Through this chapter, we will first propose a very simple classification baseline based on the assumption that pages belonging to different categories are visually dissimilar. However, this assumption does not hold true in general and the performance of the baseline is limited. Then, we will lift this assumption and propose a Bag of Pages (BOP) approach, where, similarly to the Bag of Words or Bag of Visual Words, we will represent the documents as a histogram of pages. One advantage of this representation is that it can be used for other tasks such as retrieval or clustering. Unfortunately, as we will see through the experimental section, the performance of the vanilla BOP approach improves only slightly over the baseline, and only sometimes, despite being theoretically more sound. We believe the reason is that, in an implicit way, the baseline is performing a supervised clustering of the data, missing in the BOP representation. Motivated by this, we explore two different methods to improve the BOP representation based on performing a supervised clustering of the vocabulary words. While the first method has been used before in the literature, we are not aware of the second method being used in a supervised clustering context. As we will see, both methods significatively improve the vanilla BOP results, with the novel approach obtaining the best results.

Although we focus on visual features, on a final set of experiments we show how these representation approaches can also be used with textual features, in our particular case with a bag of words histogram. Finally, combining the visual and textual representations yields even larger improvements, showing that both features contain complementary information.

The rest of this chapter is organized as follows. Sections 3.2 and 3.3 introduce the baseline and the vanilla Bag of Pages methods. These methods are experimentally tested in section 3.4 in classification tasks on two challenging datasets, using the runlength features introduced in chapter 2. Section 3.5 introduces the Bag of Pages improvements, which are tested in section 3.6. Finally, section 3.7 shows retrieval results, and section 3.8 performs classification tasks using textual features instead of the visual features used throughout the chapter.

3.2 Baseline

First, we will propose a simple baseline for classifying multipage documents. Note that this approach does not generate a document feature vector, and so its use beyond classification tasks is very limited. Our baseline system is based in the following assumption: pages belonging to different categories are visually dissimilar. Under this assumption, we can turn the document classification problem into a page classification problem. At training time, we learn page-level classifiers:

1. Extract page-level representations for each page of each training document.
2. Propagate the document-level labels to the individual pages.
3. Learn one page-level classifier per document category using the features of step 1 and the labels of step 2. We assume that, at testing time and given an input, this classifier can produce either probabilities or scores of that input for all the classes. In practice, we will use an SVM trained with SGD as in chapter 2, section 2.4.1.

At runtime, a document is classified as follows:

1. Extract one feature vector per page.
2. Compute one score per class per page. This is reminiscent of the attributes / classemes framework [102, 122], where an image q is represented with a K-dimensional vector $\hat{q} = [s(q, c_1), s(q, c_2), \ldots, s(q, c_K)]$, where K is the number of classes, c_k is class k, and $s(q, c_k)$ is the relevance of the image q with respect to class k. In general, these

relevance scores are learned using a large-margin framework, *e.g.* by training one binary SVM for each class [76, 126, 122]. This is the approach that we follow here.

3. Aggregate the page-level scores into document-level scores for each document class, and select the class with the highest score. If we let p_t be the t-th page of document d, $1 \leq t \leq T$, then the class c of document d is

$$c = \underset{c_k}{\operatorname{argmax}} \frac{1}{T} \sum_{t=1}^{T} s(p_t, c_k). \tag{3.1}$$

We found that a simple average-pooling obtained the best results, but other fusion schemes could also be used.

In most of the scenarios we have encountered, the assumption underlying our baseline system is not verified, *i.e.* two pages may be very similar (from a visual and / or textual perspective) or even identical and still belong to documents with different category labels. For instance, the copy of an ID card may be attached to different requests which have to be processed by different workflows

3.3 A Bag of Pages Approach

Let us assume the existence of page-level categories which are potentially shared across document categories. On the datasets we considered, typical page-level categories could be: "handwritten letter","typed letter", "form X", "ID copy", "phone bill", *etc.* If we had training data with page-level labels, then we could learn page classfiers and then represent an image as a histogram of the number of occurrences of each page category. For instance, a 3 pages document could be described as: 1 "handwritten letter" + 1 "subscription form" + 1 "phone bill". This document-level representation might be more amenable to classification than the original representation.

However, this approach is impractical for two reasons. First, identifying manually page-level categories is a non-trivial task. For instance, should we put a driver's license and a passport in the same "ID" category or in different categories? Second, even if page-level categories were well-identified, one would need to gather labeled training material for each page category which is a slow and tedious process. On the other hand, we can try to discover such page categories automatically through an unsupervised process. The next section describes a plain vanilla implementation of this approach.

3.3.1 Bag of Pages

Our Bag of Pages approach is inspired by the Bag of Visual Words framework used in image representation. In the Bag of Visual Words (BOV) framework [115, 27], images are represented by a histogram of quantized local features. First, interest points are detected in the images, and local features, such as SIFT [82] or SURF [10], are extracted from the images at those points. As opposed to the Bag of Words used by the text analysis community, where these features are discrete textual words, here we deal with with a continuous feature space. As a consequence of this, features need to be quantized; a visual vocabulary or codebook has to be learned offline, typically by clustering these local features. The *de facto* method to cluster these features is k-means, although other approaches such as learning a hierarchical k-means [90] or a Gaussian Mixture Model (GMM) [42] are also common. Following the approach of [27], images are then represented by a fixed-length histogram that counts the number of local features assigned to each of the codebook centres, or soft assigned

in the case of GMM. These image descriptors can then be used for categorization tasks (typically using SVMs), retrieval, *etc.* A flowchart of the process can be seen in Figure 3.2. Several improvements to this framework have been proposed for all the stages, from interest points detection and description to vocabulary construction, whole image representation, and classification.

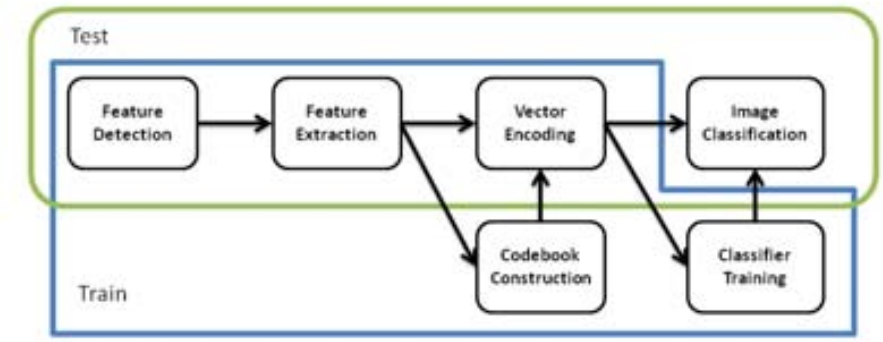


Figure 3.2: Flowchart of the basic BOVW framework

In a very similar way, documents of multiple pages can be seen as a collection of pages and represented by a histogram of quantized pages (see Fig. 3.3).

(a) (b)

Figure 3.3: Conceptual similarities between the Bag of Visual Words framework (left) and the Bag of Pages (right). Similarly to the Bag of Visual Words, multi-page documents are represented as an orderless bag of document pages. Left image courtesy of Li Fei Fei.

The first stage in the BOV framework is to find relevant points to describe, either by dense sampling or by using interest point detectors [86]. In the Bag of Pages approach, the relevant points that we want to describe are the pages of the document. Then, pages have to be described with a feature vector in the same way that we would describe interest points

with descriptors such as SIFT. We will focus our experiments on pages represented with the runlength descriptor introduced in chapter 2, but other descriptors could be used. As an example, we will also carry on one experiment using textual features represented with a Bag of Words vector instead of visual ones.

Then, we cluster the pages to discover a vocabulary, in the same way that we do in the BOV framework. Instead of k-means, we will use a Gaussian Mixture Model, which is another common approach to vocabulary construction [42]. Using a Gaussian mixture generative model allows us to soft assign the pages to the vocabulary words using the posterior probabilities in a very natural way, instead of using hard assignment. This usually yields a small improvement over the k-means and hard assignment combination. Finally, vectors are L1- and square-root normalized. As we mentioned in chapter 2, the square root corresponds to an explicit embedding of the Bhattacharyya similarity when using the dot-product, and can significantly improve the results at no cost.

Beyond Counting: The Fisher Vector Framework

The standard way in which BOV aggregates the features can be seen as simply counting the occurrences of each vocabulary word in the image. However, other approaches to aggregate the features into a global image representation have been suggested. Examples of this are Super-Vector encoding (SV) [135], Locality-constrained Linear Coding (LLC) [128] or the Fisher Vector (FV) [96]. In the independent analysis of [21], the Fisher Vector was shown to be the best performing representation.

Intuitively, the FV can be understood as a way to encode not only the word count but higher order statistics such as the position and spreadness of the features with respect to the generative model, in our case, the GMM we used to compute the vocabulary. The Fisher Vector can also be understood as an explicit embedding of the Fisher Kernel [59]: computing the dot-product between the FV representation of two samples is equivalent to computing the Fisher Kernel between the original samples.

As noted in [21], encoding higher order statistics seems to be a key aspect of the success of the FV. When dealing with multiple-page documents, where every document consists of only a few pages (in our particular datasets, the average number of pages per document is less than 4), capturing this higher order information seems even more important, to compensate for the lack of data. This is one of the reasons we decided to use the FV framework to encode our Bag of Pages. In this section we will first review the Fisher Kernel in its natural form of [59], and then we will follow [96] to show the closed form of the FV in the case of using a Gaussian Mixture Model generative model as we do in the Bag of Pages approach.

Let $X = \{x_1, x_2, \ldots, x_T\}$ be the set of T local descriptors of dimension D obtained from an image. In our case, X would be a document and $\{x_1, x_2, \ldots, x_T\}$ would be the T pages of the document described with runlength descriptors. We assume that the generation process of X can be modeled by a probability density function u_λ with parameters collectively denoted by λ. Then X can be described by the following gradient vector:

$$G_\lambda^X = \frac{1}{T} \nabla_\lambda \log u_\lambda(X). \tag{3.2}$$

The gradient of the log-likelihood describes the contribution of the parameters to the generation process. The dimensionality of this vector depends only on the number of parameters in λ, not on the number of patches T. A natural kernel on these gradients is [59]:

$$K(X, Y) = G_\lambda^{X'} F_\lambda^{-1} G_\lambda^Y \tag{3.3}$$

where F_λ is the Fisher information matrix of u_λ:

$$F_\lambda = E_{x\sim u_\lambda}\left[\nabla_\lambda \log u_\lambda(x) \nabla_\lambda \log u_\lambda(x)'\right]. \tag{3.4}$$

Unfortunately, explicitly calculating the Fisher Kernel is a costly operation and its direct application is generally unfeasible.

As F_λ is symmetric and positive definite, it has a Cholesky decomposition $F_\lambda = L'_\lambda L_\lambda$ and $K(X,Y)$ can be rewritten as a dot-product between normalized vectors $\mathcal{G}_\lambda$, with $\mathcal{G}^X_\lambda = L_\lambda G^X_\lambda$. We will refer to $\mathcal{G}^X_\lambda$ as the Fisher Vector of X. Learning a kernel classifier using the Fisher Kernel (Equation (3.3)) is equivalent to learning a linear classifier on the Fisher Vectors $\mathcal{G}^X_\lambda$.

We follow [96] and choose u_λ to be a Gaussian mixture model (GMM). We collectively denote $\lambda = \{w_i, \mu_i, \Sigma_i, i = 1, \ldots, K\}$ where w_i, μ_i, and Σ_i are respectively the mixture weight, mean vector, and covariance matrix of Gaussian u_i. We assume that the covariance matrices are diagonal and we denote by σ_i^2 the variance vector. Then,

$$u_\lambda(x) = \sum_{i=1}^{k} w_i u_i(x), \tag{3.5}$$

$$u_i(x) = \frac{exp\{-\frac{1}{2}(x-\mu_i)'\Sigma_i^{-1}(x-\mu_i)\}}{(2\pi)^{D/2}|\Sigma_i|^{1/2}}. \tag{3.6}$$

The GMM u_λ is trained on a large number of images using Maximum Likelihood (ML) estimation. It is supposed to describe the content of any image. We assume that the x_t's are generated independently[2] by u_λ and therefore:

$$G^X_\lambda = \frac{1}{T}\sum_{t=1}^{T} \nabla_\lambda \log u_\lambda(x_t). \tag{3.7}$$

We consider the gradient with respect to the mean and standard deviation parameters (the gradient with respect to the weight parameters brings little additional information).

Let $\gamma_t(i)$ be the occupancy probability of descriptor x_t to Gaussian i:

$$\gamma_t(i) = \frac{w_i u_i(x_t)}{\sum_j^k w_j u_j(x_t)}. \tag{3.8}$$

Let $\mathcal{G}^X_{\mu,i}$ (resp. $\mathcal{G}^X_{\sigma,i}$) be the D-dimensional gradient with respect to the mean μ_i (resp. standard deviation σ_i) of Gaussian i. Mathematical derivations [96] lead to:

$$\mathcal{G}^X_{\mu,i} = \frac{1}{T\sqrt{w_i}}\sum_{t=1}^{T}\gamma_t(i)\left(\frac{x_t-\mu_i}{\sigma_i}\right), \tag{3.9}$$

$$\mathcal{G}^X_{\sigma,i} = \frac{1}{T\sqrt{2w_i}}\sum_{t=1}^{T}\gamma_t(i)\left[\frac{(x_t-\mu_i)^2}{\sigma_i^2}-1\right], \tag{3.10}$$

where the division between vectors is as a term-by-term operation. The final gradient vector $\mathcal{G}^X_\lambda$ is the concatenation of the $\mathcal{G}^X_{\mu,i}$ and $\mathcal{G}^X_{\sigma,i}$ vectors for $i = 1 \ldots K$ and is therefore $2KD$-dimensional.

In [99], the Fisher Kernel framework was further improved for categorization tasks. It was shown that $L2$ normalizing the $\mathcal{G}^X_\lambda$ vectors reduces the weight of the background in the signatures and yields better classification results. It was also shown that a by-dimension square root of the signature vectors will help to 'unsparsify' the signatures and will therefore obtain better results when using the dot-product as a similarity measure.

[2] For an extremely interesting work on lifting the i.i.d. assumption, please refer to [25].

3.4 Bag of Pages Experiments

3.4.1 Datasets and Features

We experiment on two in-house datasets coming from the same source. The first is a small dataset composed of 2,060 documents and 10,097 pages, and divided in 6 categories. The number of pages per document is quite variable, ranging from 2 to 14. The number of documents per class is also quite unbalanced: from 39 in one class to more than 1,000 in another. We will refer to this dataset as IH2 small (In-House 2 small). The second dataset is larger: it contains 19,178 documents and 57,530 pages, and it is divided in 19 different classes. We will refer to this dataset as IH2 large. As in IH2 small, this dataset is quite unbalanced, both in number of pages per document and in number of documents per class. The 6 classes of IH2 small are a subset of the 19 classes of IH2 large, but the documents are disjoint. Both datasets come from real-world data, and the documents belong to classes such as "ID", "phone bill", several forms, *etc*. Most of the documents contain text in French. Some of the categories are extremely similar from a visual point of view, and discriminating them correctly without textual information is almost impossible.

Pages are represented using the runlength histograms introduced in chapter 2. We will use the default representation: resizing the images to 500,000 pixels, using a 1×1 plus a 2×2 and a 4×4 pyramid, and $[1], [2], [3-4], \ldots, [129-]$ quantization intervals. As also noted in chapter 2, histograms are L1 and square-root normalized. Finally, we reduce the dimensionality of the histograms with PCA, since this may have a large effect, particularly in the Bag of Pages approaches. Note that, as mentioned in the previous section, the covariance matrix of the GMM is typically assumed to be diagonal since it is much less expensive to learn. Since the covariance matrix will not capture the variance between different dimensions, it is very convenient to have low-level features where the dimensions are uncorrelated. However, as discussed in chapter 2, this may not be the case when using the multi-scale runlength descriptors. Therefore, we can use PCA to decorrelate the dimensions and, as a consequence, obtain a GMM that better models the runlength features. Finally, the PCA dimensionality directly affects the final size of the signatures in the FV approach, and is convenient to have low-dimensional low-level features to control the final signature size.

3.4.2 Classification Evaluation Protocol

We split the datasets in train/validation/test partitions. 40% of the documents of each class are assigned to train, 30% to validation, and 30% to test. Experiments are repeated 5 times with different partitions and the results are averaged. For performing unsupervised learning (PCA and vocabulary construction) on the small dataset, we select 5,000 random pages for the baseline (since it is page-based) and 1,000 documents for the Bag of Pages (since it is document-based). In the large dataset, we randomly draw 10,000 pages for the baseline and 2,000 documents for the Bag of Pages.

For the Bag of Pages approach, we compute a probabilistic codebook using GMM instead of k-means and use soft assignment, which leads to a small improvement in the results. The BOP histograms are then L1- and square-root-normalized, which consistently improves their results.

Classification is done with a one-vs-rest SVM with a SGD solver. The parameters of the solver (the number of iterations and the regularization factor λ) are validated on the validation set. When dealing directly with SVM scores as we do in the baseline, it is important to calibrate the output scores of the classifiers so they are directly comparable. One popular approach consists in calibrating the scores as a post-process, as done, for example, in [38]. Another approach consists in selecting only a subset of negative samples for each

class, proportional to the number of positive samples available, as done for example in [122]. This is further asserted in [95], where it is shown that balancing the positive and negative data is very important to obtain state-of-the-art results when using one-versus-rest SVM classifiers. We will follow the latter approach. Through validation, we found that the best positive:negative ratios were 1:1 for the small dataset and 1:2 for the large dataset. This had a huge impact in the baseline accuracy, and a noticeable impact on the BOP approaches.

Finally, since the classes are quite unbalanced, instead of reporting the global document accuracy, we report the mean class accuracy, *i.e.*, computing the accuracy of each class independently and then averaging the results. This gives a more meaningful measure when the classes are unbalanced as in this case.

3.4.3 Baseline Results

Table 3.1 shows the baseline results on the IH2 small and IH2 large datasets as a function of the number of PCA dimensions of the runlength descriptors We can observe how reducing the number of dimensions with PCA can lead to a small improvement in the results, about 2% absolute on both datasets. Also, as expected, the results on the large dataset are significantly worse than those of the small dataset.

Table 3.1: Mean class accuracy (in %) as a function of the number of PCA dimensions on the IH2 small and IH2 large datasets.

dimensions	128	256	512	1,024	1,512
IH2 small	55.08	56.26	55.80	53.21	54.60
IH2 large	36.87	40.02	40.87	41.72	39.69

3.4.4 Bag of Pages Results

We now report results using the Bag of Pages approach. Figure 3.4 shows the results obtained with the BOP approach using both soft assignment and FV on the IH2 small and IH2 large datasets as a function of the vocabulary size. We show the results after compressing the page runlengths with PCA for several dimensionalities. We also compare with the best possible baseline from the previous section. We can highlight the following points:

- The PCA dimensionality has a very significant impact in the results. The differences between the best and worst PCA dimensionality options can be of more than 10 points. In general, compressing the descriptors down to 32 or 64 dimensions seems to be the best option in the IH2 small dataset. However, in the more complex IH2 large, the best results are obtained when using the FV with runlengths compressed to 256 or 512 dimensions, and 32 or 64 dimensions perform significantly worse. Therefore, although it seems that more difficult problems require higher dimensional features, the right dimensionality of PCA should be validated, which is a drawback of this method.
- The FV encoding obtains significantly better results than the soft assignment with much smaller vocabularies. This is in line with previous studies comparing the FV with other encoding methods such as the soft assignment. In the recent [21], the FV encoding was shown to perform best amongst many state-of-the-art encoding methods.

- The BOP approaches can improve over the baseline results, but only slightly and only sometimes. For example, in the IH2 large dataset (Figure 3.4b), the FV improves the baseline only about a 3% absolute, while the soft assignment obtains results significantly worse than the baseline.

Through the next section we will discuss the reasons behind the limited improvement of the BOP approaches over the baseline, as well as improve the BOP representation in two related but indepentent ways.

3.5 Improving the Bag of Pages

In the previous section we observed how, despite being a more principled approach, the BOP representation did not consistently improve over the baseline. We believe one reason is that the baseline is already performing very well, given the difficulties of the datasets.

Our baseline can be understood as an average pooling over a classeme representation of the pages. After averaging the pages, the class with the highest score is selected. These classemes have been learned in a supervised way (albeit with "noisy" labels, since labels were propagated from documents to pages) and contain very important information, not only about the pages themselves but about how they relate to other pages of other classes. This type of attributes / classeme representation has been extensively used in the recent literature (*e.g.* [76, 126, 122, 103]) with very good results. In fact, [122] presented excellent results using images with noisy labels gathered directly from internet queries without human intervention, showing that this representation can still be very useful when using "noisy" labels as we do in this case.

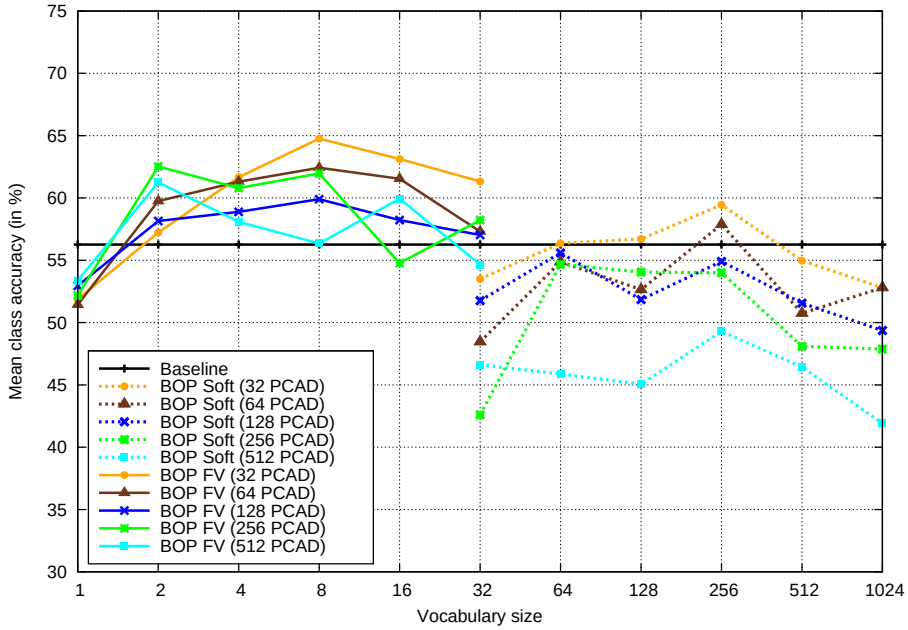
We can understand the classeme framework as a way to cluster the data in a supervised way. For simplicity, let us assume that, when computing the attribute scores, we use a true multiclass classifier that emits probabilities, *i.e.*, given any document q and any class c, $0 \leq s(q,c) \leq 1$, and given $\hat{q} = \{s(q,c_1), s(q,c_2), \ldots, s(q,c_k)\}$, $\sum_{i=1}^{k} \hat{q}_i = 1$. Assuming an almost ideal classifier, then $s(q,c^+) \approx 1$ and $s(q,c^-) \approx 0$, where c^+ is the true class of q and c^- is a different class. In that case, the classifier would be mapping the documents to the vertices of a binary k-dimensional hypercube, where documents of the same class are grouped around the same vertex. Also, intuitively, we can interpret that pages that belong to several types of documents will be "soft assigned" to those classes when calculating the scores, while pages that belong to only one type of document will be put very close of the correct vertex. Therefore, averaging the scores and selecting the class with the largest score can be understood as a majority voting, where every page soft-votes for the class or classes where it is most likely to be seen.

Although in practice we do not have a true multiclass classifier, balancing the number of positive and negative samples as we do achieves a similar goal, making the results comparable. Motivated by this, we explore two ways to improve the BOP approach by clustering the data in a supervised way.

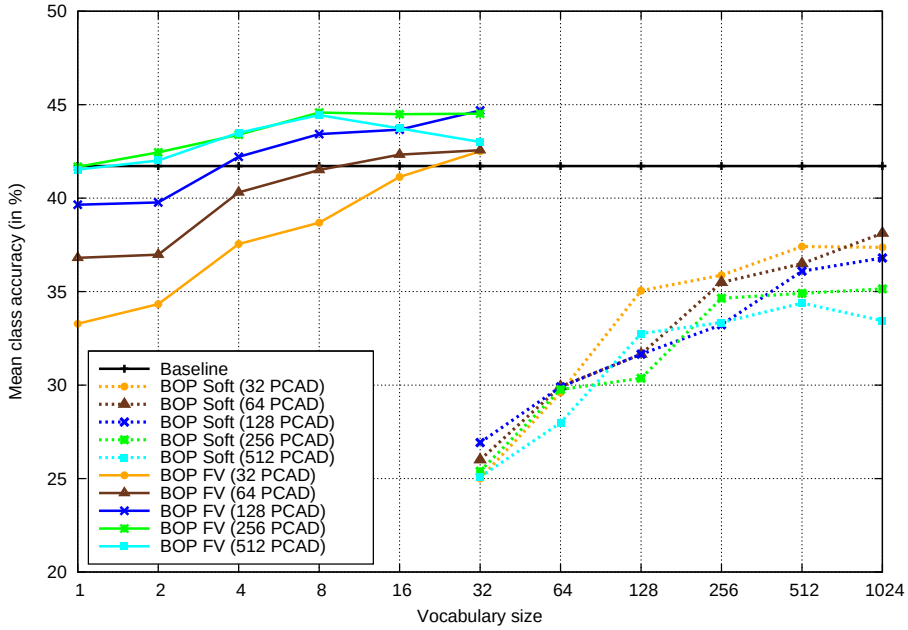
3.5.1 Per-Category Supervised Clustering

One straightforward way to perform supervised clustering consists in clustering the documents of each class independently and then merging the cluster centers. This idea was used, for example, in [42].

In the case of k-means centers, merging the centroids after learning them independently for each class is trivial. Let us consider C classes. If we denote with $\mathcal{V}_c \in \mathbb{R}^{k \times d}$ the k d-dimensional centroids learned for class $c \in C$, then we can construct the final vocabulary



(a) IH2 small



(b) IH2 large

Figure 3.4: IH2. Bag of Pages with soft assignment as a function of the vocabulary size. Top: IH2 small. Bottom: IH2 large.

$\mathcal{V} \in \mathbb{R}^{C*k \times d}$ as:

$$\mathcal{V} = \begin{bmatrix} \mathcal{V}_1 \\ \mathcal{V}_2 \\ \vdots \\ \mathcal{V}_C \end{bmatrix} \tag{3.11}$$

In the case of learning GMMs independently, the merging of the weights, means and variances is analogous to the k-means case, and the only important detail is that the Gaussian weights should be renormalized when merging dividing them by C so they add up to one.

In [42], it was shown that using a per-category supervised clustering can yield large improvements, particularly when combined with soft assignment as we do. One important drawback of this approach, however, is that the number of clusters scales linearly with the number of classes, which is not a feasible option when the number of classes is not trivially low. Finally, it is difficult to aim at a particular vocabulary size without relying on dimensionality reduction techniques. Some works have tried to make this approach more amenable. For example, in [94], images are represented with a combination of an universal and an adapted vocabulary for each class. Each image is not composed of one large histogram, but of many small ones. However, it is still necessary to compute a different vocabulary for each class, which makes the approach impractical if the number of classes is large.

3.5.2 Bag of Page-Classemes

Here we now propose to combine the ideas behind our baseline and our Bag of Pages approach to produce a more discriminant representation which we dubbed Bag of Page-Classemes (BOPC). In a nuthsell, we first compute classeme representations of the document pages as we did in the baseline, and then compute a Bag of Pages representation (either soft or FV) using the classeme signatures instead of the original runlength representations. Note that the runlength representations were still used to compute the classeme representations. This Bag of Page-Classemes offers three very important advantages with respect to the vanilla Bag of Pages:

1. As we observed in section 3.5, the classemes representation can be understood as performing a supervised clustering of the data. Since points now lie close to the vertices of a hypercube, learning a GMM over those points should be a much easier task that learning it over the original runlengths, that lie in a much more complex space.

2. The size of the page signatures is much smaller in the classemes framework, since it is bound to be the number of classes, 6 in the small dataset and 19 in the large. The advantages of this are threefold: i) Accurate clustering of low-dimensional signatures requires less training data than when using high-dimensional signatures. ii) Lower dimensionality signatures lead to a very significantly speedup when computing the GMM vocabulary and when computing the BOPC signatures, and iii) the dimensionality of the page signatures directly affects the size of the final BOPC signature when using the FV.

3. In the Bag of Pages approach, compressing the runlength signatures to the exact right size through PCA before computing the vocabulary was very important, and finding this size was not trivial. However, in this case, we already have small classeme signatures and no PCA is needed[3].

[3]One may note that PCA can still be applied to the runlength signatures *before* learning the

3.6 Supervised Bag of Pages and Bag of Page-Classemes Experiments

For these experiments we will use the same dataset and partitions that we used when evaluating the vanilla Bag of Pages method. To compute the supervised GMM we will use the training set instead of the unsupervised set we used in the vanilla Bag of Pages. The reason is that, since we are using labels to compute the supervised GMM, we would be labeling documents that later may have appeared in the test and validation sets. This was not an extremelly important issue when computing an unsupervised GMM, since no labels were used in that case. Also, using the training set makes the comparison with the Bag of Page-Classemes method fairer, since both use the same data for the learning stage.

To compute the classemes, we will use uncompressed runlengths of 1,512 dimensions. Slight improvements could be obtained by validating the right number of PCA dimensions to use, but the influence is very minor compared to the influence of the PCA in the Bag of Pages approach. We will first validate the parameters of the runlength-to-classeme classifier, choosing the parameters that maximize the score of the baseline method in the validation set. Then, we will compute the classeme signature of the pages and the Bag of Page-Classemes histogram, using both soft assignment and FV. Finally, we will learn the parameters that maximize the document classification score on the validation set. Although learning the parameters of the runlength-to-classeme classifier and the final classifier at the same time would surely produce better results, it would also be much slower to validate. We consider this trade-off between accuracy and speed reasonable.

Figures 3.5 and 3.6 show the results using the supervised GMM clustering and the Bag of Page-Classemes, and compare them to the best vanilla Bag of Pages configurations. In the case of supervised GMM, we report results using only soft assignment, since we did not observe any significant difference between supervised and unsupervised GMM clustering when using the FV encoding. This is consistent with previous works, where using a supervised generative model did not significantly affect the results of the FV [96, 49]. When experimenting with the Bag of Page-Classemes, we report results both on soft assignment and FV.

We can highlight the following points:

- The supervised clustering can slightly improve the results on the IH2 small (Figure 3.5a). More important, we can reach the best results using a much lower number of vocabulary words: the best results are obtained using descriptors of only 24 dimensions. However, these results are still worse than those obtained with an unsupervised FV, as seen on Figure 3.4a.
- A similar behavour can be observed on IH2 large (Figure 3.5b): even if the supervised clustering significantly improves over the unsupervised one when using soft assignment, the best results do not improve over the unsupervised FV results shown on Figure 3.4b.
- The Bag of Page-Classemes approach significantly outperforms the best vanilla Bag of Pages approaches, both on IH2 small (Fig 3.6a) and IH2 large (Fig 3.6b), both when using soft assignment and FV.
- When using soft assignment, the improvements obtained when using the Bag of Page-Classemes 3.6 are larger than those obtained when computing a supervised GMM (Figure 3.5).

classemes. However, the influence of this seems to be minimal compared to the influence of the PCA in the first Bag of Pages approach.

- On the small dataset, using the Bag of Page-Classemes with soft assignment obtains almost the same results than with FV (*cf*. Figure 3.6a). This is surprising, since in general FV clearly outperforms soft assignment in all other scenarios. We believe that the BOPC-FV is still a better descriptor than the BOPC-soft, but we have reached a ceil in the accuracy that we can obtain using only visual descriptors as we do. On the other hand, on the IH2 large dataset, which poses a much more difficult scenario, the BOPC with FV still clearly outperforms the BOPC with soft assignment (*cf*. Figure 3.6b).

- Overall, the Bag of Page-Classemes obtains the best results on both datasets for both soft assignment and FV, significantly improving over the baseline and usually with very small vocabularies. This shows that correctly merging the page scores can have a huge impact in the final results.

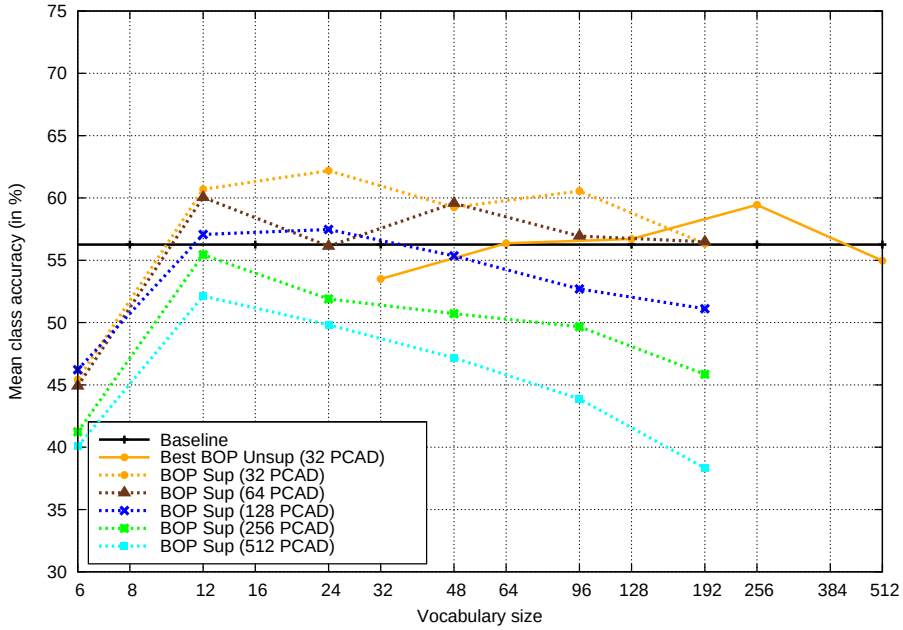
We should also note that, although both approaches are orthogonal and can be applied together, combining a Bag of Page-Classemes representation with the supervised GMM vocabulary did not yield any improvement over computing just the Bag of Page-Classemes.

Finally, it is important to remember that the FV final signature size depends not only on the vocabulary size but also on the dimensionality of the underlying feature vectors. Figure 3.7 shows the results of the Bag of Pages and Bag of Page-Classemes with soft assignment and FV as a function of the final signature size. We can observe how the Bag of Page-Classemes has a clear lead over the Bag of Pages approach at a given signature size, and also how the best results can be obtained using very small signatures: the best results on the small dataset are obtained with a signature of only 24 dimensions. The best results on the large dataset are obtained with a signature of 304 dimensions.

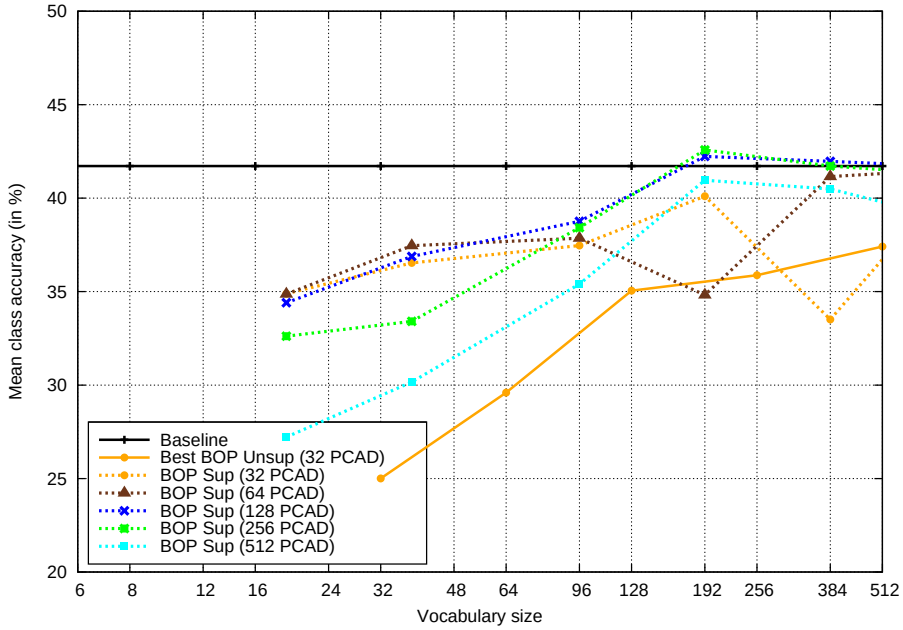
3.7 Retrieval Experiments

For reference purposes, we will also show retrieval results on the IH2 large dataset. We will come back to these results in chapter 5, where we explore the effect of binarization and asymmetric distances in large-scale retrieval. Similarly to what we did in chapter 2, we first sample 100 documents per class of IH2 large to build a balanced subset. We then construct the Bag of Pages (256 PCA dimensions, vocabulary of 8 Gaussians, 4,096 dimensions in total) and Bag of Page-Classemes (8 Gaussians, 304 dimensions in total) signatures, both with FV encoding. Those were the configurations that obtained the best trade-off between accuracy and signature size. We use exactly the same signatures that we already computed in the previous sections. Then, we evaluate following a leave-one-out strategy with an Euclidean distance, and report the Precision at 5 in Table 3.2. Experiments are repeated 5 times with different partitions and the results are averaged. To reduce the dimensionality of the final signatures, we use PCA. Although we could have directly chosen signatures of smaller size, we chose PCA for compatibility reasons, since some of the experiments in chapter 5 will make use of this.

We can observe how, as in the classification task, the Bag of Page-Classemes also outperforms the Bag of Pages when aiming at retrieval tasks, sometimes very noticeably. At 256 dimensions, the improvement is of 6% absolute. Also, in both cases applying a dimensionality reduction step can improve the results. In the Bag of Pages case, reducing the dimensionality to 128 dimensions improves the results of the uncompressed signature around a 5% absolute.

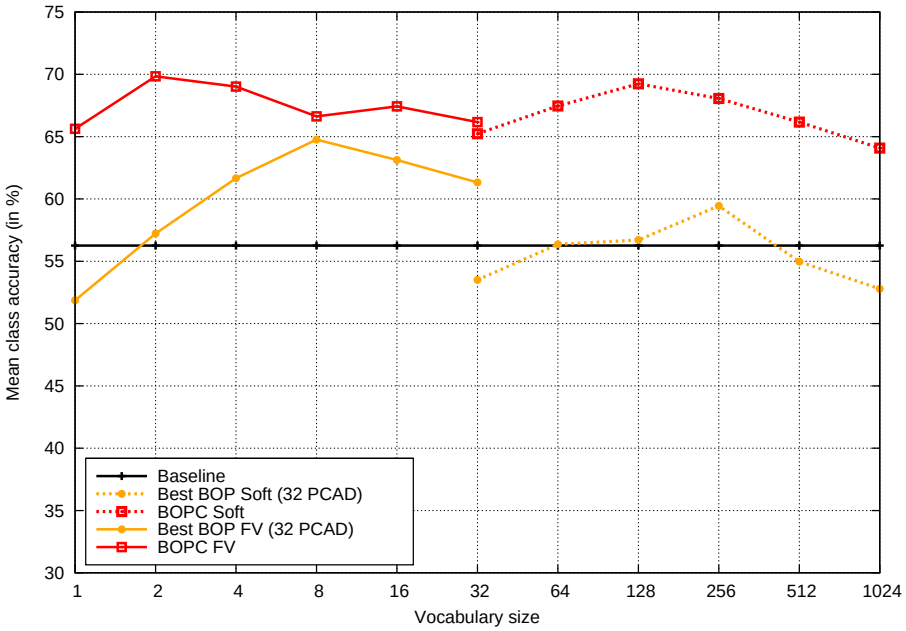


(a) IH2 small

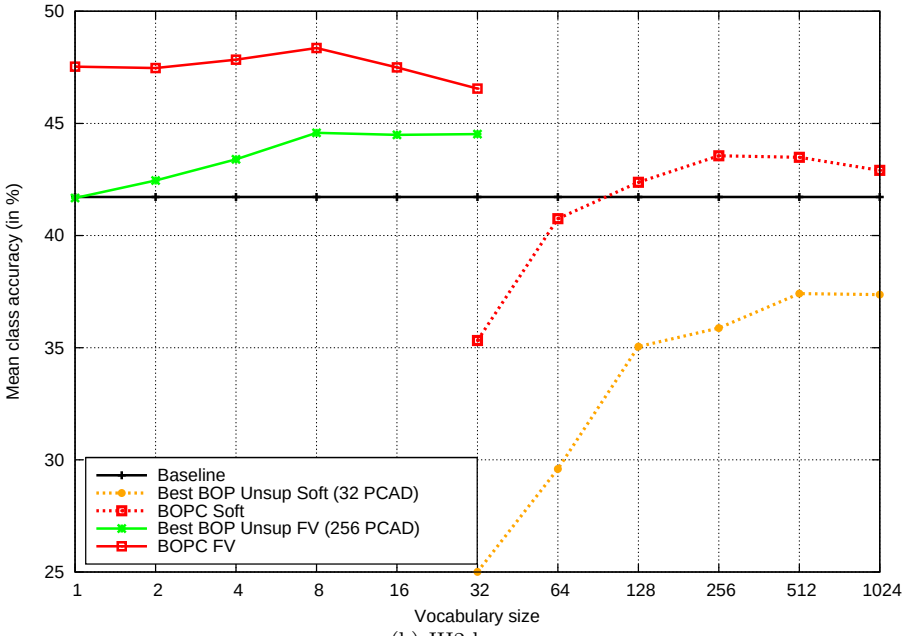


(b) IH2 large

Figure 3.5: IH2. Bag of Pages with soft assignment and supervised clustering as a function of the vocabulary size. Top: IH2 small. Bottom: IH2 large.

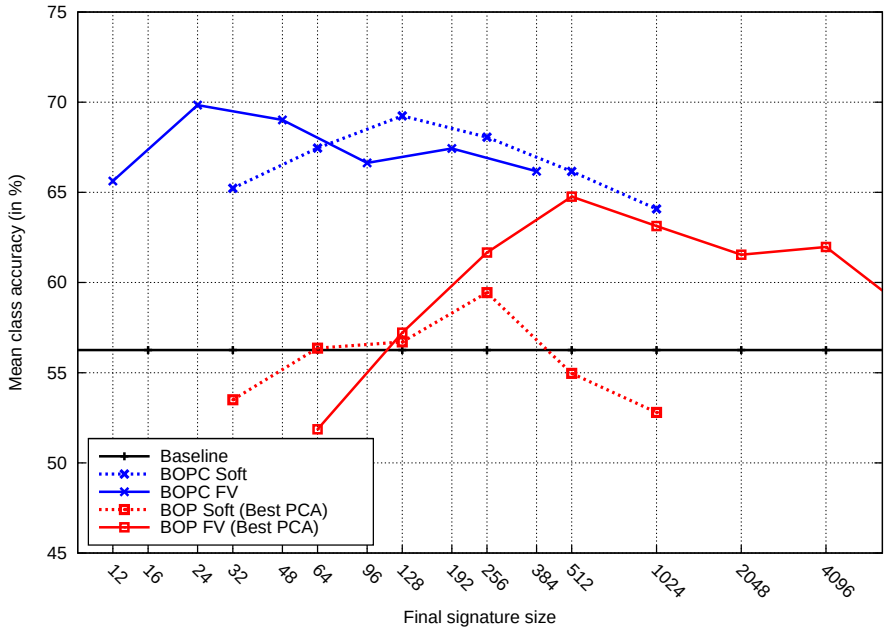


(a) IH2 small

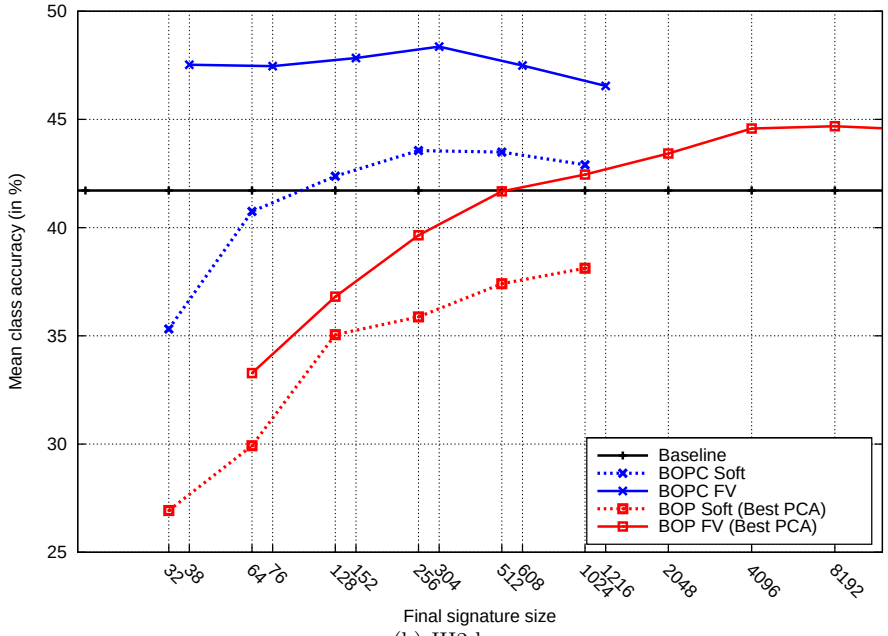


(b) IH2 large

Figure 3.6: IH2. Bag of Page-Classemes with soft assignment as a function of the vocabulary size. Top: IH2 small. Bottom: IH2 large.



(a) IH2 small



(b) IH2 large

Figure 3.7: IH2. Bag of Pages with soft assignment and FV as a function of the final signature size. Top: IH2 small. Bottom: IH2 large

Table 3.2: Precision at 5 (in %) as a function of the number of dimensions on the IH2 large dataset with the Bag of Pages and Bag of Page-Classemes approach.

dimensions	32	64	128	256	304	4,096
Bag of Pages	28.65	29.22	29.15	29.11	-	24.13
Bag of Page-Classemes	32.73	33.61	36.04	35.27	35.28	-

3.8 Bag of Page-Classemes With Textual Features

In the previous sections we have introduced different approaches to multipage representation. In all cases, we have used runlength descriptors to represent the page images, but this is not a requirement, and other features such as text could be used. For IH2 large, a set of textual descriptors is also available. An OCR system was used to extract the noisy text out of the images and construct a bag of words histogram. Common stop words as well as short words and words only appearing once were removed. This produced histograms of approximately 23,700 dimensions for every page that were tf-idf and L2 normalized.

We used these descriptors to compute the baseline results as well as the Bag of Page-Classemes following the same procedure that we used with the visual descriptors: we first find the classifier parameters that maximize the baseline score on the validation set, use those parameters to obtain a classeme representation of the pages, and then compute the Bag of Page-Classeme descriptors. We report results only with the FV encoding, since its superiority over soft assignment on IH2 large was already shown in section 3.6. In Figure 3.8 we can see the results obtained with visual and textual descriptors.

We can notice how the textual descriptors obtain significantly better results than the visual ones, both in the baseline and in the Bag of Page-Classeme approaches. Also interesting is the fact that the improvement of the Bag of Page-Classeme approach over the baseline is more pronounced on the textual than on the visual features. The improvement of the textual baseline with respect to the visual baseline is about 8% absolute, while the improvement of the textual Bag of Page-Classemes with respect to the visual Bag of Page-Classemes is about 12.5%. This suggests that, when using more informative signatures, the Bag of Page-Classemes approach can exploit this information better than the baseline approach.

Finally, Figure 3.8 also shows the results after combining the best visual and textual descriptors. We report results combining the descriptors in two different ways:

1. In the first approach, we concatenate the original runlengths (visual) with the original bag of words (textual) page descriptors, *i.e.*, early fusion of the pages, and use the baseline method with the concatenated descriptors. We can observe how there is only a very slight improvement over the text-only baseline. We believe one of the reasons is that the descriptors are very different in nature, *i.e.*, different types of features, different distributions (dense *vs.* sparse), different influence in the final results, *etc.*, and so simply concatenating them does not bring a large improvement.

2. In the second approach, we first compute the Bag of Page-Classemes signatures of both visual and textual features independently, and then concatenate the best Bag of Page-Classemes descriptors of both views (using 8 Gaussians for the visual view and 4 in the textual view) and learn a classifier in the combined space. In this case, where both textual and visual BOPC signatures have been generated using the same process, there is a very noticeable improvement over using the visual and textual descriptors

independently. This highlights the importance of using comparable representations when merging features.

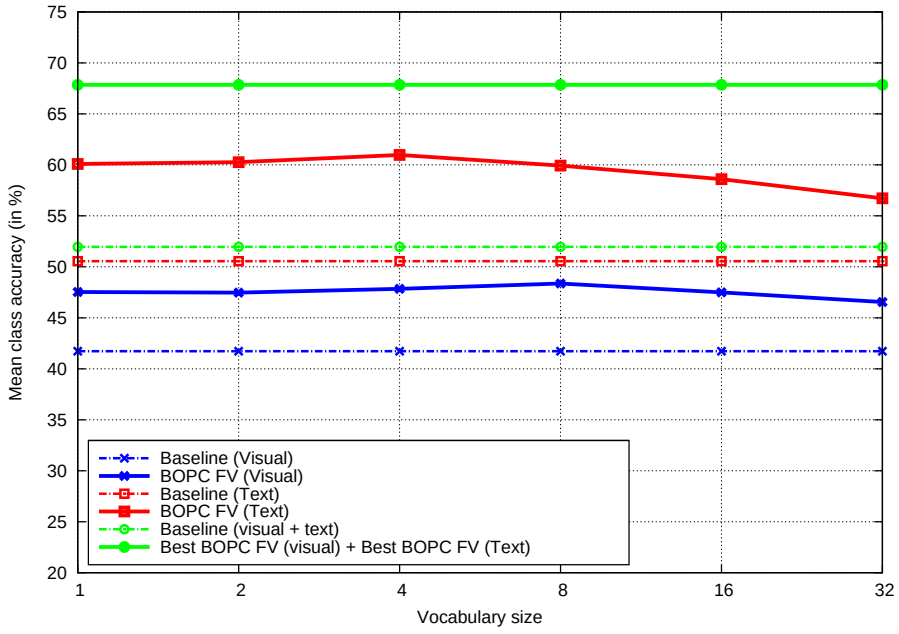


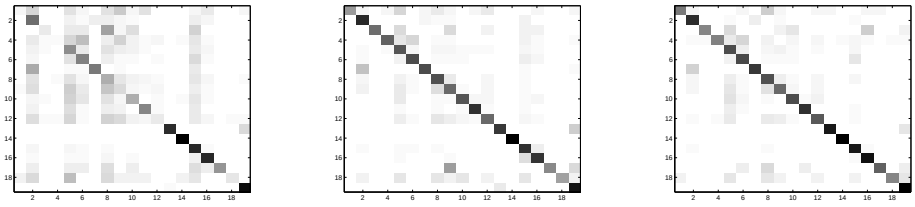
Figure 3.8: IH2 large. Bag of Page-Classemes over visual and textual features.

The improvement of the results when combining the visual and textual features also suggests that the visual and textual features are, to some extent, complementary. This becomes clear by visually inspecting the data: some classes require textual information to be distinguished, while some others barely contain text and cannot be distinguished without visual features. Figure 3.9 shows normalized confusion matrices for the visual, textual, and combined representations. We can observe how many of the cases that were confusing for the visual features can be easily classified using textual features. We can also observe how, in some cases, textual features produce newer confusions, showing that those clases are better classifier with visual features, maybe because of a lack of text. Finally, combining both features reduces the confusions that are not shared between the features.

3.9 Conclusions

Through this chapter we have approached the problem of multiple-page document representation, classification and retrieval. We have proposed different approaches to encode the documents into a single feature vector, and shown how a Bag of Pages approach paired with a supervised representation of the individual pages by means of attributes and a Fisher Vector encoding can obtain results significantly higher than our baseline on two challenging datasets.

Finally, we showed how our representations can be used with other features such as textual bag of words, and not only on visual features. In fact, we observed how textual



(a) (b) (c)

Figure 3.9: Confusion matrices. (a) Bag of Page-Classemes with visual features. (b) Textual features. (c) Visual + textual features.

features actually perform significantly better than visual features. However, we should note that text information is not always available or can be difficult to extract, for example, if documents contain too much noise, are in different languages, or simply contain no text information at all. In fact, for our textual experiments in IH2, we had to discard more than 15,000 pages since they did not contain any valid text. More than 200 documents were completely removed since none of their pages contained text. Text information can also be expensive to obtain, both economically – *e.g.*, an OCR with a per-page license fee – or time – we may require a few seconds to scan a noisy page when aiming at a high quality output. These ideas motivate chapter 4, where we explore how to use good sources of information that will not be available during test time to train more discriminative classifiers.

Chapter 4

Combining Sources of Information[1]

4.1 Introduction

In the previous chapters we overviewed the problem of document representation and we showed some features that are typically used to represent them, such as textual features – as a bag-of-words computed with the output of an OCR [36] –, layout-based features, such as a tree or a graph [84], or visual features, such as features based on texture analysis [28] or the runlength histograms proposed in chapter 2.

In most cases, however, there is no need to limit the number of features we can use at the same time to only one. The combination of multiple features to represent document images or other types of images such as natural images to improve classification results is in fact a common technique widely used in the literature. For example, in [113], the authors combine several low level features such as the average weight, height, or density of connected components at different regions of the image. In [81], visual features are combined with textual features obtained with an OCR application for the task of proper names extraction in fax images. We also combined visual and textual features in chapter 3, showing that it could lead to a very significant boost in accuracy. In [11], layout features are combined with color information for the task of identifying documents captured with low-resolution handheld devices. In a completely different domain, Nilsback and Zisserman show in [89] how to combine colour, shape, and texture vocabularies in a flower classification task. In the following, we will refer to any representation that can be extracted from an image as a view of the image. Many different views can typically be extracted from the same image.

Several strategies exist to combine these views, ranging from concatenating all the views (assuming they all can be represented as feature vectors) and training a classifier in the combined space, or learning independent classifiers and combining the scores, to more advanced frameworks such as Multiple Kernel Learning [77, 116]. In some cases it is possible to design merging strategies that exploit some particular details of the representations. For example, *pormanteau* vocabularies [69] showed very good results when combining bag of words representations of texture and color. Similarly, combining texture information with color-names descriptors instead of other traditional color representations obtained excellent results in object detection in color-rich datasets [68].

What these strategies have in common is that they assume that all the views of an image

[1] Parts of this chapter published in *A. Gordo, F. Perronnin, E. Valveny. Document Classification Using Multiple Views. In Document Analysis Systems, 2012. Best paper award.*

can be obtained both at training and testing stages. However, this is not always possible since some views may be expensive and we cannot afford to obtain them at test time for every image. If we take the example of the textual features, running an OCR on a single page may take up to a few seconds per page when aiming at a high quality output, depending on the quality of the input document, slowing down the whole document distribution workflow. This OCR system could also work with a per-page fee license, and so obtaining a view that relies on OCR will be monetarily expensive. In a completely different scenario, obtaining a view in medical imaging may require an invasive procedure that is discouraged.

We will refer to those views we cannot usually afford at test time as *costly* views, and the ones we can afford at test time as *cheap* views. Fortunately, in some scenarios, we are able to collect, *off-line*, both the cheap and costly views of some images, and this can be exploited to train more discriminative classifiers.

Some works exist about leveraging information not available at test time, sometimes referred to as "coaching". In [120], an expensive view of the data is used to coach the cheap view in a regression and classification problem. The expensive view is first used to partition the space, and a different model is then learned in each of the partitions. Given a new cheap sample, first it is assigned to one of the partitions according to an objective function score, and then the corresponding model is used for the regression or classification task. However, this method does not scale well to high-dimensional spaces, where data usually lies in complex manifolds and the space partition is ineffective.

In the recent [75], the authors explore the use of weakly-paired multimodal data (*i.e.*, the views may have different numbers of items and not be perfectly aligned) in a dimensionality reduction and transfer learning context using Maximum Covariance Analysis (MCA). Image and audio features are available at training time, and the goal is to perform dimensionality reduction of the audio features, which are the only ones available at test time. In [53], Canonical Correlation Analysis (CCA) is used to retrieve images based on text queries, where the image features are no longer available at test time. In [13], a kernelized CCA is also used to improve the clustering of images and text. In all these works, component analysis techniques (MCA, CCA, KCCA, *etc.*) seem to have a very important weight in the success of the methods. This is not unreasonable, since component analysis techniques have shown a huge value in the computer vision field. Of these, CCA and KCCA seem to have a predominant role when multiple views are involved. As we will see during the next section, one of the reasons is that CCA makes use of the many relations of the different views, while other techniques such as MCA only make use of a subset of these relations.

In this chapter, our goal is to exploit the costly views of the documents, that are available for some documents only at training time, to train a more discriminative classifier that deals only with cheap views at testing time. Motivated by the extensive use of CA methods in the literature and particularly CCA, we will use Canonical Correlation Analysis to achieve this purpose. Although CCA has been used in similar scenarios for retrieval [53] and clustering [13] as we just mentioned, we are not aware of it being used in classification tasks.

The main idea behind the method is to find a common subspace between cheap and costly views and a set of projection vectors to embed the views into that subspace. After that, the cheap views can be projected into that common subspace and be used to train a classifier. At test time, we will project the cheap views into the subspace and use the classifier without any need to access the costly view. Since we used the costly views to find this subspace, the projected cheap views are more discriminative than the original ones (Figure 4.1).

The rest of the chapter is organized as follows. Section 4.2 overviews the CCA principles, as well as its limitations and extensions. Section 4.3 shows how to use CCA to train a classifier that works in the cheap views domain but exploits the costly views information available at training time. Section 4.4 deals with the experimental evaluation, and finally section 4.5

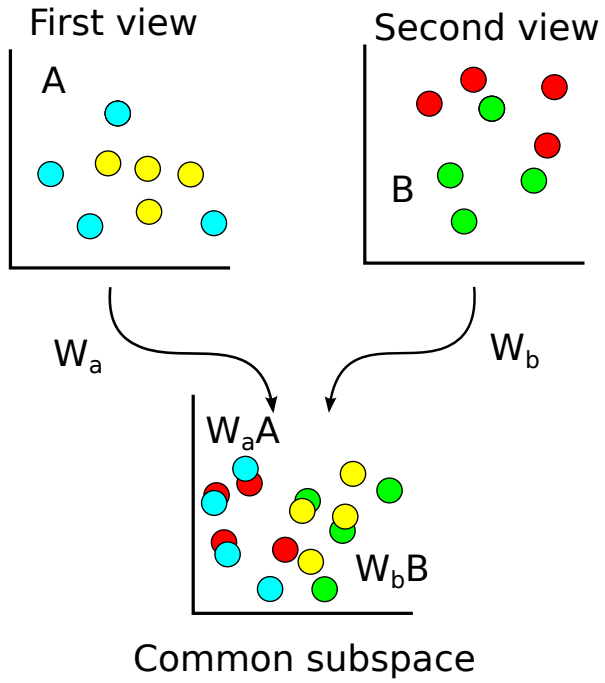


Figure 4.1: Example of two views projected with CCA into a common subspace. The second view is more discriminative, and we expect to induce some of this information into the projected first view.

concludes the chapter.

4.2 Canonical Correlation Analysis

Canonical Correlation Analysis (CCA) is a well-known tool for the analysis of multi-view data, which was first introduced by Harold Hotelling in 1936 [55]. CCA and its variants have been used, for example, in unsupervised tasks such as retrieval [53], clustering [13], supervised dimensionality reduction [46], face recognition [83], and others.

Let us first consider a set of N unlabeled training samples, and let $A \in \mathbb{R}^{d_1 \times N}$ and $B \in \mathbb{R}^{d_2 \times N}$ be two views of the data represented with column feature vectors. The dimensionality of the vectors in A and B may be different. Without loss of generality, we will assume that A is the *cheap* view and that B is the *expensive* view of the samples. Let us also define the matrices $C_{aa} = AA' + \rho I$, $C_{bb} = BB' + \rho I$, $C_{ab} = AB'$, and $C_{ba} = C'_{ab}$, where ρ is a regularization factor used to avoid numerically ill-conditioned situations.

The goal of CCA is to find a projection of each view that maximizes the correlation between the projected representations. This can be expressed as:

$$\underset{w_a, w_b}{\operatorname{argmax}} \frac{w'_a AB' w_b}{\sqrt{w'_a AA' w_a}\sqrt{w'_b BB' w_b}} = \frac{w'_a C_{ab} w_b}{\sqrt{w'_a C_{aa} w_a}\sqrt{w'_b C_{bb} w_b}}, \tag{4.1}$$

which can be rewritten as

$$\underset{w_a, w_b}{\operatorname{argmax}} w'_a C_{ab} w_b, \tag{4.2}$$

subject to the constraints $w_a' C_{aa} w_a = 1$ and $w_b' C_{bb} w_b = 1$.

Here, $w_a \in \mathbb{R}^{d_1}$ and $w_b \in \mathbb{R}^{d_2}$ are the projections that embed the data from A and B into a one-dimensional common subspace where the correlation is maximal. Usually we will be interested in a subspace of k dimensions instead of only one. To do so, we will need to solve Equation (4.1) k times to obtain the projection vectors $W_a = \{w_{a1}, w_{a2}, \ldots, w_{ak}\}$ and $W_b = \{w_{b1}, w_{b2}, \ldots, w_{bk}\}$, subject to them being uncorrelated.

To solve Equation (4.2), we first rewrite it modeling the constraints with Lagrangian multipliers:

$$L(\lambda, w_a, w_b) = w_a' C_{ab} w_b - \frac{\lambda_a}{2}(w_a' C_{aa} w_a - 1) - \frac{\lambda_b}{2}(w_b' C_{bb} w_b - 1). \tag{4.3}$$

Taking derivatives in respect to w_a and w_b we obtain

$$\frac{\partial L}{\partial w_a} = C_{ab} w_b - \lambda_a C_{aa} w_a = 0 \tag{4.4}$$

$$\frac{\partial L}{\partial w_b} = C_{ba} w_a - \lambda_b C_{bb} w_b = 0. \tag{4.5}$$

Combining both equations leads to

$$\lambda_b w_b' C_{bb} w_b - \lambda_a w_a' C_{aa} w_a = 0, \tag{4.6}$$

and because of the constraints of Equation (4.2), $\lambda_b - \lambda_a = 0$. Let then $\lambda = \lambda_a = \lambda_b$. Assuming C_{bb} can be inverted, we have

$$w_b = \frac{C_{bb}^{-1} C_{ba} w_a}{\lambda}. \tag{4.7}$$

Substituting this in Equation (4.4) leads to the following generalized eigenvalue problem:

$$Z w_{ak} = \lambda_k^2 w_{ak}, \tag{4.8}$$

with $Z = C_{aa}^{-1} C_{ab} C_{bb}^{-1} C_{ba}$.

The k leading eigenvectors of Z form the $W_a = \{w_{a1}, w_{a2}, \ldots, w_{ak}\}$ projection vectors that project the cheap view A into the k-dimensional common subspace. Similarly, we can solve for b and arrive to an equation analogous to (4.8) to obtain the $W_b = \{w_{b1}, w_{b2}, \ldots, w_{bk}\}$ projection vectors that project the expensive view B into the k-dimensional common subspace. Note that, since we will only project the cheap view into the common subspace (as the expensive view will not be available at test time), we only need to solve for the a's.

When using CCA projections, it is a common practice to scale the eigenvectors by the eigenvalues, *i.e.*, $\hat{W}_a = \{\lambda_1 w_{a1}, \lambda_2 w_{a2}, \ldots, \lambda_k w_{ak}\}$. However, we observed no major effect in our results when performing this scaling – the results where, in fact, marginally worse. The results we report in section 4.4 use projection vectors that have not been scaled.

It is also possible to reformulate CCA in one single equation as follows:

$$Y^{-1} X \hat{w} = \rho \hat{w}, \tag{4.9}$$

where

$$X = \begin{bmatrix} 0 & C_{ab} \\ C_{ba} & 0 \end{bmatrix}, \quad Y = \begin{bmatrix} C_{aa} & 0 \\ 0 & C_{bb} \end{bmatrix}, \quad \textit{and} \quad \hat{w} = \begin{bmatrix} \mu_a w_a \\ \mu_b w_b \end{bmatrix}, \tag{4.10}$$

and where μ_a and μ_b are scaling factors. This is a typical generalized eigenvalue problem that can be solved with standard techniques. Note that equation (4.9) is very general, and can be used to solve other component analysis problems just by choosing the right X and

Y matrices. For example, to solve a Partial Least Squares (PLS) problem, we would replace Y with the identity matrix I. To solve a PCA problem, we would replace X with C_{aa} and Y with I. In Maximum Covariance Analysis (MCA), we replace X with C_{ab} and Y with I. Other problems can also be solved by choosing the appropriate X and Y matrices. One of the reasons CCA seems to be particularly popular when dealing with multiple-view problems is that, as opposed to other component analysis methods, it makes use of all the relations of the data: C_{aa}, C_{bb}, C_{ab}, and C_{ba}.

4.2.1 Limitations and extensions of the CCA

As presented, CCA suffers from two important limitations. First, it is restricted to only two views, and, second, it is restricted to linear relations in vectorial data. Both limitations can be lifted using common CCA extensions.

To solve CCA using multiple views, we can exploit the generality of equation (4.9) by constructing the appropriate matrices [15, 13]. For example, assuming k views $\{V_1 \dots V_k\}$, and if we let $C_{ij} = V_i V_j'$, then we can construct X and Y as follows:

$$X = \begin{bmatrix} C_{11} & \cdots & C_{1k} \\ \vdots & \ddots & \vdots \\ C_{k1} & \cdots & C_{kk} \end{bmatrix}, \quad Y = \begin{bmatrix} C_{11} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & C_{kk} \end{bmatrix}. \tag{4.11}$$

This can be solved using the same generalized eigenvalue techniques.

When data is not in vectorial form (such as documents represented with a layout graph, or descriptors based on variable-length sequences), or is not linearly separable, a kernelized CCA (KCCA) can be used in the kernel space [53]. This follows a formulation very similar to the original CCA. Note that this requires to represent all new samples as a function of the training samples, which may not always be a viable option if the kernel computation is slow.

One important drawback about CCA is the computational cost, since solving directly equation 4.8 may be costly due to the matrix inversions and the eignevalue problem. Fortunately, some techniques such as gradient descent can be used to ease the task [31]. CCA can also be understood as a particular instance of least-squares weighted kernel reduced rank regression (LS-WKRRR) [30], and can be solved efficiently under this framework.

In some cases, we may find that the different views of the data are not perfectly paired, *i.e.*, the "A view" elements of the training set do not exactly match one-to-one the elements of the "B view". This is not very common when dealing only with images, but is a typical issue when dealing with signals that depend on time such as audio or video. In [75], Lampert and Krömer propose a method to learn a common subspace between views when these are weakly-paired, *i.e.*, groups of elements of the first view are paired with groups of elements of the second view, but the elements do not need to be aligned, or, for that matter, have the same number of elements. To do so, they combine a Maximum Covariance Analysis (MCA) approach with an assignment problem. The goal is to optimize, at the same time, the projections of the data and the assignment of the views:

$$\underset{W_a, W_b, \Pi}{\operatorname{argmax}} \; tr\left[W_a' A \Pi B' W_b\right], \tag{4.12}$$

where $\Pi_{ij} \in \{0, 1\}$ states whether samples A_i and B_j are related, and, as before, W_a and W_b are independent. This problem is solved iteratively: in one iteration, the assignment Π is fixed and the projections are updated solving an SVD problem. In the following iteration, the projections are fixed and the optimal assignment is found, using, for example, the Hungarian method [72].

When the data is perfectly paired, then $\Pi = I$, and the equation becomes

$$\underset{W_a, W_b}{\operatorname{argmax}}\ tr\left[W_a' C_{ab}' W_b\right]. \tag{4.13}$$

or, for just a one-dimensional embedding,

$$\underset{w_a, w_b}{\operatorname{argmax}}\ w_a' C_{ab} w_b, \tag{4.14}$$

which corresponds to the MCA formulation. This is very similar to the CCA formulation, but we are maximizing the covariance instead of the correlation. In fact, it is an unconstrained version of Equation (4.2). To solve for W_a, we simply take the leading eigenvectors of the C_{ab} matrix.

4.3 Learning with CCA

In this section we will first review the process of training and classifying when only two views are available, one cheap and one costly. We will assume that we have access to a set of images where both views U_{cheap} and U_{costly} are available. Note that the labels of such documents are not needed. We also have access to the cheap view of a set of images, S_{cheap}, where the labels l are available.

The training process is explained in Algorithm 1. First, the common subspace between the cheap and costly views is learned using the unsupervised data. This produces a set of projection vectors with which the supervised data S_{cheap} is projected into the common subspace. Then, a classifier (for example an SVM) is learned in this space.

Algorithm 1 Train classifier

Input: $U_{cheap} \in \mathbb{R}^{d_1 \times N}$, cheap view of the unsupervised data,
$U_{costly} \in \mathbb{R}^{d_2 \times N}$, costly view of the unsupervised data,
k, *s.t.* $k \leq min(d_1, d_2)$, dimensionality of the embedded space
$S_{cheap} \in \mathbb{R}^{d_1 \times M}$, cheap view of the supervised data, and
$L \in \mathbb{R}^{M}$, labels of the supervised data.

Output: $a \in \mathbb{R}^{k \times d_1}$, the projection matrix that embeds S into the common subspace of k dimensions, and
W, the trained classifier in the embedded space.

1- Obtain the projections of the cheap view into the common subspace with CCA. The projections of the costly view, b, can be discarded:
$[a, b] = CCA(U_{cheap}, U_{costly}, k)$
2- Project the supervised data into the subspace:
$P = a \cdot S_{cheap}$
3- Train a classifier in the embedded space:
$W = TrainClassifier(P, L)$

The classification process is explained in Algorithm 2. A new unlabeled sample x_{cheap} is first projected into the common subspace using the learned projections, and then classified using the trained classifier.

In the case of non-vectorial representations, CCA can be easily replaced with KCCA in Algorithm 1. This can be trivially extended to more views. For example, assuming we

Algorithm 2 Classify sample

Input: $x_{cheap} \in \mathbb{R}^{d_1}$, cheap view of an unlabeled sample to classify,
$a \in \mathbb{R}^{k \times d_1}$, the projection matrix that embeds s_{cheap} into the common subspace of k dimensions, and
W, the trained classifier in the embedded space.
Output: l, the label of the input document x.

1- Project x_{cheap} into the embedded subspace:
$P = a \cdot x_{cheap}$
2- Classify the sample:
$l = Classify(P, W)$

have m cheap views and n costly views, at *train* time, CCA in Algorithm 1 will return $m + n$ projections, one for each view. We will project only the m cheap views with their corresponding projection and train one independent classifier for each of the projected cheap views, m in total.

At *test* time, we will project each of the m cheap views with their projection and classify each one with their classifier. The final score can be computed, for example, averaging the scores of the m classifiers.

4.4 Experiments

4.4.1 Experimental setup

Unfortunately, we are not aware of any large enough public documents dataset where multiple views could be readily exploited. Therefore, all our experiments have been carried out in an in-house dataset, which we dubbed IH3. This in-house dataset comes from real-world data, and contains approximately 40,000 document images split into 181 categories, mostly different types of invoices and forms. The number of documents in each category varies significantly, from as few as 5 documents in one category to as many as 4,000 in another.

For each document, two views are available. First, the cheap view, a multi-scale runlength histogram of 1,512 dimensions, using the same configuration that we have used in previous chapters. This histogram captures the visual appearance of the page at several positions and scales, providing some basic structural information. Second, the costly view, a bag-of-words histogram of 5,000 dimensions constructed with the text output of an OCR application. The OCR bag-of-words histogram is the costly view both because of the economic costs associated with the licensing of a third party software, and because it takes up to a few seconds per page to obtain the descriptor. The runlength histograms, however, can be computed in a few tens of milliseconds. As the runlength histogram, the text histogram is L_1 normalized and then square-rooted.

As we will see in the experiments, the textual bag-of-words has a significantly better accuracy than the visual features. However, the computational cost and a per-page fee makes its use for all documents inviable at test time.

We divide the documents in 4 different sets:

Unsupervised training set: We select 10,000 random documents to be used as unsupervised training set. Both cheap and costly views are available, but the labels of the

documents are not available. We ensure that at least one document from each class is available in the unsupervised set.

Supervised training set: We vary the size of the supervised training set, from 5 to 20 documents per class. Only the cheap view is available, but we have access to the true labels of the documents.

Validation set: The validation set contains 50% of the remaining documents of each class. We ensure that at least one document of each class is available in the validation set, subtracting it from the train set if necessary.

Test set: The remaining documents comprise the test set. As with the validation set, we ensure that at least one document of each class is available.

We repeat the experiments with 5 different supervised training/validation/test partitions and average the results. The unsupervised train partition remains constant through all the experiments. For computing the CCA projections, we used the code available at [15], where the only parameter is the regularization factor ρ. For the supervised classification, we used a linear SVM trained with SGD as in previous chapters. All the parameters (the regularization ρ of CCA and the number of iterations and regularization λ of the SVM) are validated on the validation set.

We perform two different experiments. The first one is standard classification without rejection, where we report the mean class accuracy, *i.e.*, computing the accuracy of each category independently and then averaging the results. When dealing with very unbalanced categories like in this dataset, the mean class accuracy gives a more meaningful result than the mean document accuracy.

In the second experiment we introduce rejection: we only assign a label to a document if the classification score is higher than a threshold, and otherwise we reject the document. This can ensure a very high classification rate, although it may lead to the rejection of many documents. To represent these results we use accuracy-coverage plots, reminiscent of the precision-recall plots used in retrieval. The main difference is that precision-recall plot shows the precision as a function of the percentage of *relevant* documents retrieved, while the coverage plot shows the precision as a function of the percentage of documents for which we accept the classification result, whether they are correctly classified or not. The coverage plot shows, therefore, what is the expected number of documents that will be accepted or rejected when aiming at a given average classification rate.

4.4.2 Classification without rejection

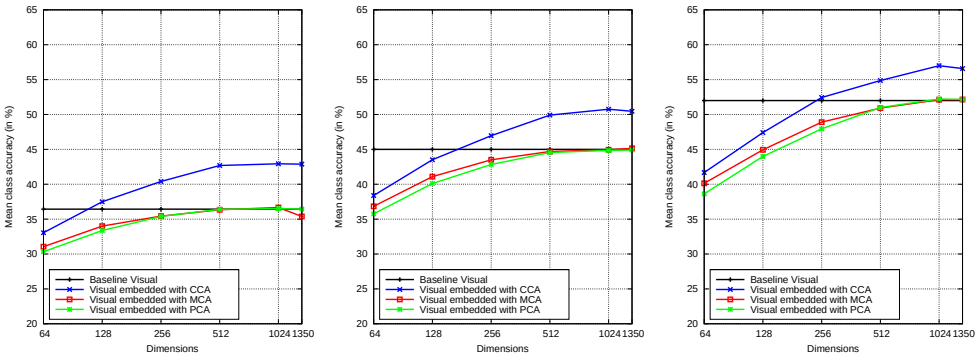
The classification results using 5, 10, and 20 supervised training samples per class can be seen in Figure 4.2. We plot the classification results using only the cheap visual features, as well as the results after embedding the samples into the common subspace with CCA, as a function of the number of dimensions of the subspace. The maximum number of dimensions is limited by the minimum number of independent dimensions in the original views, in this case 1,350. This is due to the fact that the runlength histograms contain a significant amount of zeros, limiting the number of independent dimensions. For comparison purposes, we also report results using MCA. As with CCA, MCA uses both views during training but only the cheap view during testing. Since we are also performing dimensionality reduction, we also plot the classification results obtained using PCA. In this case, the expensive view is not used. Finally, we also report the results using the text features. Note that these are the expensive features, and would not be available at test time. This corresponds to a hypothetical upper bound of the system. We can draw the following conclusions:

Cheap *vs.* costly baselines: As expected, the costly features perform significantly better than the cheap features. This is essentially by design: a costly feature that performs

worse than a cheap one would probably not be considered in the first place.

Visual *vs.* visual after CCA embedding: We can observe how, in all three settings, using CCA can noticeably improve the baseline results when using the maximum number of dimensions available in the subspace. When performing dimensionality reduction, we can significantly reduce the dimensionality while still obtaining better results than the baseline. This is particularly true in the case when few supervised samples are available: with 5 samples per class, we can reduce the descriptors down to 128 dimensions and still obtain better results than the baseline. However, when using 20 documents per class for training, we can only reduce down to 256 dimensions. This suggests that this CCA embedding is particularly suited in the case where little supervised data is available.

CCA *vs.* MCA *vs.* PCA embeddings: In all three settings, using CCA for dimensionality reduction systematically produces better results than using MCA and PCA. The differences are small when reducing to a very low number of dimensions, but increasing the number of dimensions also increases the differences between CCA and the other methods. This is not unexpected, since one of the main uses of CCA is precisely to perform a dimensionality reduction. Note how, as in the previous case, the differences also become smaller when using a larger number of supervised documents for the learning stage, supporting the idea that CCA is particularly suited when few supervised samples are available. MCA very slightly improves over PCA when using very few dimensions, but converges when the number of dimensions increase. The differences between CCA and MCA are usually quite large. This is reasonable, since CCA uses more complex information that MCA: MCA only uses information about the cross-covariance of the views, while CCA also uses information about the covariance of the independent views to normalize the projections.



(a) 5 training samples per class. (b) 10 training samples per class. (c) 20 training samples per class.

Figure 4.2: Classification results using 5 (a), 10 (b), and 20 (c) training samples per class. The text baselines (which cannot be computed in practice) are 74.02% (a), 79.16% (b), and 81.71% (c).

4.4.3 Classification with rejection

In practice, the classification results that we have observed are not useful in a real production environment. Usually, rejection is integrated in the system: if the score of a sample does not reach a given threshold, the sample will not be classified in this stage, and will be sent to a different pipeline, which will probably use more expensive features or human intervention. Typically, to guarantee a high accuracy, we set a high threshold, but then most of the

documents will be rejected, leading to a low coverage. On the other hand, setting a low score will yield a high coverage, but the classification rate will drop.

Figure 4.3 shows accuracy-coverage plots for the visual baseline as well as text baseline and the visual features embedded in a 1,024 dimensional subspace with CCA, using 5 and 20 training samples per class. We do not report results with MCA since it did not improve over the baseline when using a large number of dimensions. We can observe that:

i) Given an accuracy threshold, the coverage with CCA is significantly larger than the visual baseline. Aiming at a 90% classification accuracy and using 10 training samples per class, we can cover approximately 12% of the dataset when using CCA, while using only the visual baseline we cannot reach a 90% accuracy. When using 20 training samples per class, we can cover approximately 12% more of the dataset (from 4% to 16%). Since all the rejected documents will be sent to the more expensive pipeline with textual features, these quantities can be directly translated into savings.

ii) When using 10 or 20 samples, the visual CCA and the textual baseline perform very similarly when the goal is to obtain a very high precision accuracy. This is particularly relevant because it shows that, depending on the objectives, the expensive view can be replaced with the embedded cheap view without significant loss.

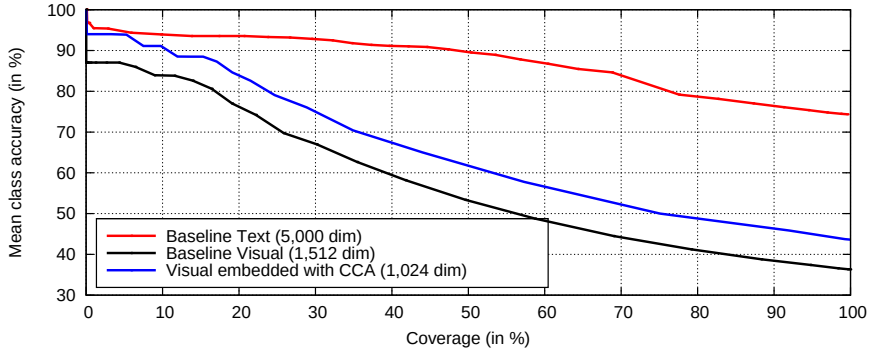
4.4.4 Retrieval

Finally, we report retrieval results. As in previous chapters, we sample 50 documents of each of the 181 classes to have a balanced dataset and perform retrieval in a leaving-one-out strategy. We will report results using the CCA embeddings and compare it to the PCA baseline. We will not include MCA since the results were only slightly better than PCA and significantly worse than CCA. In the case of CCA, we will first project the original signatures with CCA to obtain descriptors of 1,350 dimensions. These descriptors are then L2 normalized, since we observed this to make a very significant difference in the case of CCA. Then, similarly to what we did on chapter 3, we will apply PCA to those 1,350 dimensional descriptors to further reduce their dimensionality. This is for compatibility reasons with some experiments that will be carried on on chapter 5. We would like to note that applying only CCA instead of CCA+PCA yielded only very minor differences in the results, and not always favourable to CCA. As before, we will report Precision at 5.

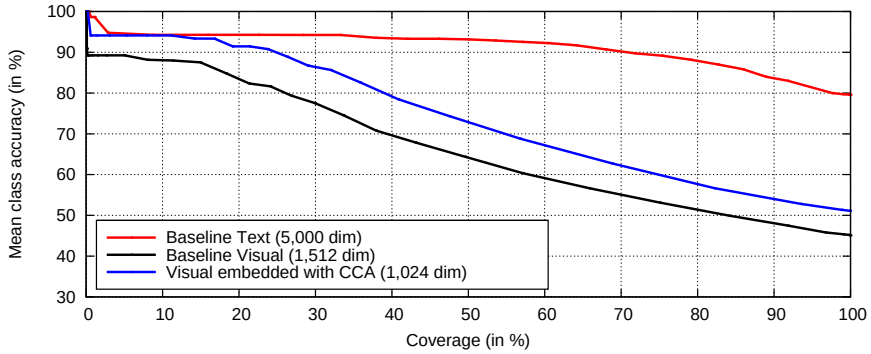
We can observe the results in Table 4.1. It is clear how the CCA embedding is also useful in retrieval tasks. In fact, the differences between the baseline and the CCA embedding are larger in this retrieval task than they were in classification. This is in line with our findings in the previous section, where the differences between CCA and PCA decreased as we used more training data. Here, in the extreme case where we have no supervised data, the differences between them become largest.

Table 4.1: Precision at 5 (in %) as a function of the dimensionality on the IH3 dataset.

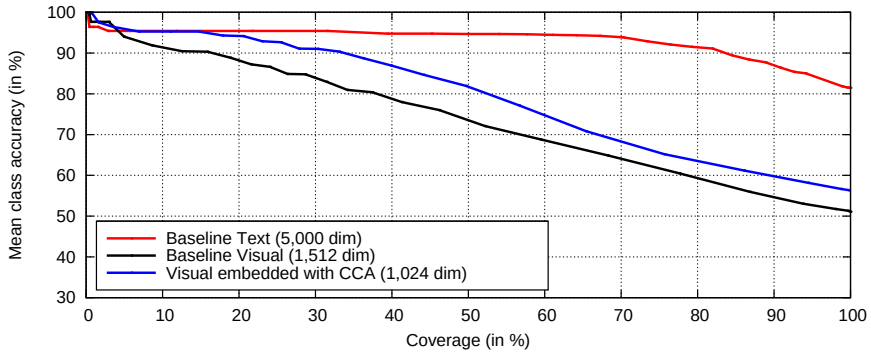
dimensions	64	128	256	512	1,024	1,350	1,512
Visual embedded with PCA	35.62	36.44	36.87	36.98	36.99	-	36.99
Visual embedded with CCA+PCA	43.94	45.70	45.48	45.77	47.04	47.00	-



(a) 5 training samples per class.



(b) 10 training samples per class.



(c) 20 training samples per class.

Figure 4.3: Accuracy-coverage plot using 5 (a), 10 (b), and 20 (c) training samples per class.

4.5 Conclusions

In this chapter we have shown how Canonical Correlation Analysis (CCA) can be used to improve the accuracy in a document classification task where some views of the data are only available at the training stage, since they are too expensive to be obtained at test time. We have shown how, in a system with rejection, CCA can significantly increase the coverage over

the baseline when aiming at the same classification accuracy. Finally, we have also shown how CCA is particularly useful when supervised data is scarce, and how the improvement becomes largest in the case where we have no supervised data at all.

Part II

Large-Scale Retrieval

Through the previous chapters we have dealt with the problem of document representation aimed at classification and retrieval tasks, and focusing on the digital mailroom scenario. We have first focused on the problem of single-page document representation, and then we have proposed some approaches to encode more complex documents, such as multiple-page documents, or documents containing more than one source of information. We have showed the *generality* and the *soundness* of the approaches, obtaining state-of-the-art results on public and in-house datasets of very different nature and quality. However, we have not made many claims about our third requirement: the *scalability* of the methods. Although we have made some design choices to avoid approaches that would not scale well – such as avoiding layout-based representations –, we have not actively researched the scalability of the methods.

Through the next chapters we will focus precisely on the scalability issues. For this task, we will go beyond document representation and study what are the main problems of dealing with hundreds of thousands or millions of images, as well as proposing some effective solutions. One of the principal problems is the need to maintain all the document signatures in RAM memory at the same time for retrieval to be efficient. One popular approach consists in aggressively compress the signatures down to just a few hundreds of bits per image through binarization. In chapter 5 we propose two generic asymmetric distances that can significantly improve the accuracy of many popular embedding methods with a minimum extra cost at query time, as well as a very simple binarization method which we dubbed PCAE. When combined with the asymmetric distances, PCAE can obtain results comparable to the state-of-the-art but with a conceptually much simpler process. A second issue is the importance of performing some supervised learning when performing large-scale retrieval. As noted in chapter 2, using labeled samples to learn how to rank could yield very significant improvements. These improvements are even more important in large-scale, more difficult problems. In chapter 6, we study several methods to exploit labeled information to improve the retrieval performance with no overhead at query time.

We would like to note that most of the experiments through the next chapters will be performed on natural images instead of document images. This is mostly due the scarcity of publicly available large document datasets, as opposed to the many public natural image datasets such as ImageNet, LabelMe, TinyImages, *etc.* Nonetheless, in some cases we will also perform experiments on the document datasets that we have introduced through the previous chapters. The techniques and methods presented in these chapters are quite general, and can be applied directly to document retrieval problems if the data is available.

Chapter 5

Asymmetric Distances for Binary Embeddings[1]

5.1 Introduction

One typical task to be performed in the digital mailroom consists in retrieving documents from potentially large databases, which can be useful for several tasks. One obvious application is to perform a nearest neighbor classification of the documents. Another practical application is to transfer information from the already analyzed documents from the dataset into the new incoming document. Lately, these document databases have been experiencing a large increase in volume, which renders traditional retrieval methods infeasible.

These changes in the amount of data that needs to be handled is not exclusive to the document domains. Recently, the computer vision community has witnessed an explosion in the scale of the datasets it has had to handle. While standard image benchmarks such as PASCAL VOC [41] or CalTech 101 [43] used to contain only a few thousand images, resources such as ImageNet [34] (14 million images) and Tiny images [121] (80 million images) are now available. In parallel, more and more sophisticated image descriptors have been proposed including the GIST [91], the bag-of-visual-words (BOV) histogram [115, 27], the Fisher Vector (FV) [96, 99] or the Vector of Locally Aggregated Descriptors (VLAD) [64]. Descriptors with thousands or tens of thousands of dimensions have become the norm rather than the exception. Consequently, handling these gigantic quantities of data has become a challenge on its own.

When dealing with large amounts of data, there are two considerations of paramount importance. The first one is the *computational cost*: the computation of the distance between two image signatures should rely on efficient operations. The second one is the *memory cost*: the memory footprint of the objects should be small enough so that all database image signatures fit in RAM. If this is not the case, *i.e.* if a significant portion of the database signatures has to be stored on disk, then the response time of a query collapses because the disk access is much slower than that of RAM access.

These considerations have directly motivated research in learning compact binary codes [56, 20, 100, 130, 73, 64, 17, 127, 122, 12, 46]. A desirable property of such coding schemes is that they should map similar data points (with respect to a given metric such as the

[1] Parts of this chapter published in *A. Gordo and F. Perronnin. Asymmetric Distances for Binary Embeddings. In CVPR, 2011.* This chapter submitted to *IEEE TPAMI.*

Euclidean distance) to similar binary vectors (*i.e.* vectors with a small Hamming distance). Transforming high-dimensional real-valued image descriptors into compact binary codes directly addresses both memory and computational problems. First the compression enables to store a large number of codes in RAM. Second, the Hamming distance is extremely efficient to compute in hardware, which enables the exhaustive computation of millions of distances per second, even on a single CPU.

However, it has been noted that compressing the query signature is not mandatory [37, 63, 17]. Indeed, the additional cost of storing in memory a single non-binarized signature is negligible. Also, the distance between an original signature and a compressed signature can still be computed efficiently through look-up table operations. As the distance is computed between two different spaces, these algorithms are referred to as *asymmetric*. A major benefit of asymmetric algorithms is that they can achieve higher accuracy for a fixed compression rate because they take advantage of the more precise position information of the query. We note however that the asymmetric algorithms presented in [37, 63, 17] are tied to specific compression schemes. Dong *et al.* [37] presented an asymmetric algorithm for compression schemes based on random projections. Jégou *et al.* [63] proposed an asymmetric algorithm for compression schemes based on vector quantization. Brandt [17] subsequently used a similar idea.

In this chapter, we generalize these asymmetric schemes to a broader family of binary embeddings. We first provide an overview of several binary embedding algorithms (including Locality Sensitive Hashing (LSH) [56, 20], Locality-Sensitive Binary Codes (LSBC) [100], Spectral Hashing (SH) [130], PCA Embedding (PCAE) [50, 46], PCAE with random rotations (PCAE-RR), and PCAE with iterative quantization (PCAE-ITQ) [46]) showing that they can be decomposed into two steps: i) the signatures are first embedded in an intermediate real-valued space and ii) thresholding is performed in this space to obtain binary outputs. A key insight that our asymmetric distances will exploit is that the Euclidean distance is a natural metric in the intermediate real-valued space.

Building on the previous analysis we propose two asymmetric distances which can be broadly applied to binary embedding algorithms. The first one is an expectation-based technique inspired by [63]. The second one is a lower-bound-based technique which generalizes [37].

We show experimentally on four datasets of different nature that the proposed asymmetric distances consistently and significantly improve the retrieval accuracy of LSH, LSBC, SH, PCAE, PCAE-RR, and PCAE-ITQ over the symmetric Hamming distance. Although the lower-bound and expectation-based techniques are very different in nature, they are shown to yield very similar improvements.

The remainder of this chapter is organized as follows. In the next section, we provide an analysis of several binary embedding techniques. In section 5.3 we build on the previous analysis to propose two asymmetric distance computation algorithms for binary embeddings. In section 5.4 we provide experimental results on natural images, while in section 5.5 we provide experimental results using the document datasets that we introduced in the previous part of this thesis. Finally in section 5.6 we discuss conclusions.

5.2 Theoretical Analysis of Binary Embeddings

We now provide a review of several successful binary embedding techniques: LSH, LSBC, SH, PCAE, PCAE-RR, and PCAE-ITQ.

Let us introduce a set of notations. Let x be an image signature in a space Ω and let h_k be a binary embedding function, *i.e.* $h_k : \Omega \rightarrow \{0, 1\}$ (some authors prefer the convention

$h_k : \Omega \to \{-1, +1\}$). A set $\mathcal{H} = \{h_k, k = 1 \dots K\}$ of K functions defines a multi-dimensional embedding function $h : \Omega \to \{0, 1\}^K$ with $h(x) = [h_1(x), \dots, h_K(x)]'$ (and the apostrophe denotes the transpose).

We show that for LSH, LSBC, SH, PCAE, PCAE-RR, and PCAE-ITQ, the functions h_k can be decomposed as follows:

$$h_k(x) = q_k[g_k(x)], \tag{5.1}$$

where $g_k : \Omega \to \mathbb{R}$ is the real-valued embedding function, and $q_k : \mathbb{R} \to \{0, 1\}$ is the binarization function. We denote $g : \Omega \to \mathbb{R}^K$ with $g(x) = [g_1(x), \dots, g_K(x)]'$. If we have two image signatures x and y, we also show that the Euclidean distance is the natural metric between $g(x)$ and $g(y)$ and that it approximates the original distance between x and y. Thus, we can write the squared Euclidean distance as $d(x, y) \approx d(g(x), g(y)) = \sum_k d(g_k(x), g_k(y))$.

In the rest of this section, we survey a number of binary embeddings, which can be classified into two types: those based on random projections (LSH and LSBC), and those based on learning the hashing functions (PCAE, PCAE-RR, PCAE-ITQ, or SH).

5.2.1 Hashing with Random Projections

Locality Sensitive Hashing (LSH)

In LSH, the functions h_k are called hash functions and are selected to approximate a similarity function sim in the original space $\Omega \in \mathbb{R}^D$. Valid hash functions h_k must satisfy the LSH property:

$$Pr\left[h_k(x) = h_k(y)\right] = sim(x, y). \tag{5.2}$$

Here we focus on the case where sim is the cosine similarity $sim(x, y) = 1 - \frac{\theta(x,y)}{\pi}$, for which a suitable hash function is [20]:

$$h_k(x) = \sigma\left(r_k' x\right), \tag{5.3}$$

with

$$\sigma(u) = \begin{cases} 0 & \text{if } u < 0, \\ 1 & \text{if } u \geq 0. \end{cases} \tag{5.4}$$

The vectors $r_k \in \mathbb{R}^D$ are drawn from a multi-dimensional Gaussian distribution p with zero mean and identity covariance matrix I_D. We therefore have $q_k(u) = \sigma(u)$ and $g_k(x) = r_k' x$. In such a case the natural distance between $g(x)$ and $g(y)$ in the intermediate space is the Euclidean distance as random Gaussian projections preserve the Euclidean distance in expectation. This property stems from the following equality:

$$\mathbb{E}_{r \sim p}\left[||r'x - r'y||^2\right] = ||x - y||^2. \tag{5.5}$$

We note that centering the data around the origin can impact LSH very positively, especially when dealing with non-negative data such as GIST vectors [91]. This is because, given a set of points $\{x_i, i = 1 \dots N\}$ with zero-mean and a random direction r_k, we have the guarantee that $\frac{1}{N}\sum_{i=1}^{N} r_k' x_i = 0$, *i.e.* the distribution of values $r_k' x_i$ is centered around the LSH binarization threshold. If the data is not centered around the origin, the mean of the $r_k' x_i$ values can be very different from zero. We have observed cases where the projections all had the same sign, leading to weakly discriminative embedding functions [2].

The previous analysis can be readily extended to the Kernelized LSH approach of Kulis and Grauman [73] as the Mercer kernel between two objects is just a dot-product in another space using a non-linear mapping $\phi(x)$ of the input vectors. This enables one to extend LSH as well as the asymmetric distance computations beyond vectorial representations.

[2] An alternative would be to choose a per-dimension threshold equal to the median of the $r_k' x_i$ values. However, this would not guarantee anymore the convergence to the cosine.

Locality-Sensitive Binary Codes (LSBC)

Consider a Mercer kernel $K : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}$ that satisfies the following properties for all points x and y:

1. It is translation invariant: $K(x, y) = K(x - y)$.
2. It is normalized: *i.e.*, $K(x - y) \leq 1$ and $K(0) = 1$.
3. $\forall \alpha \in \mathbb{R} : \alpha \geq 1$, $K(\alpha x - \alpha y) \leq K(x, y)$.

Two well known examples are the Gaussian kernel and the Laplacian kernel. Raginsky and Lazebnik [100] showed that for such kernels, $K(x, y)$ can be approximated by $1 - 2\text{Ha}(h(x), h(y))/n$ with $h_k(x) = q_k(g_k(x))$ where:

$$g_k(x) = \cos(r_k' x + b_k), \tag{5.6}$$

$$q_k(u) = \sigma(u - t_k). \tag{5.7}$$

b_k and t_k are random values drawn respectively from unif$[0, 2\pi]$ and unif$[-1, +1]$, and n is the number of bits. The vectors r_k are drawn from a distribution p_K which depends on the particular choice of the kernel K. For instance, if K is the Gaussian kernel with bandwidth γ (note that the bandwidth is the inverse of the variance), then p_K is a Gaussian distribution with mean zero and covariance matrix γI_D. As the size of the binary embedding space increases, $1 - 2\text{Ha}(h(x), h(y))/n$ is guaranteed to converge to a function closely related to $K(x, y)$.

We know from Rahimi and Recht [101] that $g(x)'g(y)$ is guaranteed to converge to $K(x, y)$ since $\mathbb{E}_{w_k \sim p_K} g_k(x) g_k(y) = K(x, y)$ and therefore that the embedding g preserves the dot-product in expectation. Since $K(x, x) = K(x - x) = K(0) = 1$, the norm $||g(x)||^2$ is also guaranteed to converge to 1, $\forall x$. In that case, $||g(x) - g(y)||^2 = ||g(x)||^2 + ||g(y)||^2 - 2g(x)'g(y) = 2(1 - g(x)'g(y))$. Therefore the Euclidean distance is equivalent to the dot-product and the Euclidean distance is preserved in the intermediate space in expectation.

5.2.2 Learning Hashing Functions

Hashing methods based on random projections such as LSH and LSBC have important properties, such as the guarantee to converge to the target kernel when the number of bits grows to infinity. However, a large number of bits may be necessary to obtain a sufficiently good approximation. When aiming at short codes, it may be more fruitful to learn the hashing functions rather than to resort to randomness.

We will focus on unsupervised code learning techniques, and especially on those based on PCA, since PCA seems to be a core component of the best-performing binary embedding methods: in Product Quantization [63] and Transform Coding [17], PCA is used as a preliminary step before binarizing the data. In [50] and [46], a direct PCA embedding is used. Spectral Hashing [130] can be understood as a way to assign more bits to the PCA dimensions with more energy.

In the following, let $\mathcal{S} = \{x_i, i = 1 \dots N\}$, be a set of N signatures in $\Omega \in \mathbb{R}^D$ that are available for training purposes.

PCA Embedding (PCAE)

A very simple encoding technique is PCA embedding (PCAE) [46, 50]. We can define PCAE as $h_k(x) = q_k(g_k(x))$, with

$$g_k(x) = w_k'(x - \mu), \tag{5.8}$$

$$q_k(u) = \sigma(u), \tag{5.9}$$

where μ is the mean of the signatures of $\mathcal{S}$, $\mu = \frac{1}{N}\sum_{i=1}^{N} x_i$, and where w_k is the eigenvector associated with the k-th largest eigenvalue of the covariance matrix of the signatures of $\mathcal{S}$. Despite its simplicity, PCAE can obtain very competitive results as will be shown in the experiments of section 6.7.

According to [130], when producing binary codes, two desirable properties are: i) that the bits are pairwise uncorrelated, and ii) that the variance of each bit is maximized. As seen in [127], this leads to an NP hard problem that needs to be relaxed to be solved efficiently. The analysis of [127] also shows that projecting the data with PCA is an optimal solution of the relaxed version of this problem, conferring some theoretical soundness to PCAE.

In the case of PCAE, the Euclidean distance is the natural distance in the intermediate space, since PCA projections preserve, approximately, the Euclidean distance (the PCA directions are those that minimize the mean squared reconstruction error).

A possible drawback of using PCA projections for binarization purposes is that not all the dimensions contain the same energy after the projection. Since, after thresholding, all bits have the same weight, it can be important to balance the variance before quantizing the vectors.

One possible solution is to rotate the projected data. This rotation can be random, such as in PCAE-RR, or learned, such as in PCAE-ITQ. Another option is to assign more bits to the more relevant dimensions. In practice, Spectral Hashing can be seen as a way to achieve this goal, even though its theoretical foundations are different.

PCA Embedding + Random Rotations (PCAE-RR)

As noted in [63, 46], one simple way to balance the variances is to project the data with the PCA projections and then rotate the result with a random orthogonal matrix $R \in \mathbb{R}^{K\times K}$ (or, equivalently, if we put the column eigenvectors in a matrix $W \in \mathbb{R}^{D\times K}$, to project and rotate the data at the same time with a matrix $\tilde{W} = WR$). One way to generate this random orthogonal matrix is to first create a random matrix drawn from a $\mathcal{N}(0,1)$ distribution and perform a QR decomposition, as done in [60]. Another option is to perform an SVD decomposition of such matrix, as done in [46]. We follow the latter approach.

Therefore, the PCAE-RR embedding can be defined as $h_k(x) = q_k(g_k(x))$, with

$$g_k(x) = \tilde{w}_k'(x-\mu), \tag{5.10}$$

$$q_k(u) = \sigma(u), \tag{5.11}$$

where $\tilde{w}_k$ is the kth column of $\tilde{W}$. Note that rotating the data with an orthogonal matrix after the PCA projection is still an optimal solution of the formulation of [127]. Also, since orthogonal rotations preserve the Euclidean distance – already approximately preserved after the PCA rotation –, the natural distance in the intermediate space for PCAE-RR is also the Euclidean distance.

PCA Embedding + Iterative Quantization (PCAE-ITQ)

In [46], the idea of rotating the projections to balance the variances after PCA is taken a step further. The goal is to find the optimal orthogonal rotation R that minimizes the quantization loss in a training set:

$$\underset{R}{\operatorname{argmin}} \sum_{x\in\mathcal{S}} ||q(g(x)) - g(x)||^2, \tag{5.12}$$

with

$$g_k(x) = \tilde{w}_k'(x - \mu), \tag{5.13}$$

$$q_k(u) = 2\sigma(u) - 1, \tag{5.14}$$

and, as before, $\tilde{w}_k = (WR)_k$.

Intuitively, the goal is to map the points into the vertices of a binary hypercube. The closer the points are to the vertices, the smaller the quantization error will be. This optimization problem is related to the Orthogonal Procrustes problem [110], in which one tries to find an orthogonal rotation to align one set of points with another. The optimization can be solved iteratively, and involves computing an SVD decomposition of a $K \times K$ matrix at every iteration. A random orthogonal matrix is used as initial values of this matrix. Since we are usually interested in compact codes (*e.g.*, $K \leq 512$), obtaining the orthogonal rotation matrix R is quite fast. Note also that this optimization has to be computed only once, offline.

As in the case of PCAE-RR, we perform an orthogonal rotation after the PCA projection, and so the Euclidean distance is approximately preserved in the intermediate space.

Spectral Hashing (SH)

Given a similarity *sim* between objects in $\Omega \in \mathbb{R}^D$, and assuming that the distribution of objects in Ω may be described by a probability density function p, SH [130] attempts to minimize the following objective function with respect to h:

$$\int_{x,y} ||h(x) - h(y)||^2 sim(x, y)p(x)p(y)dxdy, \tag{5.15}$$

subject to the following constraints:

$$h(x) \in \{-1, 1\}^K, \tag{5.16}$$

$$\int_x h(x)p(x)dx = 0, \tag{5.17}$$

$$\int_x h(x)h(x)'p(x)dx = I_D. \tag{5.18}$$

As optimizing (5.15) under constraint (5.16) is a NP hard problem (even in the case where $K = 1$, *i.e.* of a single bit), Weiss *et al.* propose to optimize a *relaxed* version of their problem, *i.e.* to remove constraint (5.16), and then to binarize the real-valued output at 0. This is equivalent to minimizing:

$$\int_{x,y} ||g(x) - g(y)||^2 sim(x, y)p(x)p(y)dxdy, \tag{5.19}$$

with respect to g and then writing $h(x) = 2\sigma(g(x)) - 1$, *i.e.* $q(u) = 2\sigma(u) - 1$. The solutions to the relaxed problem are eigenfunctions of the weighted Laplace-Beltrami operators for which there exists a closed-form formula in certain cases, *e.g.* when *sim* is the Gaussian kernel and p is separable and uniform. To satisfy, at least approximately, the separability condition a PCA is first performed on the input vectors.

Minimizing the objective function (5.19) enforces the Euclidean distance between $g(x)$ and $g(y)$ to be (inversely) correlated with the Gaussian kernel for points whose kernel value is high (equivalently, whose Euclidean distance is low). This shows that in the case of SH the natural measure in the intermediate space is the Euclidean distance.

5.3 Asymmetric Distances

In the previous section we decomposed several binary embedding functions h_k into real-valued embedding functions g_k and quantization functions q_k. Let d denote the squared Euclidean distance [3]. We also showed that:

$$d(g(x), g(y)) = \sum_k d(g_k(x), g_k(y)). \tag{5.20}$$

is a natural distance in the intermediate space. We now propose two approximations of the quantity (5.20). In the following, x is assumed non-binarized, *i.e.* we have access to the values $g_k(x)$ (and therefore also to $h_k(x)$), while y is binarized, *i.e.* we only have access to the values $h_k(y)$ (but not to $g_k(y)$).

5.3.1 Expectation-Based Asymmetric Distance

In [63] Jégou *et al.* proposed an asymmetric algorithm for compression schemes based on vector quantization. A codebook is learned through k-means clustering and a database vector is encoded by the index of its closest centroid in the codebook. The distance between the uncompressed query and a quantized database signature is simply computed as the Euclidean distance between the query and the corresponding centroid.

We now adapt this idea to binary embeddings. We note that in the case of LSH, LSBC, SH, or PCAE we have no notion of centroid. However, we note that the centroid of a given cell in k-means clustering can be interpreted as the expected value of the vectors assigned to this particular cell. Similarly, we propose an asymmetric expectation-based approximation d_E for binary embeddings.

Assuming that the samples in the original space Ω are drawn from a distribution p, we define:

$$d_E(x, y) = \sum_k d(g_k(x), \mathbb{E}_{u\sim p}[g_k(u)|h_k(u) = h_k(y)]). \tag{5.21}$$

$d(g_k(x), \mathbb{E}_{u\sim p}[g_k(u)|h_k(u) = h_k(y)])$ is the distance in the intermediate space between $g_k(x)$ and the expected value of the samples $g_k(u)$ such that $h_k(u) = h_k(y)$.

Since we generally do not have access to the distribution p, the expectation operator is approximated by a sample average. In practice, we randomly draw a set of signatures $\mathcal{S} = \{x_i, i = 1 \dots N\}$ from Ω. For each dimension k of the embedding, we partition $\mathcal{S}$ into two subsets: $\mathcal{S}_k^0$ contains the signatures x_i such that $h_k(x_i) = 0$ and $\mathcal{S}_k^1$ those signatures x_i that satisfy $h_k(x_i) = 1$. We compute *offline* the following *query-independent* values (see Fig. 5.1):

$$\alpha_k^0 = \frac{1}{|\mathcal{S}_k^0|} \sum_{u\in\mathcal{S}_k^0} g_k(u), \tag{5.22}$$

$$\alpha_k^1 = \frac{1}{|\mathcal{S}_k^1|} \sum_{u\in\mathcal{S}_k^1} g_k(u). \tag{5.23}$$

[3] We use the following abuse of notation for simplicity: d denotes both the distance between the vectors $g(x)$ and $g(y)$, *i.e.* $d : \mathbb{R}^K \times \mathbb{R}^K \to \mathbb{R}$, and the distance between the individual dimensions, *i.e.* $d : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$.

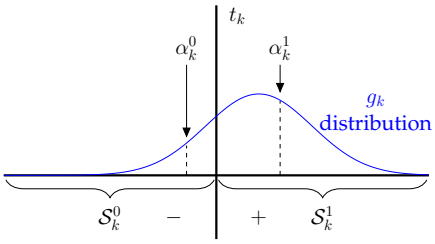


Figure 5.1: Expectation-based asymmetric distance

Online, for a given query x, we first pre-compute and store in look-up tables the following *query-dependent* values:

$$\beta_k^0 = d(g_k(x), \alpha_k^0), \tag{5.24}$$

$$\beta_k^1 = d(g_k(x), \alpha_k^1). \tag{5.25}$$

By definition, we have:

$$d_E(x, y) = \sum_k \beta_k^{h_k(y)}. \tag{5.26}$$

The cost of pre-computing the β values is negligible with respect to the cost of computing many $d_E(x, y)$'s for a large number of database signatures y. Note that the sum (5.26) can be efficiently computed, *e.g.* by grouping the dimensions in blocks of 8 bits and having one 256-dimensional look-up table per block (rather than one 2-dimensional look-up table per dimension). This reduces the number of summations as well as the number of accesses to the memory.

Instead of computing the distance between the query and the expected values of the dataset items as in Equation (5.21), one may find more intuitive to compute the expected distance between the query and the dataset items, *i.e.*,

$$d_{\hat{E}}(x, y) = \sum_k \mathbb{E}_{u\sim p}[d(g_k(x), g_k(u))|h_k(u) = h_k(y)]. \tag{5.27}$$

However, we did not notice any significant difference between this approach and d_E. Furthermore, we would like to note that computing $d_{\hat{E}}$ would require us to calculate and store more statistics than d_E. Particularly, we would need to compute the expected squared values, *i.e.*,

$$\hat{\alpha}_k^0 = \frac{1}{|\mathcal{S}_k^0|} \sum_{u\in\mathcal{S}_k^0} g_k^2(u), \tag{5.28}$$

$$\hat{\alpha}_k^1 = \frac{1}{|\mathcal{S}_k^1|} \sum_{u\in\mathcal{S}_k^1} g_k^2(u). \tag{5.29}$$

as well as α_k^0 and α_k^1.

5.3.2 Lower-Bound Based Asymmetric Distance

In [37], Dong *et al.* proposed an asymmetric algorithm for binary embeddings based on random projections. We now generalize this idea and show that a similar approach can be

applied to a much wider range of binary embedding techniques. For the simplicity of the presentation, we assume that q_k has the form $q_k(u) = \sigma(u - t_k)$ where t_k is a threshold but this can be trivially generalized to other quantization functions.

The idea is to lower-bound the quantity (5.20) by bounding each of its terms. We note that t_k splits $\mathbb{R}$ into two half-lines and consider two cases:

- If $h_k(x) \neq h_k(y)$, *i.e.* $g_k(x)$ and $g_k(y)$ are on different sides of t_k, then a lower-bound between $g_k(x)$ and $g_k(y)$ is the distance between $g_k(x)$ and the threshold t_k, *i.e.* $d(g_k(x), g_k(y)) \geq d(g_k(x), t_k)$.
- If $h_k(x) = h_k(y)$, *i.e.* $g_k(x)$ and $g_k(y)$ are on the same half-line, then we have the following obvious lower-bound: $d(g_k(x), g_k(y)) \geq 0$ (actually, this bound is always true).

Merging the two cases in a single equation, we have the following lower-bound on $d(g(x), g(y))$:

$$d_{LB}(x, y) = \sum_k \bar{\delta}_{h_k(x), h_k(y)} d(g_k(x), t_k), \tag{5.30}$$

where $\bar{\delta}_{i,j}$ is the negation of the Kronecker delta, *i.e.* $\bar{\delta}_{i,j} = 0$ if $i = j$ and 1 otherwise. We note that, in the case of LSH, equation (5.30) is equivalent to the asymmetric LSH distance proposed in [37].

In practice, for a given query signature x, we can pre-compute *online* the values (see Fig. 5.2):

$$\gamma_k^0 = \bar{\delta}_{h_k(x), 0} d(g_k(x), t_k), \tag{5.31}$$

$$\gamma_k^1 = \bar{\delta}_{h_k(x), 1} d(g_k(x), t_k). \tag{5.32}$$

We note that one of these two values is guaranteed to be 0 for each dimension k. We can subsequently pack these values by blocks of 8 dimensions for faster computation as was the case of the expectation-based approximation. By definition we have:

$$d_{LB}(x, y) = \sum_k \gamma_k^{h_k(y)}. \tag{5.33}$$

A major difference between d_E and d_{LB} is that the former one makes use of the data distribution while the latter one does not. Despite its very crude nature, we will see that d_{LB} leads to excellent results on a variety of binary embedding algorithms.

5.3.3 Asymmetric Distances and Variance Preservation

As we will see in the experiments of section 6.7, PCAE seems to benefit particularly from the asymmetric distances (see, *e.g.*, the results on CIFAR in Figures 5.3c and 5.4c). Remember from section 5.2.2 that a disadvantage of PCAE is that each PCA dimension is assigned one bit, independently of the information it may carry. Especially, low-variance dimensions are given as much weight by the Hamming distance as high-variance dimensions. We now show that, in the case of PCAE, asymmetric distances in each dimension are proportional in expectation to the variance of the data, thus giving more weight to the first PCA dimensions. This may explain, at least partly, the significant improvements of asymetric distances for PCAE.

After PCA, the dimensions are uncorrelated and each dimension is supposed to have been generated by a Gaussian p_i (see top row of Figure 5.10) with zero mean and standard deviation σ_i. We assume the dimensions ordered such that $\sigma_i > \sigma_{i+1}$.

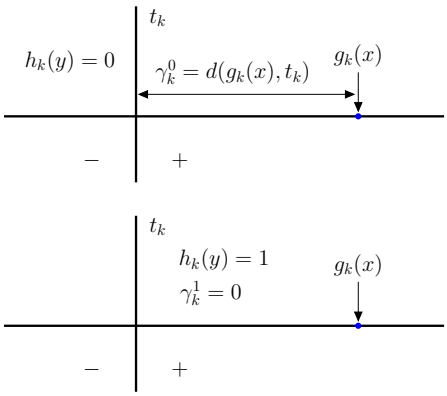


Figure 5.2: Lower-bound-based asymmetric distance. Top: case where $h_k(x) \neq h_k(y)$, and therefore $d(g_k(x), g_k(y)) \geq d(g_k(x), t_k)$. Bottom: case where $h_k(x) = h_k(y)$, and therefore $d(g_k(x), g_k(y)) \geq 0$.

a) d_E **case:** In a given dimension i, the expectation of the positive samples is the expectation of a half-normal distribution

$$E = \int_0^\infty x p_i(x) dx = \frac{\sigma_i}{\sqrt{2\pi}}$$

and the expectation of the negative samples is $-E = -\frac{\sigma_i}{\sqrt{2\pi}}$. These values correspond to the α's of equations (5.22) and (5.23). Consequently, the expectation of the distance to these values (i.e. the expectation of the β's of equations (5.24) and (5.25)) is:

$$\int_{-\infty}^{+\infty} (x - (\pm E))^2 p_i(x) dx = \sigma_i^2 (1 + \frac{1}{2\pi}). \tag{5.34}$$

Therefore *in expectation* the contribution of a given dimension to the asymmetric distance is proportional to the variance in this dimension.

b) d_{LB} **case:** Let us consider the case where $h_k(x) = 1$ and $h_k(y) = 0$. The other relevant case can be treated analogously. The expectations of the γ's (*cf.* equations (5.31) and (5.32)) in dimension i are equal to

$$\int_{-\infty}^{0} 0 \; p_i(x) dx + \int_0^{+\infty} x^2 p_i(x) dx = \frac{\sigma_i^2}{2}.$$

Again, *in expectation*, the contributions of the dimensions to the asymmetric distance decrease with the index i.

5.4 Experiments

We now show the benefits of the asymmetric distances proposed in section 5.3 on the binary embeddings we reviewed in section 5.2. We first describe in section 5.4.1 the four public benchmarks we experiment on. We then provide in section 5.4.2 implementation details for the different embedding algorithms. We finally report and discuss results in section 5.4.3.

5.4.1 Datasets and Features

We run experiments on two category-level retrieval benchmarks, CIFAR and Caltech256, and on two instance-level retrieval benchmarks, the University of Kentucky Benchmark (UKB) and INRIA Holidays. This diverse selection of datasets allows us to experiment with different setups. On CIFAR, we evaluate both semantic retrieval and retrieval using Euclidean neighbors as groundtruth, and there are hundreds or even thousands of relevant items per query. On Caltech256, we evaluate the effect of different descriptors on the results: GIST, Bag of Words and Fisher Vector. UKB and Holidays are standard instance-level retrieval datasets. As opposed to CIFAR or Caltech256, the number of relevant items per query is much smaller, always 4 for UKB and from 2 to 10 for Holidays. We now describe these datasets as well as the associated standard experimental protocols in detail.

CIFAR. The CIFAR dataset [71] is a subset of the Tiny Images dataset [121]. We use the same version of CIFAR that was used in [46]. It contains 64,184 images of size 32×32 that have been manually grouped into 11 ground-truth classes. Images are described using a greyscale GIST descriptor [91] computed at 3 different scales (8, 8, 4) producing a 320-dimensional vector. 1,000 images are used as queries, 5,000 are used for unsupervised training purposes, and the remaining images are use as database images.

Similarly to [46], we report results on two different problems.

- Nearest neighbor retrieval: we discard the class labels. A nominal threshold of the average distance to the 50th nearest neighbor is used to determine whether a database image returned for a given query is considered a true positive. We note that the number of true positives varies widely from one query to another (from 0 to 2,353). We compute the Average Precision (AP) for each query (with a non-zero number of true positives) and report the mean over the 1,000 queries (Mean AP or MAP).
- Semantic retrieval: we use the class labels as groundtruth and report the precision at 1.

Caltech256. The Caltech256 dataset [52] contains approximately 30,000 images grouped in 257 classes. Through our experiments, we use only 256 classes and we discard the "clutter" class. As in CIFAR, we split the dataset in three different sets. We select 5 images per class (1,280 images in total) to serve as queries, and 5,000 random images to serve as unsupervised training data. The remaining images are used as the database. We describe the images and report precision at 1 using 3 different descriptors:

- GIST descriptor with 320 dimensions (same configuration as in CIFAR).
- Bag of Visual Words (BOV) with 1,024 vocabulary words using SIFT descriptors [82]. The low-level features are densely sampled and their dimensionality is reduced from 128 to 64 dimensions with PCA. Instead of k-means, we used a Gaussian Mixture Model to learn a probabilistic codebook and use soft assignment instead of hard assignment. The final histograms are L1-normalized and then square-rooted. As noted in [98, 125], this corresponds to an explicit embedding of the Bhattacharyya kernel when using the dot-product as a similarity measure, and can yield very significant improvements at virtually zero cost.

  All the involved learning (PCA for the low-level features and vocabulary construction) is done using the 5,000 training images.
- Fisher Vector (FV) [96], which was recently shown to yield excellent results for object and scene retrieval [97, 64]. As is the case of BOV, low-level SIFT descriptors are densely sampled and their dimensionality is reduced from 128 to 64 dimensions. We use probabilistic codebooks (*i.e.* GMMs) with 64 Gaussians so that an image is described

by a $64 \times 64 = 4,096$ dimensional FV. Again, the 5,000 training images are used to learn the low-level PCA and the vocabulary.

UKB. The University of Kentucky Benchmark (UKB) [90] contains 10,200 images of 2,550 objects (4 images per object). Each image is used in turn as query to search through the 10,200 images. The accuracy is measured in terms of the number of relevant images retrieved in the top 4, *i.e.* 4 × recall@4. We use the same low-level feature detection and description procedure as in [60] (Hessian-Affine extractor and SIFT descriptor) since the code is available online. As in Caltech256, we reduce the dimensionality of the SIFT features down to 64 dimensions and aggregate them into a single image-level descriptor of 4,096 dimensions using the FV framework. For learning purposes (*e.g.* to learn the PCA on the SIFT descriptors and the GMM for the FV), we use an additional set of 60,000 images (Flickr60K) made available by the authors of [60].

Holidays. The INRIA Holidays dataset [60] contains 1,491 images of 500 scenes and objects. The first image of each scene is used as query to search through the remaining 1,490 images. We measure the accuracy for each query using AP and report the MAP over the 500 queries. As was the case for UKB, images are described using 4,096 dimensional FVs, using the same low-level feature detection and description procedure, as well as the same Flickr learning set.

5.4.2 Implementation Details

To learn the parameters of the embedding functions and to compute offline the α values for d_E we need unlabeled training data. For CIFAR and Caltech256, we will use the $5,000$ training samples, that are used neither as queries nor as database items. For UKB and Holidays we use the Flickr60K dataset [60].

For CIFAR and Caltech256, where no predefined partitions exist, we repeated the experiments 3 times using different queries and database partitions and averaged the results.

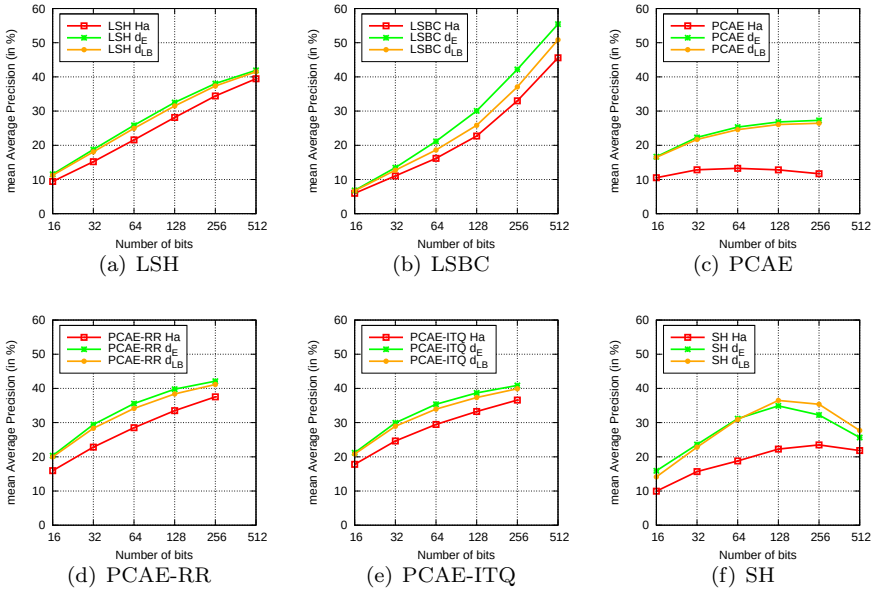
For LSH and LSBC, which perform binarization through random projections, as well as for PCAE-RR and PCAE-ITQ, which use random rotations, experiments are repeated 5 times with 5 different projection matrices and we report the average results.

As discussed in section 5.2.1, mean-centering the data can impact LSH very positively. Therefore, we have mean-centered the GIST and BOV descriptors for CIFAR and Caltech256, learning the means on their respective training sets. This centering has no significant impact for FVs since, by definition, they are already (approximately) zero-mean. We note that centering the data on the origin does not impact PCAE, PCAE-RR, PCAE-ITQ, and SH (which perform a PCA of the signatures) or LSBC (which is shift-invariant).

5.4.3 Results and Analysis

We report results on the four datasets in Figures 5.4 - 5.9 with the symmetric Hamming distance as well as with the proposed asymmetric distances d_E and d_{LB}. The following is a detailed discussion of our findings.

Asymmetric vs symmetric. The use of asymmetric distances consistently improves the results over the symmetric Hamming distance, independently of the dataset, of the descriptor used, and of the binary embedding technique. In general, the gain in accuracy is impressive both in terms of absolute and relative improvement. Here are just two examples: on the CIFAR dataset with semantic labels (Figure 5.4), when using PCAE, we can observe an improvement of 8% absolute and 22% relative at 128 bits. On Holidays, when using SH, we can observe an improvement of 8% absolute and 21% relative (Figure 5.9), also at 128 bits.



(a) LSH (b) LSBC (c) PCAE

(d) PCAE-RR (e) PCAE-ITQ (f) SH

Figure 5.3: Influence of the asymmetric distances on the CIFAR dataset with Euclidean neighbors. The dimensionality of PCAE, PCAE-RR and PCAE-ITQ is limited by the dimensionality of the original GIST descriptor, 320 dimensions.

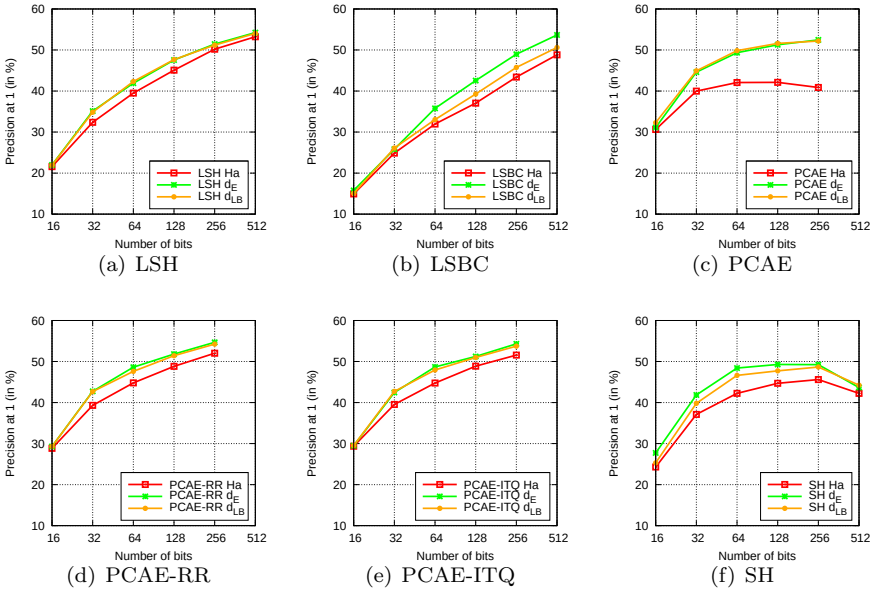
The GIST descriptor results on Caltech256 (Figure 5.5) are an exception to this rule. Indeed, in this setting the simpler Hamming distance can outperform the proposed asymmetric distances, see especially the PCAE results of Figure 5.5c . This seems to indicate that the Euclidean distance we are trying to approximate in the PCA projected space is suboptimal (at least on this dataset and with these features) and that the Hamming distance in the projected space approximates a better metric.

Expectation vs lower-bound. For almost all embedding techniques, d_E and d_{LB} yield very similar results, which is somewhat surprising given that the two approximations are very different in nature. The slight advantage of d_E over d_{LB} comes from the fact that the former approach uses information about the data distribution (through the pre-computed values α) while the latter does not.

We note, however, two exceptions. The first is LSBC, for which in most cases d_E performs significantly better than d_{LB} (see Figures 5.3b, 5.4b, 5.6b, 5.7b, 5.8b, 5.9b). We are still investigating this difference but we observed that the distributions of the values $g_k(x)$ in the intermediate real-valued space for LSBC are significantly different from those observed for the other embedding methods (typically U- or half-U-shaped for LSBC, as opposed to Gaussian-shaped for the others, particularly PCAE, see Figure 5.10). The second exception is on the Caltech256 with GIST descriptors on Figure 5.5, where d_{LB} usually and sometimes very clearly outperforms d_E. As we noted in the previous point, the Euclidean may not be the best measure for GIST descriptors.

Finally, preliminary experiments on fusing d_E and d_{LB} yielded only marginal improvements.

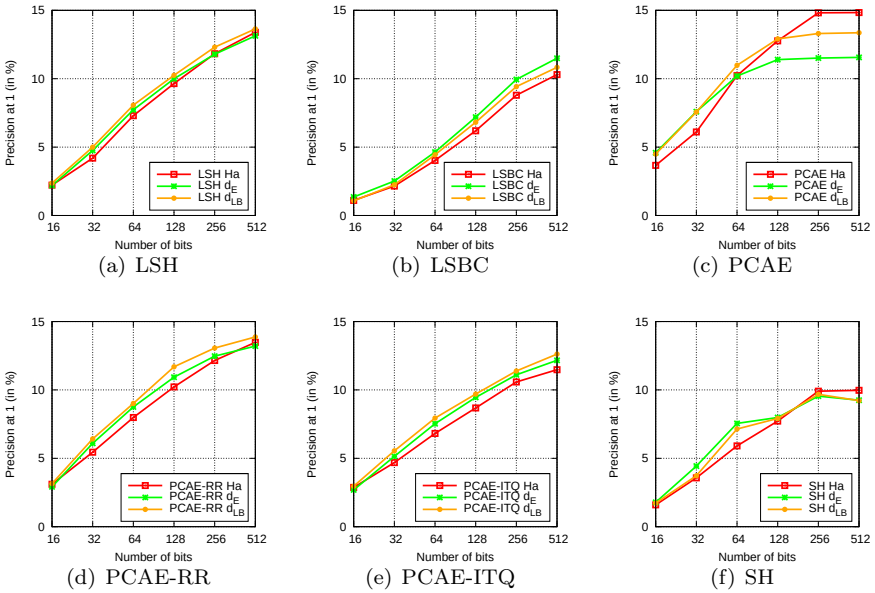
Influence of the retrieval problem. In Figures 5.3 and 5.4 we report results on the CIFAR dataset for two different problems: retrieval of Euclidean neighbors and semantic



(a) LSH (b) LSBC (c) PCAE

(d) PCAE-RR (e) PCAE-ITQ (f) SH

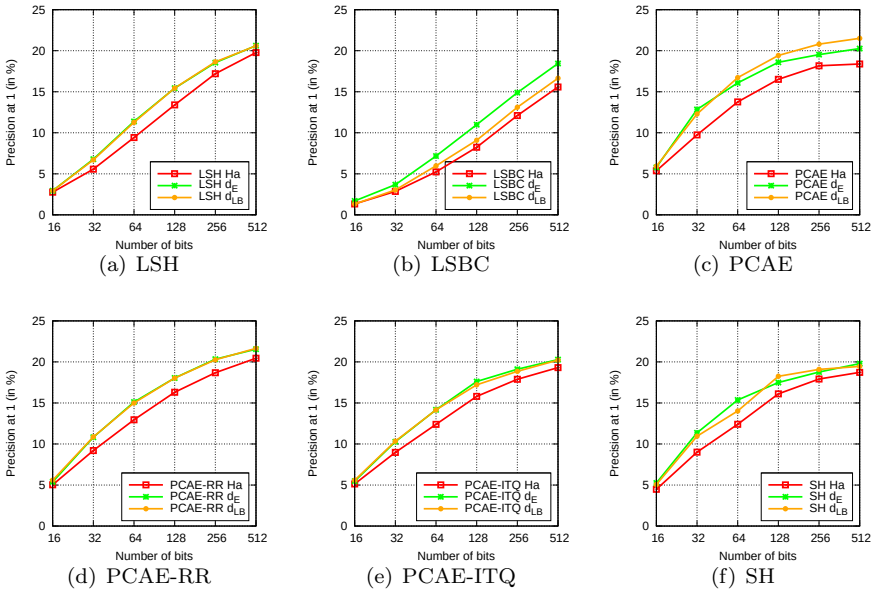
Figure 5.4: Influence of the asymmetric distances on the CIFAR dataset with semantic labels for 6 different encoding methods: LSH, LSBC, PCAE, PCAE-RR, PCAE-ITQ, and SH. The dimensionality of PCAE, PCAE-RR and PCAE-ITQ is limited by the dimensionality of the original GIST descriptor, 320 dimensions.



(a) LSH (b) LSBC (c) PCAE

(d) PCAE-RR (e) PCAE-ITQ (f) SH

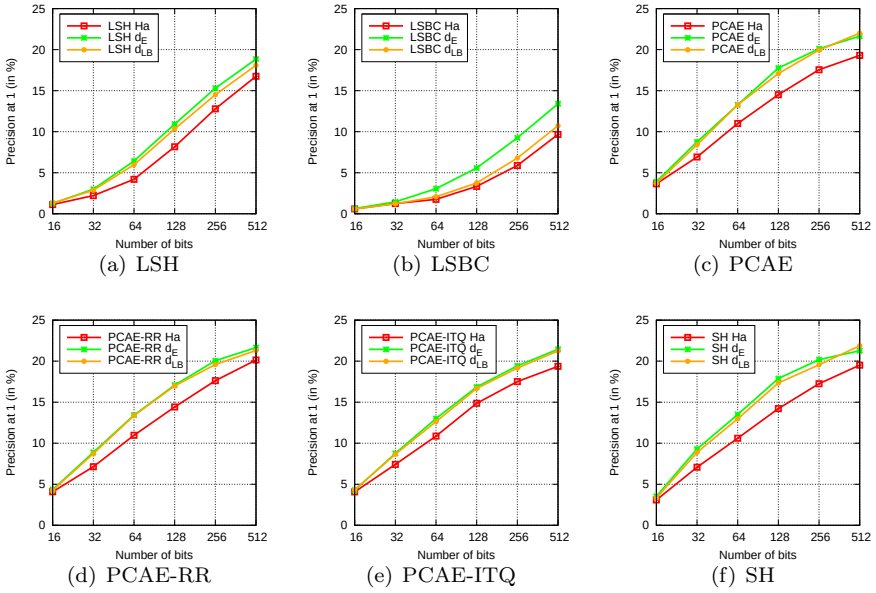
Figure 5.5: Influence of the asymmetric distances on the Caltech256 dataset using GIST descriptors.



(a) LSH (b) LSBC (c) PCAE

(d) PCAE-RR (e) PCAE-ITQ (f) SH

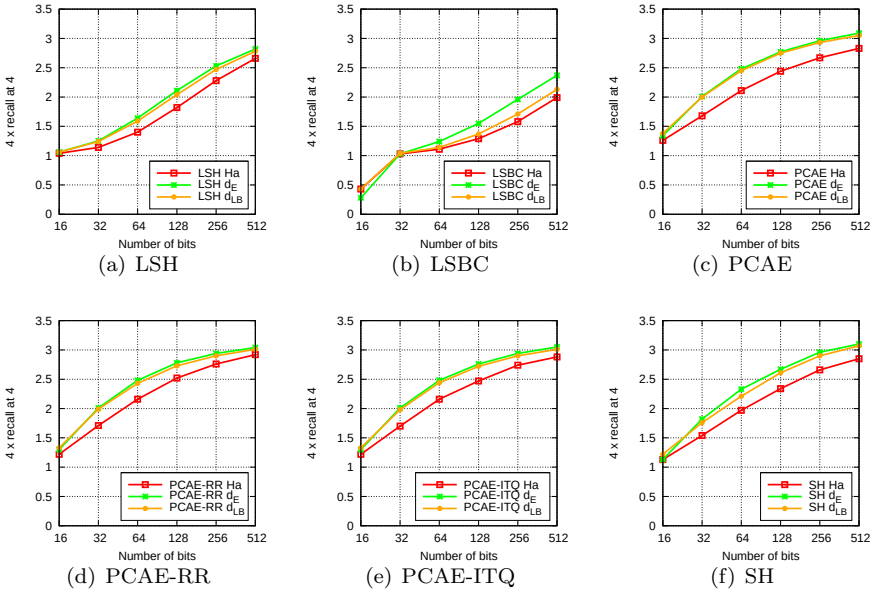
Figure 5.6: Influence of the asymmetric distances on the Caltech256 dataset using BOV descriptors.



(a) LSH (b) LSBC (c) PCAE

(d) PCAE-RR (e) PCAE-ITQ (f) SH

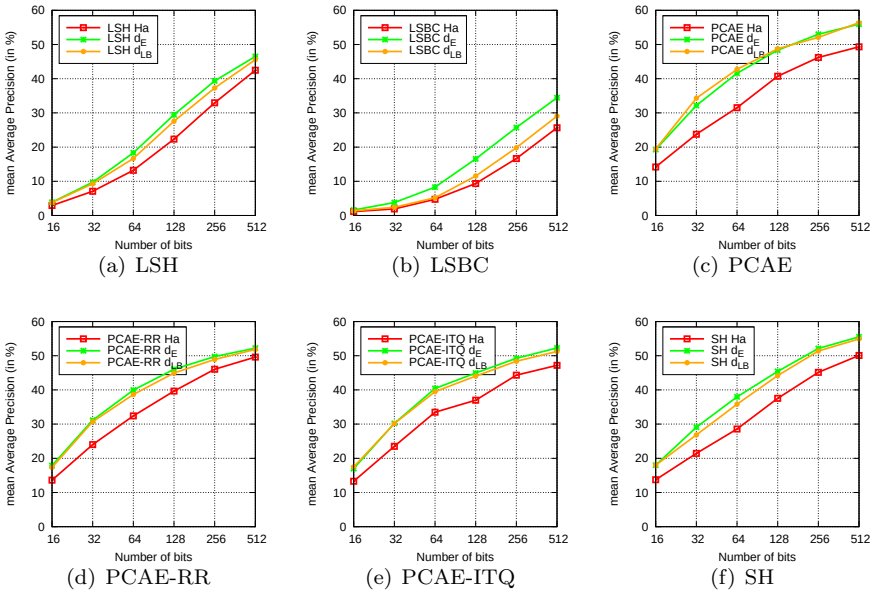
Figure 5.7: Influence of the asymmetric distances on the Caltech256 dataset using FV descriptors.



(a) LSH (b) LSBC (c) PCAE

(d) PCAE-RR (e) PCAE-ITQ (f) SH

Figure 5.8: Influence of the asymmetric distances on the UKB dataset.



(a) LSH (b) LSBC (c) PCAE

(d) PCAE-RR (e) PCAE-ITQ (f) SH

Figure 5.9: Influence of the asymmetric distances on the Holidays dataset.

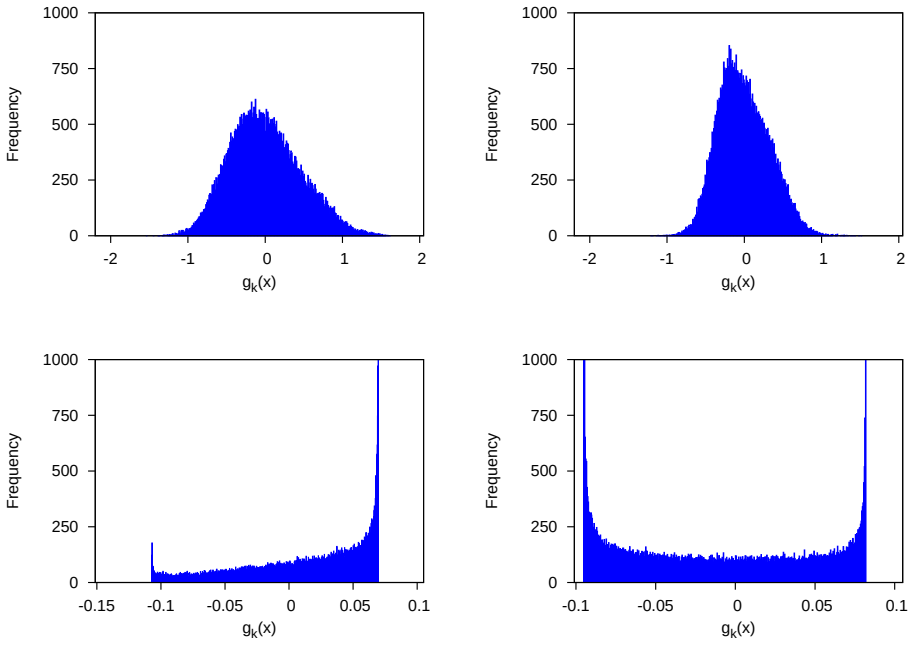


Figure 5.10: Histograms of the projected values of the 60,000 CIFAR images. Top: projected on the first two PCA dimensions. Bottom: projected with two random LSBC dimensions.

retrieval. We can observe how asymmetric distances provide similar improvements for both cases. We can also note how, in the Euclidean problem, PCAE with Hamming distance seems to perform poorly (Fig. 5.3c) compared to the results of PCAE on other datasets, and also how the asymmetric improvements seem larger in this case. We believe the difference stems not from the problem (semantic vs Euclidean retrieval) but because of the evaluation measure: this is the only experiment that combines, at the same time, a large number of relevant items per query and a global measure such as mAP. In such a case, balancing the data with PCAE-RR or PCAE-ITQ seems to yield a large benefit. To attest this, we experimented on Caltech256, which has many relevant items per query. Using BOV descriptors, we computed the mAP score instead of the precision at 1 reported in Figure 5.6c. In that case, PCAE results drastically dropped below those of PCAE-RR and PCAE-ITQ, supporting this idea.

Influence of the descriptor. In Figures 5.5 to 5.7 we can observe the influence of the descriptors on the Caltech256 dataset. In general, the improvements in BOV and FV are larger than the improvements obtained with GIST, showing that the improvement can be dependent on the feature type, particularly if the Euclidean distance was not a good measure in the original space.

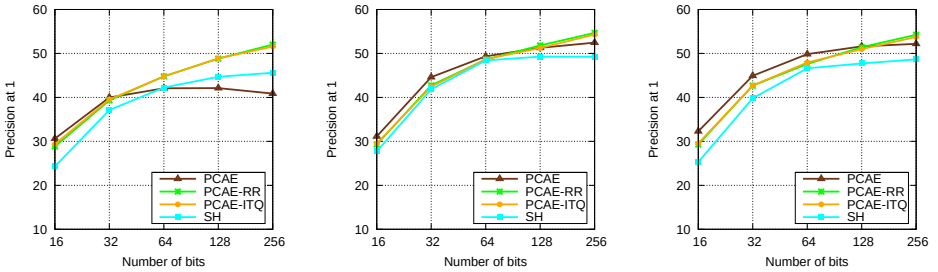
We can also observe how, particularly in the PCA-based methods, FV has a slight edge over BOV when aiming at 256 bits or more. However, BOV can obtain better results than the FV when aiming at signatures of 128 bits or less. This is in line with the observations of [64], where they notice that, when producing small codes, it is usually better to start with a

smaller image signature. In our case, the BOV has 1,024 dimensions and the FV has 4,096. When we can afford larger codes, the FV usually still outperforms BOV: the uncompressed BOV baseline is 22.11%, while the uncompressed FV baseline is 24.11%.

Influence of the embedding method. All methods benefit significantly from the asymmetric distances. This can be easily understood: since we are not binarizing the query, there is less loss of information on the query side.

PCAE seems to benefit particularly from the asymmetric distances (see, *e.g.*, the results on CIFAR in Figures 5.3c and 5.4c). This may be explained by the variance-preservation effect of the asymmetric distances (see section 5.3.3). The variance problem of the other methods is not so severe: LSH and LSBC use random projections, and their variances are balanced in expectation. PCAE-RR, PCAE-ITQ, and SH all balance the variances, either explicitly as in PCAE-RR and PCAE-ITQ, or implicitly, as in SH, assigning more bits to the more important dimensions. Therefore, the impact of the asymmetric distances on these methods is not as pronounced, yet still sizeable. For example, on Caltech256 with FV (Figure 5.7), we show improvements on LSBC of about 4% absolute but almost 40% relative.

Asymmetric distances also seem to bridge the gap between the binary encoding methods, particularly between those based on PCA. Figure 5.11 compares the different encoding methods using Hamming distances and both d_E and d_{LB} on the CIFAR semantic problem with the same data we used for Figure 5.4. The results suggest that asymmetric distances can be used to compensate for the quality of the embedding method; the difference between the encoding methods is significant when using the Hamming distance (more than 10% absolute at 256 bits), but much less pronounced when using asymmetric distances (less than 5% absolute, again at 256 bits).



(a) Hamming distance (b) Asymmetric d_E distance (c) Asymmetric d_{LB} distance

Figure 5.11: Comparison of Hamming and asymmetric distances on the CIFAR dataset with semantic labels. Same data as in Figure 5.4.

Qualitative results. Finally, Figure 5.11 shows qualitative results. We show the top 5 ranked images for four random queries of the Holidays dataset using PCAE with 128 bits, both for Hamming and for asymmetric distances. The false positives have been framed in red. We can observe how, in general, asymmetric distances obtain better and more consistent results than the Hamming distance. We believe that the asymmetric distances allow for better generalization. Figures 5.11b and 5.11c would be an example of this. However, this generalization capability can be pernicious when looking for almost-exact duplicates as in Figure 5.11d, where the Hamming distance retrieves better images.

(a)

(b)

(c)

(d)

Figure 5.11: Top five results of four random queries of Holidays using codes of 128 bits. First row: PCAE + Hamming. Middle row: PCAE + asymmetric d_E distance. Bottom row: PCAE + asymmetric d_{LB} distance.

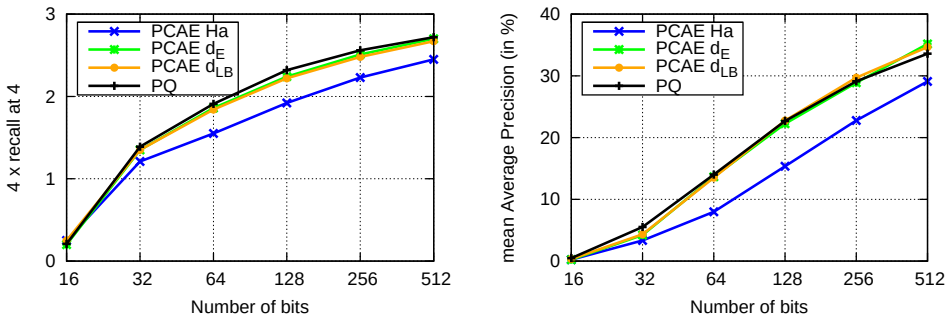
5.4.4 Large-Scale Experiments

We now show that the good results achieved with asymmetric distances scale to large datasets. For these experiments we merge Holidays and UKB with a set of 1M Flickr distractors made available by the authors of [60]. We refer to these combined datasets as Holidays+1M and UKB+1M. In both cases we use the original queries, 500 in Holidays and 10,200 in UKB. We experiment with PCAE, since it is the method that obtained, in general, the best results, and it is simpler than PCAE-RR and PCAE-ITQ.

Figure 5.12 shows the results on both datasets as a function of the number of bits. We compare them with Product quantization (PQ) [63, 64], since, to the best of our knowledge, these are the best results reported on Holidays+1M for very small operating points. For PQ, we employ the same pipeline as [64] which is composed of the following steps: i) PCA compression of the signatures, ii) random orthogonal rotation of the PCA projected signatures, iii) Product quantization and iv) comparison using PQ's asymmetric distances, referred to as ADC in [64].

To fix the number of dimensions D' in the PCA projection step of PQ, the authors minimized the mean square error of the projection and the quantization over a training set. We follow a different heuristic: we set D' to be equal to the number of output bits we are aiming at, and assign 8 dimensions to each subquantizer. We then fix the number of bits per subquantizer to 8, since this seems to be a standard choice that usually offers excellent results. Experimentally, we observed this heuristic to obtain comparable or better results than minimizing the mean square error, and in most cases was the best possible configuration. As was the case before, experiments are repeated 5 times with different projection matrices and the results are averaged.

We can observe how both asymmetric distances with PCAE perform comparably to PQ on both datasets although PCAE is simpler than PQ both from a conceptual and an engineering standpoint. Furthermore, as opposed to PQ, the lower-bound asymmetric distance does not require any training.



(a) UKB+1M (b) Holidays+1M

Figure 5.12: Comparison of the proposed asymmetric distances and PQ [64] on UKB+1M (left) and Holidays+1M (right). MAP and 4 × recall at 4 as a function of the number of bits.

5.5 Experiments with Documents

So far, for convenience reasons, all experiments in this chapter have been carried on on natural image datasets. However, the proposed methods are general and can be directly applied to our document signatures. Through this section we will revisit some of the experiments performed on chapters 2 to 4, and show how the signatures can be significantly compressed down to just a few bits while still preserving most of the accuracy. This would allow us to keep millions of documents simultaneously in memory, as well as significantly speeding up the retrieval process.

5.5.1 Single-Page Documents

Let us begin with the single-page experiments of chapter 2. In Table 2.1 we showed classification results on the NIST and MARG datasets using a 1-NN classifier. Pages were described with our runlength descriptor and the dimensionality was reduced with PCA. In Tables 5.1 and 5.2 we replicate those results. We also inlude the classification results after applying PCAE and PCAE-RR binarization instead of just PCA, using both Hamming and the proposed asymmetric distances. We can observe how we can very significantly reduce the dimensionality of the data while preserving the good results. On NIST, using PCAE-RR with asymmetric distances, we can obtain a 99.92 accuracy (state-of-the-art, *cf*. Table 2.2) using *only* 16 bits per descriptor. Similarly, we can obtain results extremely close to the state-of-the-art on the MARG dataset using as few as 64 bits per descriptor, and improve the state-of-the-art using 128 bits descriptors. Furthermore, in chapter 2 we reported times of 100ms to classify the 5,590 documents of the NIST dataset using 200 training samples, and 250ms to classify all the documents of the MARG dataset in a leave-one-out strategy. The descriptors had 64 dimensions, but where not binarized. These timings were already significantly faster than those of the state-of-the-art methods that we used for reference purposes. Now, using binary descriptors of 64 bits, we can perform the whole classification task on NIST in less than 20ms, and less than 25ms on MARG, a 5-10 fold improvement in speed.

Table 5.1: Classification accuracy (in %) as a function of the number of dimensions / bits on the NIST dataset.

dimensions / bits	8	16	32	64	128
PCA (non-binary)	99.99	100.0	100.0	100.0	100.0
PCAE Ha	90.90	99.22	97.28	91.99	83.75
PCAE d_E	90.81	99.43	99.94	99.95	99.95
PCAE d_{LB}	91.89	99.84	99.88	99.89	99.90
PCAE-RR Ha	93.06	99.72	99.91	99.99	**100.0**
PCAE-RR d_E	**94.93**	99.89	99.97	**100.0**	**100.0**
PCAE-RR d_{LB}	93.37	**99.92**	**99.99**	**100.0**	**100.0**

In chapter 2 we also performed retrieval experiments on the NIST, MARG, and IH1 datasets. Those results were reported in Table 2.6. Again, we replicate those experiments, including binarization through PCAE and PCAE-RR on Tables 5.3, 5.4, and 5.5. We can observe how asymmetric distances improve over the Hamming distance in almost all cases, and how that difference can be quite significant. We can also observe how descriptors can

Table 5.2: Classification accuracy (in %) as a function of the number of dimensions / bits on the MARG dataset.

dimensions / bits	8	16	32	64	128
PCA (non-binary)	74.31	89.76	92.79	94.46	94.66
PCAE Ha	32.90	65.87	84.35	90.60	92.21
PCAE d_E	30.26	68.90	85.45	89.50	91.50
PCAE d_{LB}	**32.97**	**71.02**	**87.89**	**92.53**	93.63
PCAE-RR Ha	32.71	63.68	78.49	87.70	91.44
PCAE-RR d_E	29.75	64.38	86.22	91.57	92.79
PCAE-RR d_{LB}	31.68	68.38	86.74	91.44	**93.69**

be compressed down to 128 bits with no significant loss. In some cases, we can compress the descriptors even more: on NIST, we can obtain a 100% accuracy when reporting precision at 5 using as few as 16 bits. On IH1, we can compress down to 32-64 bits with minimal loss.

Table 5.3: Precision at 5 (in %) as a function of the number of dimensions / bits on the NIST dataset.

dimensions / bits	8	16	32	64	128
PCA (non-binary)	100.0	100.0	100.0	100.0	100.0
PCAE Ha	90.03	**100.0**	**100.0**	99.90	99.97
PCAE d_E	90.21	99.91	**100.0**	**100.0**	**100.0**
PCAE d_{LB}	91.23	**100.0**	99.91	**100.0**	**100.0**
PCAE-RR Ha	94.51	**100.0**	99.66	99.97	**100.0**
PCAE-RR d_E	93.34	99.91	99.99	**100.0**	**100.0**
PCAE-RR d_{LB}	**94.79**	**100.0**	99.93	**100.0**	**100.0**

Table 5.4: Precision at 5 (in %) as a function of the number of dimensions / bits on the MARG dataset.

dimensions / bits	8	16	32	64	128
PCA (non-binary)	64.97	77.63	82.09	84.11	84.73
PCAE Ha	33.24	56.89	71.79	78.53	79.36
PCAE d_E	32.96	62.60	74.07	79.31	80.62
PCAE d_{LB}	34.80	**64.67**	**77.20**	**82.56**	**83.75**
PCAE-RR Ha	34.45	53.72	68.32	76.54	80.38
PCAE-RR d_E	32.48	56.67	73.55	80.03	82.62
PCAE-RR d_{LB}	**35.35**	59.35	73.91	80.41	83.03

Table 5.5: Precision at 5 (in %) as a function of the number of dimensions / bits on the IH1 dataset.

dimensions / bits	8	16	32	64	128
PCA (non-binary)	85.88	90.87	91.58	91.68	91.53
PCAE Ha	61.43	81.07	85.35	86.71	87.34
PCAE d_E	62.13	83.56	88.03	88.91	89.46
PCAE d_{LB}	**62.42**	**84.42**	88.70	90.05	90.67
PCAE-RR Ha	59.52	78.67	86.81	89.44	90.53
PCAE-RR d_E	58.37	80.78	88.55	90.54	91.27
PCAE-RR d_{LB}	60.84	81.87	**89.07**	**90.88**	**91.48**

5.5.2 Multiple-Page Documents

The PCAE binarization and asymmetric distances can also be applied to the Bag of Pages and Bag of Page-Classemes signatures that we used to represent multiple-page documents in chapter 3. In Tables 5.6 and 5.7 we show the results using binarized Bag of Pages and Bag of Page-Classemes descriptors on the IH2 large dataset. These results complement the ones shown in Table 3.2, which used non-binary descriptors. We can observe how, as with the single-page descriptors, we can significantly compress and binarize the signatures with a minimum loss with respect to the PCA compressed ones. For example, using the preferred Bag of Page-Classemes representation, we can reduce the descriptors down to 64 bits with no loss thanks to the asymmetric distances. We can further reduce the descriptors down to just 32 bits, and still obtain resutls that are only one point below the PCA compressed signatures.

Table 5.6: Precision at 5 (in %) as a function of the number of dimensions / bits on the IH2 dataset using a Bag of Pages representation.

dimensions / bits	32	64	128	256
PCA (non-binary)	28.65	29.22	29.15	29.11
PCAE Ha	24.45	25.50	25.57	25.15
PCAE d_E	26.95	28.78	29.90	**30.23**
PCAE d_{LB}	**27.46**	**28.91**	**29.94**	29.95
PCAE-RR Ha	22.53	24.92	26.32	27.10
PCAE-RR d_E	23.81	26.00	26.85	27.23
PCAE-RR d_{LB}	24.24	26.37	27.35	27.97

5.5.3 Documents with Combined Sources of Information

Finally, we show that the PCAE binarization and asymmetric distances can also be applied to the descriptors we used in chapter 4. In Table 4.1 we presented results on the IH3 dataset, compressing the descriptors with PCA and with CCA+PCA, which enriched the

Table 5.7: Precision at 5 (in %) as a function of the number of dimensions / bits on the IH2 dataset using a Bag of Page-Classemes representation.

dimensions / bits	32	64	128	256
PCA (non-binary)	32.73	33.61	36.04	35.27
PCAE Ha	26.44	29.19	28.81	22.70
PCAE d_E	**31.58**	34.27	34.55	34.57
PCAE d_{LB}	30.89	33.31	33.14	33.11
PCAE-RR Ha	27.93	31.90	33.45	34.90
PCAE-RR d_E	30.82	**34.46**	**35.13**	**35.73**
PCAE-RR d_{LB}	30.63	34.14	34.80	35.60

visual representation with some textual information learned at training time. As in the previous sections, we replicate those experiments on Tables 5.8 and 5.9. We can observe how the gain obtained with CCA to include textual information is still preserved after performing the PCAE binarization. We can also observe how, in this dataset, we need 256 bits or more to replicate the non-binary results. This is reasonable since this is a much more difficult dataset containing 181 classes, and so it requires larger signatures to obtain decent results.

Table 5.8: Precision at 5 (in %) as a function of the number of dimensions / bits on the IH3 dataset using PCA.

dimensions / bits	32	64	128	256	512
PCA (non-binary)	33.98	35.62	36.44	36.87	36.98
PCAE Ha	25.77	31.10	35.12	**37.96**	**39.46**
PCAE d_E	28.23	32.22	34.35	35.61	36.16
PCAE d_{LB}	**29.30**	**33.29**	**35.72**	37.08	37.78
PCAE-RR Ha	22.71	28.50	32.14	34.52	36.01
PCAE-RR d_E	25.37	30.84	33.72	35.39	36.42
PCAE-RR d_{LB}	25.96	31.03	33.92	35.57	36.71

5.6 Conclusions

In this chapter we proposed two asymmetric distances for binary embedding techniques, *i.e.* distances between binarized and non-binarized signatures. We showed their applicability to several embedding algorithms: LSH, LSBC, SH, PCAE, PCAE-RR, and PCAE-ITQ. We demonstrated on four natural image datasets with up to 1M images that the proposed asymmetric distances consistently, and often very significantly, improve the retrieval accuracy over the symmetric Hamming distance. We also showed how this asymmetric distances can achieve results comparable to state-of-the-art methods such as PQ, while being conceptually much simpler. The lower-bound asymmetric distance can also be applied on datasets that have already been binarized, with no need to perform any reencoding or extra training.

Table 5.9: Precision at 5 (in %) as a function of the number of dimensions / bits on the IH3 dataset using CCA+PCA.

dimensions / bits	32	64	128	256	512
CCA+PCA (non-binary)	40.42	43.94	45.70	45.48	45.77
CCA+PCAE Ha	25.98	34.04	38.30	39.370	39.21
CCA+PCAE d_E	30.63	39.21	**43.49**	44.81	45.24
CCA+PCAE d_{LB}	**31.62**	39.04	42.74	43.82	44.25
CCA+PCAE-RR Ha	26.77	35.22	40.18	42.66	44.05
CCA+PCAE-RR d_E	31.49	**40.03**	43.96	**45.25**	**45.82**
CCA+PCAE-RR d_{LB}	31.61	39.34	43.18	44.61	45.36

Finally, we also tested the PCAE binarization plus asymmetric distances on several document datasets used through the previous chapters, revisiting some of the experiments we already performed and showing that document signatures can also be compressed with a minimal accuracy loss.

Chapter 6

Leveraging Category-Level Labels[1]

6.1 Introduction

In this chapter we consider the problem of query-by-example instance-level image retrieval. For example, if we deal with natural images, given a query image of an object or a scene, we want to retrieve within a potentially large dataset other instances of the exact same object or scene. We are interested not only in the semantic category of the image, but we also want the retrieved images to be as similar to the instance query as possible. Similarly, when dealing with documents in our digital mailroom, we are interested in retrieving documents that are not only from the same document class, but as similar to the query as possible according to some criteria. This is useful for several reasons. For example, if we retrieve the k most similar documents from the dataset, both from a semantic and visual standpoint, we can use those documents to transfer information such as the layout structure. Note that this is a different problem from retrieving the *exact* document in a dataset but captured from a different source, *e.g.*, the query image has been captured using a scanner or a camera, but the database item is a pure digital copy. Although that is an important topic (see, for example, the best paper award at DAS 2010 [58]), it is out of the scope of this thesis.

Most state-of-the-art large-scale retrieval systems consist in extracting local descriptors, such as SIFT [82], and aggregating them into a fixed-length vector. Within this broad framework, we can distinguish two fairly different lines of research. The first one is based on the bag-of-visual-words (BOV) framework [115] and describes an image as a very high-dimensional and very sparse histogram of visual-word counts. Retrieval efficiency is achieved through the use of inverted files. While such an approach can obtain excellent results [60, 92], it is difficult to scale to more than a couple of millions of images without dedicated hardware. The second one consists in describing images with typically smaller and denser vectors, such as the GIST [91], the Fisher Vector [96, 99] or the VLAD [64], and then performing some form of encoding. Note that our runlength descriptors belong to this category. It has been shown that, even with fairly small codes consisting of a few hundreds of bits, this approach could yield excellent results at a very low cost (see *e.g.* [130, 100, 97, 64, 46]). In this work, we follow this second line of research.

As noted in chapter 5, most encoding techniques include a projection step which is generally learned in an unsupervised manner. Our goal in this chapter is to learn a better

[1]This chapter published in *A. Gordo, J. A. Rodriguez-Serrano, F. Perronnin and E. Valveny. Leveraging Category-Level Labels for Instance-Level Retrieval. In CVPR, 2012.*

projection by leveraging labeled data to improve the retrieval accuracy for a target compression rate (or the compression rate for a target accuracy). Note that, since we learn the dimensionality reduction in a manner which is independent of a particular encoding technique, our work has the potential to impact a broad range of retrieval algorithms.

An important question is the source of labeled data which we should use for supervised learning. Since our goal is to perform *instance-level* retrieval, it would only seem natural to use datasets labeled at the instance level. However, these datasets are typically small and as a consequence insufficient to learn a good subspace (this is shown experimentally in section 6.7.2). For instance, the two standard instance-level datasets we use in our experiments contain only 1,500 and 10,000 images approximately. On the other hand, there exist very large datasets of images annotated at the *category-level* such as ImageNet [34] which contains as of today around 14M images of 22,000 categories. Therefore, we ask in this chapter the following question: *can category-level labels be used to improve instance-level image retrieval?*.

This actually calls for another question: why should category-level labels help instance-level retrieval in the first place? We note that typical instance-level retrieval systems sometimes make gross mistakes, *i.e.* return among the top ranked results images which are visually similar but semantically unrelated. Injecting category-level information in the dimensionality reduction step should guide the retrieval system towards more semantically consistent results as shown for instance in Figure 6.1.

In this chapter we will study four algorithms which learn a set of projections from labeled data. The first one is based on *metric learning* and casts the problem of dimensionality reduction as that of learning a low-rank Mahalanobis metric [9, 22]. Using a large margin framework, similar images are enforced to be closer in the subspace than dissimilar ones. The second one proposes to learn a set of classifiers and to represent an image as a vector of *attribute scores* [126, 122, 38]. The similarity between two images is then computed in this attribute space. The third one is based on *Canonical Correlation Analysis* (CCA) [55] and performs an embedding of labels and images in a common subspace in which the similarity can be computed [13, 46]. The fourth one consists in learning *jointly a subspace and classifiers* (JSCL). The classifiers are subsequently discarded and only the subspace information is used for retrieval.

Our experiments show that the joint classifier and subspace learning approach performs best. For instance, in large-scale experiments on the Holidays dataset, we improve the PCA baseline from 39.3% to 48.6% for a target of 32 dimensions. Hence our two main contributions in this chapter are to show (i) that category-level labeled data can be leveraged to improve instance-level retrieval and (ii) that jointly learning a set of classifiers and a dimensionality reduction using a large margin framework achieves this goal.

The remainder of this chapter is organized as follows. In the next section we review the related work. In sections 6.3 to 6.6, we describe the subspace learning approaches we experimented with: metric learning, attributes, CCA and joint classifier and subspace learning. In section 6.7 we compare these four algorithms on two public benchmarks.

6.2 Related Work

We now review related work in the fields which are closest to our large-scale retrieval problem – data encoding, metric learning and attribute-based retrieval – while emphasizing the differences with our own work.

Data encoding. Many works have proposed to transform high-dimensional vectorial representations into compact codes. This includes Locality Sensitive Hashing (LSH) [56, 20],

Figure 6.1: Results for four Holiday queries on a dataset of 1M+ images. For each query (left image), we show the top 5 retrieved images using PQ codes of 128 bits: the top row corresponds to the PCA projection baseline and the bottom row to the semantic projection with the proposed JSCL. Green frames denote correct results. See section 6.7 for experimental details.

Spectral Hashing (SH) [130], Hamming Embedding (HE) [60], Locality Sensitive Binary Coding (LSBC) [100], Packing [62], Semi-Supervised Hashing (SSH) [127], Transform Coding (TC) [17], PCA Embedding (PCAE) [50], Iterative Quantization (ITQ) [46] or Product Quantization (PQ) [63, 64]. Despite the significant differences between these algorithms, all of them include a projection of the original image signatures into an intermediate real-valued space, as noted in chapter 5. The projections are either random (as in LSH, LSBC, HE or Packing) or learned in an unsupervised manner, for instance with PCA (as in SH, TC, SSH, PCAE, PQ) or with an algorithm which reduces the quantization error (as in ITQ). The only work we are aware of which leverages labeled data to learn better embeddings for large-scale retrieval is that of Gong and Lazebnik [46]. For this purpose, they propose to use CCA. This is one of the approaches we will experiment with (*cf*. section 6.5). Note however that [46] uses category-level labels to improve category-level retrieval (also referred to as "semantic" retrieval) while we are interested in leveraging category-level labels to improve *instance-level* retrieval.

Metric learning[2]**.** Several works have proposed to leverage category-level labels to learn a similarity measure (or a distance) between two image descriptors. Note that there is a significant body of work in the machine learning community on how to "learn to retrieve" [65, 129, 9, 22, 29]. Metric learning has application to category-level image retrieval [22] but also to problems such as domain adaptation [74].

Attributes. An alternative to metric learning which has recently become popular consists in learning a set of attributes and in describing an image by a vector of attribute scores (see [76, 126, 122, 38, 33, 35, 103] among others). Again, almost all these works have considered the problem of leveraging category-level labeled data to improve category-level retrieval. A noticeable exception is the work of Douze *et al*. who proposed to use category-level labels to improve instance-level retrieval by fusing Fisher Vectors and attributes [38]. Therefore, we will experiment with attributes in our study (*cf*. section 6.4). However, while [38] reports a significant accuracy improvement with respect to a PCA baseline, our results are somewhat different (*cf*. section 6.7.3).

6.3 Metric Learning

In an image retrieval task, let $q, d \in \mathbb{R}^D$ denote the D-dimensional feature vectors representing a query and a database image, respectively. We consider parametric image similarities given by the bilinear form

$$s(q,d) = q^T W d, \tag{6.1}$$

where $W \in \mathbb{R}^{D \times D}$. When $W = I$, $s(q,d)$ reduces to the dot-product. Instead of optimizing W directly, we consider the decomposition $W = U^T U$, as proposed for instance in [9], where $U \in \mathbb{R}^{R \times D}$ (with $R < D$). Then Eq. (6.1) can be re-written as

$$s(q,d) = q^T U^T U d = (Uq)^T (Ud). \tag{6.2}$$

Eq. (6.2) is interesting from the point of view of data compression, since it expresses the similarity as a dot-product in a low dimensional space given by the projection matrix U. Optimizing U thus amounts to finding the linear sub-space in which the dot-product is an optimal similarity measure.

A natural framework to learn U is the large margin ranking framework [9]. Given a query q, a relevant image d^+ and an irrelevant image d^-, a good similarity measure satisfies the

[2]In what follows, we abuse the language and use the term "metric learning" to refer to the body of work which includes both distance and similarity learning

property: $s(q, d^+) > s(q, d^-)$, *i.e.* matching pairs should have a higher similarity than non-matching pairs. Given a set of triplets (q, d^+, d^-), the goal is to minimize an upper-bound on the ranking loss:

$$\sum_{(q,d^+,d^-)} \max\{0, 1 - s(q, d^+) + s(q, d^-)\}. \tag{6.3}$$

Since it is typically infeasible to scan all possible triplets, this loss function can be optimized using Stochastic Gradient Descent (SGD) [16]. Following straightforward derivations, it is possible to show that the training procedure consists in repeating the two following steps: (i) sample a triplet (q, d^+, d^-) randomly, and (ii) perform the gradient update

$$U \leftarrow U + \eta U(q\Delta^T + \Delta q^T) \tag{6.4}$$

if the loss $\max\{0, 1 - s(q, d^+) + s(q, d^-)\}$ is positive, where $\Delta = d^+ - d^-$ and η is the learning rate. Although the objective function (6.3) is not convex after the low-rank decomposition, it was shown in [9] that good results are obtained in practice by initializing the values of U randomly (from a zero-mean Normal distribution). We also experimented with an initialization from the PCA solution but this did not make a major difference. Also, following [9] we do not regularize U explicitly (e.g. by penalizing the Frobenius norm of U) but implicitly with early stopping.

6.4 Attributes

The principle of attribute-based representations is to describe an image with respect to a set of K "discriminative" concepts $\mathcal{A} = \{a_1, \dots a_K\}$ referred to as attributes. The relevance $s(q, a_k)$ of the image q with respect to each attribute a_k is measured and the final representation is a K-dimensional vector of attribute scores:

$$[s(q, a_1), \dots, s(q, a_K)]\,. \tag{6.5}$$

In the vast majority of cases, the attributes are learned using a large margin framework[3], *e.g.* by training one binary Support Vector Machine (SVM) classifier for each attribute [76, 126, 122, 38, 33, 103]. If the number of attributes is smaller than the number of dimensions in the original space (a desirable property in general), then this representation can be understood as the projection of a high-dimensional representation onto a "semantic subspace". Simple metrics, such as the dot-product of the Euclidean distance are typically used to measure the similarity within the attribute space.

An issue with the attribute-based approach is that the dimensionality of the subspace is fixed given the number of attribute classes. However, in practice, one would like to be able to tune the dimensionality of the subspace based, for instance on a target compression factor. Douze *et al.* explored two simple approaches to circumvent this problem [38]. The first one consists in selecting a subset of attribute classes while the second one simply consists in applying PCA on the vector of attribute scores. Since the later approach was found to yield better results, this is the one we used in our own experiments. Note that Douze *et al.* also proposed to merge Fisher Vectors and attribute vectors by concatenating these representations. This is an approach we will also evaluate in section 6.7.3.

[3] An exception is [35] which uses k-NN classification to measure the relevance of an image with respect to an attribute. While this approach was reported to yield excellent results for category-level retrieval, we found it to yield poor results in our instance-level retrieval scenario.

6.5 Canonical Correlation Analysis

Canonical Correlation Analysis (CCA) [55] is a well-known tool for multi-view dimensionality reduction. In a nutshell, the goal of CCA is to project the multiple views into a common subspace where the correlation is maximal.

Let us consider a set of N samples, and let $A \in \mathbb{R}^{D_a \times N}$ and $B \in \mathbb{R}^{D_b \times N}$ be two views of the data represented by mean-centered column feature vectors. In general, the dimensionality of the vectors in A and B are different, *i.e.* $D_a \neq D_b$. Let us also define the matrices $C_{aa} = AA^T + \rho I$, $C_{bb} = BB^T + \rho I$, $C_{ab} = AB^T$, and $C_{ba} = C_{ab}^T$, where ρ is a small regularization factor usually added to avoid numerically ill-conditioned situations.

The goal of CCA is to find a projection of each view that maximizes the correlation between the projected representations. This can be expressed as:

$$\underset{u \in \mathbb{R}^{D_a}, v \in \mathbb{R}^{D_b}}{\operatorname{argmax}} \quad u^T C_{ab} v \tag{6.6}$$

$$\text{s.t.} \quad u^T C_{aa} u = 1 \text{ and } v^T C_{bb} v = 1. \tag{6.7}$$

u and v are respectively the projections that embed the data from A and B into a one-dimensional common subspace where the correlation is maximal. To obtain a subspace of R dimensions we need to solve equation (6.6) R times to obtain the set of projections $\{u_1, u_2, \ldots, u_R\}$ and $\{v_1, v_2, \ldots, v_R\}$, subject to them being uncorrelated. This can be casted as a generalized eigenvalue problem:

$$C_{aa}^{-1} C_{ab} C_{bb}^{-1} C_{ba} u_R = \lambda_R^2 u_R. \tag{6.8}$$

The R leading eigenvectors of equation (6.8) constitute the projection matrix $U \in \mathbb{R}^{R \times D_a}$ used to embed A into the R-dimensional subspace. The embedding of B, if needed, can be solved analogously.

In [46], CCA was used to perform supervised dimensionality reduction using respectively the image descriptors and labels as the two views. The labels were encoded as a matrix $B \in \{0,1\}^{K \times N}$, where K is the number of classes, and where $B_{k,n} = 1$ if image n belongs to category k, and 0 otherwise. In such a case, CCA can be understood as an embedding of images and labels in a common subspace.

6.6 Joint Subspace and Classifier Learning

As is the case of CCA, we seek to embed labels and images in a common subspace. However, we wish to do so in a large margin framework. Given an image and a set of relevant and irrelevant labels, we want to enforce the relevant labels to be closer to the image in the subspace than the irrelevant ones. This process can be understood as jointly learning a set of classifiers and a dimensionality reduction. This is more optimal than learning a set of attribute classifiers and then a dimensionality reduction as in [38].

We now describe the mathematical framework. Let q be an image descriptor and let y be a category. We assume that $q \in \mathbb{R}^D$ and that there are K categories, *i.e.* $y \in \{1, \ldots, K\}$. Let us measure the relevance of y with respect to q (*i.e.* the score of class y on q) as follows:

$$s(q, y) = (Uq)^T w_y \tag{6.9}$$

where $U \in \mathbb{R}^{R \times D}$ matrix which projects q in a R dimensional subspace (with $R < D$ and $R \leq K$) and w_y is the classifier of class y in the low-dimensional space. Hence, the projection matrix U is shared across all classes. Given a set of triplets (q, y^+, y^-) where y^+ is relevant

to q and y^- is irrelevant to q (*i.e.* q is labeled with y^+ but not with y^-), we minimize an upper-bound on the label ranking loss:

$$\sum_{(q,y^+,y^-)} \max\left\{0, 1 - s(q,y^+) + s(q,y^-)\right\} \tag{6.10}$$

Weston *et al.* proposed a similar objective function in [131] for annotation purposes. In what follows, we choose to optimize equation (6.10) because it is more similar to the metric learning framework of [9] that we use as a baseline and therefore, it offers a fairer comparison. Note that we also ran experiments with the objective function proposed by Weston *et al.* and we found it to yield very similar results.

As was the case for metric learning, this objective function can be optimized with SGD by sampling a triplet (q, y^+, y^-). If the loss $\max\left\{0, 1 - s(q,y^+) + s(q,y^-)\right\}$ is positive, then the following update rules are applied:

$$U \leftarrow U + \eta(w_{y^+} - w_{y^-})q^T \tag{6.11}$$

$$w_{y^+} \leftarrow w_{y^+} + \eta U q \tag{6.12}$$

$$w_{y^-} \leftarrow w_{y^-} - \eta U q \tag{6.13}$$

where η is again the learning step size. As was the case for metric learning, we initial the matrix U randomly (from a zero-mean Normal distribution) and use early stopping for regularization. After learning, we discard the classifiers w_y and keep only the projection matrix U.

6.7 Experimental validation

We first describe the datasets and features we used in our experiments. We then provide results for the metric learning and attribute-based approaches. Finally, we present results for the two label-image embedding techniques: CCA and joint classifier and subspace learning.

6.7.1 Datasets and features

Datasets. We use the two following public benchmarks for evaluation. *INRIA Holidays*[4] [60] contains 1,491 images of 500 scenes and objects. One image per scene / object is used as query to search within the remaining 1,490 images and accuracy is measured as the Average Precision (AP) averaged over the 500 queries (mAP). The *University of Kentucky Benchmark* (UKB)[5] [90] contains 10,200 images of 2,550 objects. Each image is used in turn as query to search within the 10,200 images and accuracy is measured as 4×recall@4 averaged over the 10,200 queries. Hence, the maximum achievable score is 4 on this dataset.

We use the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) 2010 dataset[6] for learning purposes. We use it both for unsupervised learning (*e.g.* to learn a PCA) and for supervised learning (*e.g.* to learn a metric, attributes, CCA, etc.) This dataset contains 1,000 classes and consists of 3 sets: a training, a validation and a test set. In our experiments, we only make use of the training set which contains 1.2M images.

For the large-scale experiments reported in section 6.7.4, we also use a subset of 1M ImageNet images to serve as distractors. They were randomly sampled from the full ImageNet dataset [34] (excluding the ILSVRC 2010 categories)

[4] `http://lear.inrialpes.fr/~jegou/data.php`
[5] `http://www.vis.uky.edu/~stewe/ukbench/`
[6] `http://www.image-net.org/challenges/LSVRC/2010`

Table 6.1: Subspace learning as metric learning. Results on Holidays (mAP, in %) when learning with Instance Level (IL) and Category Level (CL) labels.

$R =$	16	32	64	128	256	512
PCA	**53.1**	61.3	**68.0**	72.3	75.0	76.8
IL	52.1	**62.9**	67.1	**73.2**	**75.8**	77.1
CL	36.8	54.2	65.1	68.9	75.4	**78.6**

Table 6.2: Subspace learning as metric learning. Results on UKB (4× recall@4) when learning with Instance Level (IL) and Category Level (CL) labels.

$R =$	16	32	64	128	256	512
PCA	**2.56**	**2.82**	**3.01**	**3.08**	**3.15**	**3.18**
IL	1.09	1.99	2.55	2.90	3.07	3.16
CL	1.80	2.37	2.78	2.95	3.09	3.16

Features. We extract 128-dimensional SIFT descriptors [82] and 96-dimensional color descriptors [99] on regular grids at multiple scales. Contrarily to most previous instance-level retrieval works, we do not make use of interest-point detectors. We found dense extraction to yield somewhat better (resp. worse) results on Holidays (resp. UKB). Note however that this saves the interest-point detection time which is substantial in our large-scale experiments. These features are reduced to 64 dimensions with PCA. We compute separately for each descriptor a 2,048-dimensional Fisher Vector (FV) which is power- and L2-normalized [97, 99]. The SIFT and color FVs are subsequently concatenated, thus yielding a 4,096-dim image descriptor. The distance between two FVs is computed with a dot-product [97]. This provides a baseline of 77.4% on Holidays and 3.19 on UKB.

6.7.2 Results with metric learning

When learning a metric for a target subspace dimension R, two parameters need to be tuned: the step size η as well as the number of iterations *niter*. We performed two sets of experiments. In the first set of experiments we learn the subspace from ILSVRC 2010. To avoid tuning η and *niter* on the test data, we validated our Holidays results on UKB and vice versa, *i.e.* we report results on Holidays (resp. UKB) with the parameters that lead to the best results on UKB (resp. Holidays). This shows the ability of the learning algorithm to generalize to new data. In the second set of experiments, we learn the subspace from instance-level labels. For the Holidays experiments, we therefore trained the subspace on UKB and vice-versa. In this set of experiments, η and *niter* were tuned directly to maximize test accuracy which gives an unfair advantage to this approach.

The metric learning results are reported in Tables 6.1 and 6.2 and compared to the PCA baseline. We can draw the two following conclusions. First, metric learning with instance-level labels (IL) does not significantly improve accuracy on Holidays or UKB. It is actually significantly worse than the PCA baseline on UKB. We believe this is because the training datasets (UKB for Holidays and Holidays for UKB) are too small to learn a meaningful subspace. Note that we are not aware of any significantly larger dataset with instance-level labels. Second, metric learning on category-level labels (CL) yields poor results, especially

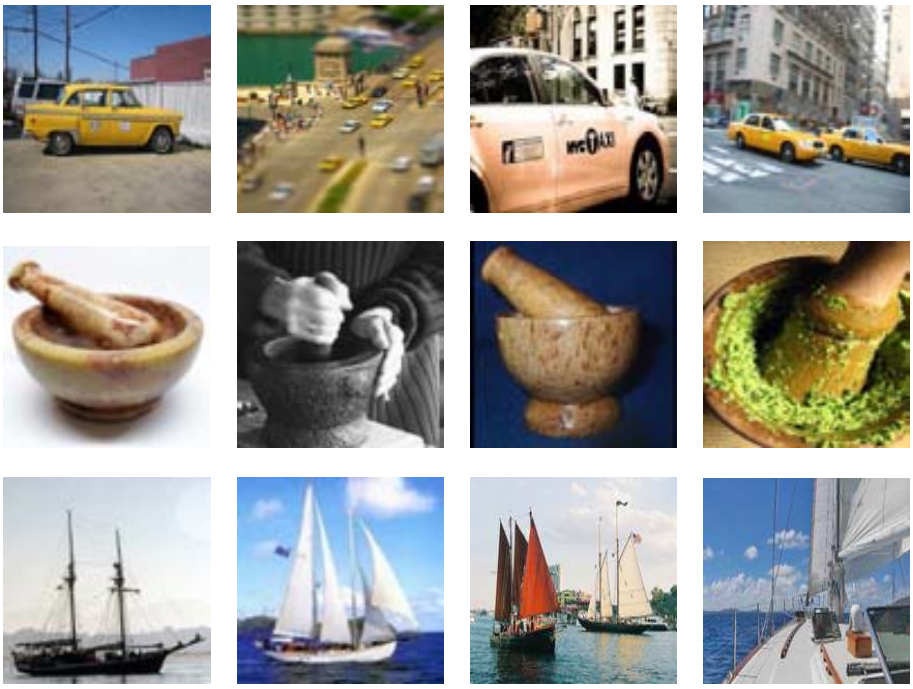


Figure 6.2: Random images from different ImageNet categories. First row: n02930766 ("cab, hack, taxi, taxicab"). Second row: n03786901 ("mortar"). Third row: n04147183 ("Schooner").

for a small number of dimensions R. We observe a small improvement with respect to the PCA baseline on Holidays for a larger R (*e.g.* $R = 512$). Our intuition to explain these poor results is the following one: although images within the same category might be visually dissimilar (*cf.* Fig 6.2), metric learning tries to enforce them explicitly to be closer to each other than to images in other categories.

6.7.3 Results with attributes

To learn the attribute classifiers, we first extract from the 1.2M ILSVRC 2010 training images the same 4,096-dimensional FV features we use for retrieval (*cf.* section 6.7.1). We then learn 1,000 one-vs-all binary linear SVMs using SGD[7]. Note that learning classifiers on FVs makes sense as shown for instance in [99]. Given an image, we construct its attribute vector by concatenating the 1,000 classifier scores, which yields a 1,000-dimensional vector. Hence, the computation of the attribute scores can be understood as a linear projection in a 1,000-dimensional subspace. The attribute vector is subsequently L2-normalized and we use the dot-product as a measure of similarity. Following [38], we also report results combining the FV and the attributes. As suggested by [38], we apply a weighting factor to increase the contribution of the FV. To avoid tuning this parameter on the test data, the optimal weight for Holidays (resp. UKB) was cross-validated on UKB (resp. Holidays). Results are reported

[7] `http://leon.bottou.org/projects/sgd`

Table 6.3: Combining FVs and attributes. Results on Holidays (mAP, in %) and UKB (4× recall@4).

	Holidays	UKB
FV (4,096 dim)	77.4%	3.19
Attr (1,000 dim)	76.2%	3.27
FV + Attr (5,096 dim)	78.1%	3.27

Table 6.4: Combining FVs and attributes after PCA. Results on Holidays (mAP, in %).

$R =$	16	32	64	128	256	512
FV	**53.1**	**61.3**	**68.0**	**72.3**	75.0	**76.8**
FV + Attr	49.3	60.3	66.4	71.2	**75.2**	**76.8**

in Table 6.3. We observe that attributes perform slightly worse than FVs on Holidays and slightly better on UKB. We also note that there seems to be little complementarity between FVs and attributes.

Since our focus is on subspace learning, we also perform dimensionality reduction by applying PCA to the FV and the attributes independently and by concatenating the resulting vectors, as suggested in [38]. To produce a signature of R dimensions, FVs and attributes are reduced to $R/2$ dimensions and concatenated. Again, we tune the relative weight of the FV part with respect to the attribute part. Table 6.4 compares this approach with the PCA baseline on the Holidays dataset and we observe no improvement.

These results somewhat contradict those of [38] who reported a significant improvement on Holidays when merging FVs and attributes. We believe this is because the features used by [38] to learn the attributes contained information not available in the FV. For instance, their attributes used, among others, color information, while their FVs were computed from SIFT descriptors only. To test this conjecture, we computed 2,048-dimensional FVs using only SIFT descriptors as well as 1,000-dimensional attribute vectors computed from color-only descriptors. Combining the FV and attributes in this case makes a significant difference on Holidays: from 68.5.% using SIFT FVs (2,048 dimensions) to 76.2% when concatenating SIFT FVs and color attributes (3,048 dimensions). We believe this experiment validates our point. Note that we can obtain a similar accuracy of 76.8% in a simple manner, by reducing the dimensionality of the 4,096-dimensional SIFT+color FV to 512 dimensions.

Our conclusion is therefore that attributes do not seem to improve instance-level retrieval significantly on these datasets.

6.7.4 Results with label-image embedding

We now report results for those two approaches which perform an embedding of images and labels in a common subspace: CCA and the proposed Joint Subspace and Classifier Learning (JSCL).

For both the CCA and JSCL, we use again ILSVRC 2010 for learning. For CCA, there is a single parameter to tune: the regularization parameter ρ (*cf*. section 6.5). As for JSCL, there are two parameters to tune (as was the case for metric learning): the step size η and

Table 6.5: Results of CCA and the proposed JSCL as compared to the PCA baseline on Holidays (mAP, in %).

$R =$	16	32	64	128	256	512
PCA	53.1	61.3	68.0	72.3	75.0	76.8
CCA	54.5	62.9	71.0	74.7	77.6	**79.0**
JSCL	**56.7**	**67.7**	**73.6**	**76.4**	**78.3**	78.9

Table 6.6: Results of CCA and the proposed JSCL as compared to the PCA baseline on UKB (4× recall@4).

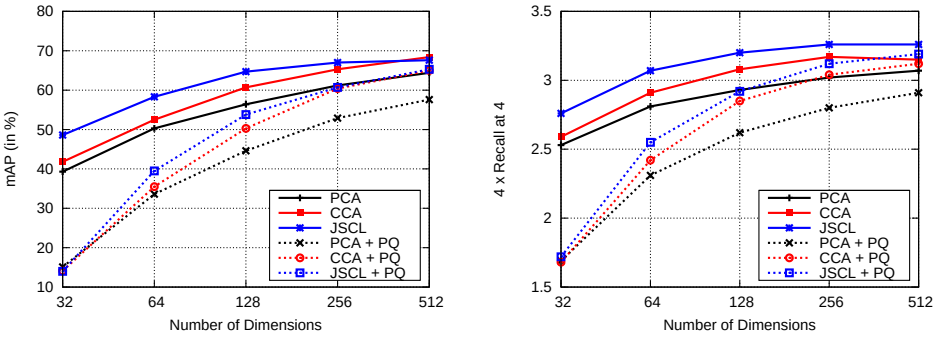
$R =$	16	32	64	128	256	512
PCA	2.56	2.82	3.01	3.08	3.15	3.18
CCA	2.52	2.90	3.11	3.22	3.29	3.32
JSCL	**2.67**	**3.04**	**3.23**	**3.31**	**3.36**	**3.36**

the number of iterations *niter*. As was the case in our previews experiments, to avoid tuning the parameters on the test data, we validate the Holidays (resp. UKB) parameters on UKB (resp. Holidays). We report results on Holidays in Table 6.5 and UKB in Table 6.6.

We can make the two following observations. First, both label-image embedding methods improve over the PCA baseline, especially for a small number of dimensions R of the subspace (*e.g.* $R = 32$). Second, JSCL generally yields better results than CCA. This seems to indicate that using a large margin framework enables to uncover a more discriminative subspace. On UKB, we point out that we can both reduce the dimensionality of the initial 4,096-dimensional FV representation down to 256 dimensions and increase the retrieval accuracy from 3.19 to 3.36.

Since our focus is on large-scale retrieval, we also performed an evaluation with a large set of distractor images as is common practice (see *e.g.* [60, 92, 64, 38]). In our experiments, we use 1M ImageNet images (*cf.* section 6.7.1). Hence, when running a search on Holidays (resp. UKB), the system compares the query to the 1,490 (resp. 10,200) database images + 1M distractors. We ran two sets of experiments. In the first set of experiments, the dimensionality of the FV is reduced (through PCA, CCA or JSCL) but no further compression is applied. In the second set of experiments, the dimensionality of the FV is reduced and a Product Quantization (PQ) [63] is further applied to encode the descriptors. We chose PQ since it yields state-of-the-art codes when combined with dimensionality reduction [64] but other encoding techniques could have been applied too. In a nutshell, PQ splits the large FV into small sub-vectors and applies a separate Vector Quantizer (VQ) to each subvector independently (see [63] for more details). In our experiments we use sub-vectors of 8 dimensions and each subvector is encoded on 8 bits. Hence, with such a configuration, if PQ takes as input a K-dimensional vector, it outputs a K bits code.

Results for Holidays and UKB are presented in Fig. 6.3. We can draw the following conclusions. As in the case of small-scale experiments, CCA and JSCL improve over PCA. Moreover, JSCL seems to have an edge over CCA. These observations are valid whether PQ encoding is applied or not. The differences with the PCA baseline seem more acute in large-scale experiments than in small-scale experiments. For instance, the retrieval accuracy is improved from 39.3% with PCA to 48.6% with JSCL with $R = 32$ (without compression)



(a) Holidays+1M (b) UKB+1M

Figure 6.3: Large-scale results of CCA and the proposed JSCL as compared to the PCA baseline. Left: Holidays + 1M distractors (mAP in %). Right: UKB + 1M distractors (4× recall@4).

on Holidays. This seems to indicate that learning good projections has a larger impact for more complex problems, *e.g.* when the relevant images are lost in a sea of irrelevant ones.

Finally, Figure 6.1 shows qualitative results on Holidays+1M using PQ codes of 128 bits. We show the top 5 results on 4 queries and compare the PCA and JSCL embeddings. The queries have been chosen such as that there is no intersection between the top PCA and JSCL results. We observe how the JSCL results are semantically more consistent than those of PCA, even if sometimes PCA finds true positives that JSCL misses, such as that of 6.1d.

6.8 Conclusion

At the beginning of this chapter, we raised the following question: can category-level labels be used to improve instance-level image retrieval? We can now answer this question positively. To reach this conclusion, we experimented with four learning techniques: the first one is based on a metric learning framework, the second one on attribute representations, the third one on Canonical Correlation Analysis (CCA) and the fourth one on Joint Subspace and Classifier Learning (JSCL). While the first three approaches had been applied to some extent to the image retrieval problem in the past, we believe we are the first to show the usefulness of JSCL in this context.

Our experimental evaluation showed that metric-learning and attributes do not improve significantly over the baseline system. In some cases, it can even lead to a decrease in accuracy. We also showed that CCA and JSCL, which both consist in embedding labels and images in a common subspace, can lead to substantial improvements, especially in large-scale experiments. Overall, JSCL yields the best results and we believe that it superiority with respect to the simpler CCA approach comes from the use of a large margin formulation.

Thus, a key conclusion of our work is that one might get a superior performance with a method such as JSCL which optimizes a categorization objective function (which is consistent with the category-level labels we use for training but which is only loosely consistent with our retrieval objective) than with a method such as metric learning which optimizes a retrieval objective function (which is consistent with our instance-level retrieval problem but which is inconsistent with our category-level labels).

We finally note that those methods which jointly embed labels and images in a common

subspace, such as CCA and JSCL, have an additional advantage which we have not exploited in this work. Indeed, since labels and images have a common representation, one could easily perform query-by-example and query-by-text searches within a unified framework.

Chapter 7

Conclusions

Automatic document image classification and retrieval has become a very important part of the incoming workflow of many businesses. Documents arrive to the "digital mailroom", where they are scanned, analyzed, and dispatched to the appropriate workflows in a mostly unsupervised way. Meanwhile, the amount of documents that these digital mailrooms have to deal with has been steadily increasing during the last few years, making the task more and more complex as time goes by. Surprisingly, despite the importance of this task, research works that deal with document image representation, classification, and retrieval are not common in the literature. Works that also consider the large-scale scalability problems are extremely scarce.

Through this thesis we have studied this particular problem: the document image representation aimed at classification and retrieval tasks, focusing on realistic large scale domains. One of the consequences of working in large-scale domains is that the methods need to be *general*, *sound*, and *scalable*. However, as noted in the related work review of chapter 2, most available methods do not satisfy all of these conditions. Most methods rely on a particular structure of the document (*i.e.*, a method used to represents forms would not be useful for representing ID cards), are sensitive to noise, or are slow to produce, classify, or retrieve. Most of these methods had only been evaluated on in-house datasets, usually containing only a few tens or hundreds of images. Motivated by this we proposed a method based on runlength histograms that was general and sound, producing state-of-the-art results on public and in-house datasets of different nature in retrieval and classification tasks. In chapter 4, we evaluated our descriptor in a dataset containing approximately 40,000 documents and 181 classes. Compared to other datasets typically used by the document analysis comunity, this is at least one order of magnitude more documents and classes.

These descriptors were later extended to deal with other problems. In chapter 3, we approached the problem of unordered multiple-page document representation. As far as we know, we are the first to propose a method to classify and retrieve such document images. This is extremely surprising, considering the importance of this problem for companies dealing with a significant volume of incoming documents. We presented a simple Bag of Pages approach to encode the documents, that was later improved with a better representation of the pages – an attributes-based representation of the page runlengths – and a better encoding method – the Fisher Vector –, and significantly improved the baseline retrieval and classification results. We evaluated on datasets containing up to 20,000 documents and 60,000 images, which, once again, is a significantly larger number of documents than those used by other works.

In chapter 4, the descriptors were also enriched, through Canonical Correlation Analysis, to encode information not available during testing time. In particular, we showed that the visual descriptors we used could be enriched with textual information, significantly improving their performance in classification and retrieval tasks. Although we only experimented with text, this could be useful for other discriminative features that are expensive to obtain at testing time, for example, information about checkboxes.

Although we used "large" datasets through many of our experiments, we should note that these are not representative of the huge amount of incoming documents that can be received by large businesses. As presented, our signatures would not scale well enough. Although computing them is fast enough, maintaining them in RAM memory for rapid access in retrieval tasks can still be an issue. Motivated by this, chapter 5 studied the use of compression techniques such as binarization, and proposed the use of asymmetric distances for many popular binary embedding techniques, which can substantially improve the results at a minimum cost. These methods were evaluated on many publicly-available natural image datasets, including some large-scale ones, and were later tested on the same document datasets that were used through previous chapters, showing that their utility is by no means restricted to natural images.

For retrieval tasks such as this one, it is important to note that using supervised information to "learn to rank" is a key aspect of well performing systems. This was already noted in chapter 2, where the use of metric learning dramatically improved the results of a retrieval task when we could use some labeled documents for training purposes. Through chapter 6 we studied several techniques for supervised learning using category-level annotated images which led to significant improvements in instance-level retrieval tasks. Interestingly, these techniques can perform, either explicitly or implicitly, a dimensionality reduction of the data. This is relevant because it relates to the binarization methods of chapter 5: the studied binary encoding methods make use of unsupervised dimensionality reduction, *i.e.* PCA, which can be seamlessly replaced with the supervised dimensionality reduction learned in chapter 6. Although experiments were only carried on in natural image datasets, this framework can also be useful in our digital mailroom scenario, as already hinted in chapter 2: category-level annotated documents available for training can be used to learn a better subspace, where, given a new query, similar instances will be ranked first.

7.1 Future Work

Finally, we provide some possible lines of future work for both parts of this thesis, document representation and large-scale retrieval.

Document Representation:

In chapter 2 we introduced a document image descriptor based on runlength histograms. One of the less attractive parts of the approach is the need to manually tune the quantization intervals as well as the pyramid structure. Although we showed that the influence of these parameters in the final accuracy is small, it would be desirable to automatically learn these parameters in a discriminative way. For example, when dealing with spatial pyramids, some works have proposed to automatically learn the pyramid structure to maximize the classification accuracy [112]. In some cases, the methods go beyond spatial pyramids and encode the spatial distributions in different ways, such as using Fisher Vectors over the distribution of the features' positions [70]. In [114], Simonyan *et al.* discriminatively learn SIFT-like low-level descriptors by considering a large number of pooling regions and learning

the appropriate weights of every region while enforcing a sparsity norm on the weights vector. In this way they obtain a weighting of the regions while at the same time the sparsity norm ensures that only the relevant regions are taken into account. We believe some of these techniques could be adapted to discriminatively learn the structure as well as the quantization intervals of our descriptor.

In chapter 3 we presented a Bag of Pages approach to multiple-page document representation, as well as several improvements. We note, however, that our Bag of Pages approaches – with and without FV encoding – make one important assumption, namely that the pages of a document are independently and identically distributed (i.i.d.). However, although extremely common, this is not a very reasonable assumption in general, and particularly not on the documents domain. For example, a form page included in the document may request a photo ID. In that case, if such a page is found, it is likely than one of the following pages drawn from the document will be an ID.

We believe that our representation could benefit from more complex topic models such as Latent Dirichlet Allocation (LDA) [14], which could help to capture some correlations between the pages drawn from documents. In [25], the authors propose a latent GMM model over which Fisher Vectors are computed. This model no longer assumes that the image patches (pages in our case) are i.i.d. distributed. The authors also propose a Fisher Vector representation of an LDA model that captures the co-occurrence statistics. We believe using such models could help to improve our representations.

In chapter 4 we used CCA to find a common subspace between visual (cheap) and textual (expensive) representations that could be used to improve the cheap representation of the documents with no need to access the expensive representation at testing time. We believe, however, that CCA is not fully exploiting the potential of the available expensive views. Although the accuracy improvement after CCA is very significant, it is far from the results that can be obtained using the textual, expensive view. One of the reason is probably related to how different both views are: the visual view is a relatively dense histogram with some structural information, while the textual view is a very sparse bag of words representation of the whole page. Finding a linear correlation between so different views seems a difficult task, particularly since none of the views is very high-dimensional.

Based on this, it is reasonable to expect that implicitly projecting the data in a higher dimensional space through KCCA (using, for example, a Gaussian kernel) could significantly improve the results. Unfortunately, applying KCCA directly is not very efficient, since it would require one to encode every new sample at test time as a function of all the training samples. Under the constraints that we imposed in chapter 1, we may not be able to afford this transformation. A possible solution to this problem was proposed by Gong *et al.* in the very recent [47], where, instead of implicitly projecting the data using KCCA, the data was *explicitly* projected using an approximate explicit embedding [101], and standard CCA was applied in that projected space. Gong *et al.* obtained excellent results using this approximate explicit embedding + linear CCA approach, and we believe that that formulation could also significantly improve our approach at a negligible cost.

On a different note, we find important to remark that, although CCA proved itself very useful to learn a more meaningful subspace, the CCA projections were not learned with a discriminative goal in mind. In [131, 132], Weston *et al.* showed that it is possible to discriminatively learn, at the same time, a projection into a low-dimensional space and a classifier in this subspace. The projections and the classifier can be learned efficiently by means of Stochastic Gradient Descent. We believe that we could adopt a similar framework, where we learn the projections of the cheap and expensive views and the classifier of the projected cheap view in a discriminative way, while at the same time we regularize the objective function by enforcing the projected cheap and expensive views to be close together;

Instead of finding a common subspace between the cheap and expensive views and learn a classifier in that subspace, we directly find a subspace and classifier for the cheap view, while using the expensive view to "guide" the projections.

Large-Scale retrieval:

In chapter 5 we proposed two asymmetric distances between binary datasets and non-binary queries that significantly improved the retrieval accuracy at a minimum extra cost. These techniques could be applied to a large variety of binary encoding methods, but we note that these encoding methods were not designed with the asymmetric distances in mind. In future work, we would be interested in investigating coding techniques which would be designed with the asymmetric distances in mind. One possible example which was inspired to us by ITQ would be to learn a rotation matrix which does not minimize the quantization error between the signatures and the binary codes, but between the signatures and the *reconstructed* version of the binary codes using the expected values.

Also, in the asymmetric expectation-based approach, we currently learn the α coefficients (Equations (5.22) and (5.23)) which minimize a reconstruction error on the training data. It would be interesting to understand whether we could learn these coefficients with supervised data to optimize directly the retrieval objective function.

Finally, in chapter 6, we showed how we can exploit category-labeled data to perform instance-level retrieval. Although we obtained very significant improvements, it is reasonable to expect that, if we were able to collect as much instance-level annotated data as we have category-level annotated data, learning on the instance-level data could yield even better results for the instance-level retrieval task. Unfortunately, such amounts of instance-level annotated data are not available. We would like to explore some options, such as creating noisy instance-level datasets. For example, we could use ImageNet to retrieve the top k most similar neighbors of any image *inside* its class, and consider those the relevant items during training. In that way, we guarantee that the relevant items are semantically similar, but also visually similar, and can be used as noisy instance-level data.

Publications

The following publication are a consequence of the research carried out during the elaboration of this thesis as well as related works, and give an idea of the progression that has been achieved.

Journals

- 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. To appear.*
- A. Fornés, A. Dutta, A. Gordo and J. Lladós. CVC-MUSCIMA: A Groundtruth of Handwritten Music Score Images for Writer Identification and Staff Removal. *International Journal on Document Analysis and Recognition.* 2011.
- A. Gordo, A. Fornés and E. Valveny. Writer Identification in Handwritten Musical Scores with Bags of Notes. *Pattern Recognition.* 2012.

International Conferences and Workshops

- J. Almazán, A. Gordo, A. Fornés, E. Valveny. Efficient Exemplar Word Spotting. In *British Machine Vision Conference.* 2012.
- A. Gordo, J. A. Rodriguez-Serrano, F. Perronnin and E. Valveny. Leveraging Category-Level Labels for Instance-Level Image Retrieval. In *Computer Vision & Pattern Recognition.* 2012.
- A. Gordo, F. Perronnin and E. Valveny. Document Classification Using Multiple Views. In *Document Analysis Systems.* 2012. **[Best paper award]**
- A. Gordo and F. Perronnin. Asymmetric Distances for Binary Embeddings. In *Computer Vision & Pattern Recognition.* 2011.
- A. Fornés, A. Dutta, A. Gordo and J. Lladós. The ICDAR 2011 Music Scores Competition: Staff Removal and Writer Identification. In *International Conference on Document Analysis and Recognition.* 2011.
- A. Gordo, A. Fornés, E. Valveny and J. Lladós. A Bag of Notes Approach to Writer Identification in Old Handwritten Musical Scores. In *Document Analysis Systems.* 2010.
- A. Gordo, J. Gibert, E. Valveny and M. Rusiñol. A Kernel-based Approach to Document Retrieval. In *Document Analysis Systems.* 2010.

- A. Gordo and F. Perronnin. A Bag of Pages Approach to Unordered Multipage Document Classification. In *International Conference on Pattern Recognition*. 2010.
- O. Ramos, N. Serrano, A. Gordo, E. Valveny and A. Juan. Interactive-predictive Detection of Handwritten Text Blocks. In *Document Recognition and Retrieval*. 2010.
- A. Gordo and E. Valveny. A Rotation Invariant Page Layout Descriptor for Document Classification and Retrieval. In *International Conference on Document Analysis and Recognition*. 2009.

Submitted Journals

- A. Gordo, F. Perronnin and E. Valveny. Large-Scale Document Image Retrieval and Classification with Runlength Histograms and Binary Embeddings. *Pattern Recognition*. 2012.
- A. Gordo, F. Perronnin, Y. Gong and S. Lazebnik. Asymmetric Distances for Binary Embeddings. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2012.

Bibliography

[1] The Medical Article Records Groundtruth Dataset. http://marg.nlm.nih.gov/roverintro.asp.

[2] The NIST Structured Forms Database (NIST Special Database 2). http://www.nist.gov/ts/msd/srd/nistsd2.cfm.

[3] Document separation and form identification. Technical report, KOFAX, 2005.

[4] Classification and separation. Technical report, KOFAX, 2008.

[5] Implementing a digital mailroom - a white paper discussing the advantages of digital mailrooms. Technical report, Datafinity Ltd., 2009.

[6] M. Agrawal and D. Doermann. Voronoi++: A dynamic page segmentation approach based on voronoi and docstrum features. In *International Conference on Document Analysis and Recognition*, 2009.

[7] A. Bagdanov and M. Worring. First Order Gaussian Graphs for efficient structure classification. *Pattern Recognition*, 2003.

[8] A. Bagdanov and M. Worring. Multiscale document description using rectangular granulometries. *International Journal on Document Analysis and Recognition*, 2003.

[9] B. Bai, J. Weston, D. Grangier, R. Collobert, K. Sadamasa, Y. Qi, O. Chapelle, and K. Weinberger. Supervised semantic indexing. In *ACM Conference on Information and Knowledge Management*, 2009.

[10] H. Baya, A. Essa, T. Tuytelaarsb, and L. V. Gool. Speeded-Up Robust Features (SURF). *Computer Vision and Image Understanding*, 2008.

[11] A. Behera, D. Lalanne, and R. Ingold. Combining color and layout features for the identification of low-resolution documents. *International Journal of Signal Processing*, 2005.

[12] A. Bergamo, L. Torresani, and A. Fitzgibbon. PiCoDes: Learning a compact code for novel-category recognition. In *Neural Information Processing Systems*, 2011.

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

[14] D. Blei, A. Ng, and M. Jordan. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 2003.

[15] M. Borga. Canonical correlation: a tutorial, 2001. http://www.imt.liu.se/~magnus/cca/.

[16] L. Bottou. Stochastic learning. In *Advanced Lectures on Machine Learning*, 2003.

[17] J. Brandt. Transform coding for fast approximate nearest neighbor search in high dimensions. In *IEEE Conference on Computer Vision & Pattern Recognition*, 2010.

[18] S. Branson, C. Wah, F. Schroff, B. Babenko, P. Welinder, P. Perona, and S. Belongie. Visual recognition with humans in the loop. In *European Conference on Computer Vision*, 2010.

[19] F. Cesarini, M. Lastri, S. Marinai, and G. Soda. Encoding of modified x-y trees for document classification. In *International Conference on Document Analysis and Recognition*, 2001.

[20] M. Charikar. Similarity estimation techniques from rounding algorithms. In *ACM Symposium on Theory of Computing*, 2002.

[21] 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.

[22] G. Chechik, U. Shalit, V. Sharma, and S. Bengio. An online algorithm for large scale image similarity learning. In *Neural Information Processing Systems*, 2009.

[23] N. Chen and D. Blostein. A survey of document image classification: problem statement, classifier architecture and performance evaluation. *International Journal of Document Analysis and Recognition*, 2006.

[24] S. Chen, S. Mao, and G. Thoma. Simultaneous layout style and logical entity recognition in a heterogeneous collection of documents. In *International Conference on Document Analysis and Recognition*, 2007.

[25] R. G. Cinbis, J. Verbeek, and C. Schmid. Image categorization using Fisher kernels of non-iid image models. In *IEEE Conference on Computer Vision & Pattern Recognition*, 2012.

[26] K. Collins-Thompson and R. Nickolov. A clustering-based algorithm for automatic document separation. In *SIGIR Workshop on Information Retrieval and OCR*, 2002.

[27] G. Csurka, C. R. Dance, L. Fan, J. Willamowski, and C. Bray. Visual categorization with bags of keypoints. In *Workshop on Statistical Learning in Computer Vision, ECCV*, 2004.

[28] J. F. Cullen, J. J. Hull, and P. E. Hart. Document image database retrieval and browsing using texture analysis. In *International Conference on Document Analysis and Recognition*, 1997.

[29] J. V. Davis, B. Kulis, P. Jain, S. Sra, and I. S. Dhillon. Information-theoretic metric learning. In *International Conference on Machine Learning*, 2007.

[30] F. de la Torre. A least-squares framework for component analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2011.

[31] F. de la Torre and M. J. Black. A framework for robust subspace learning. *International Journal of Computer Vision*, 2003.

[32] F. Deckert, B. Seidler, M. Ebbecke, and M. Gillmann. Table content understanding in smartfix. In *International Conference on Document Analysis and Recognition*, 2011.

[33] J. Deng, A. Berg, and L. Fei-Fei. Hierarchical semantic indexing for large scale image retrieval. In *IEEE Conference on Computer Vision & Pattern Recognition*, 2011.

[34] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision & Pattern Recognition*, 2009.

[35] T. Deselaers and V. Ferrari. Visual and semantic similarity in ImageNet. In *IEEE Conference on Computer Vision & Pattern Recognition*, 2011.

[36] D. Doermann. The indexing and retrieval of document images: A survey. *Computer Vision and Image Understanding*, 1998.

[37] W. Dong, M. Charikar, and K. Li. Asymmetric distance estimation with sketches for similarity search in high-dimensional spaces. In *ACM Special Interest Group on Information Retrieval*, 2008.

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

[39] F. Esposito, D. Malerba, and F. A. List. Machine learning for intelligent processing of printed documents. *Journal of Intelligent Information Systems*, 2000.

[40] F. Esposito, D. Malerba, and G. Semeraro. Multistrategy learning for document recognition. *Applied Artificial Intelligence*, 1994.

[41] M. Everingham, L. V. Gool, C. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (VOC) challenge. *International Journal of Computer Vision*, 2010.

[42] J. Farquhar, S. Szedmak, H. Meng, and J. Shawe-Taylor. Improving "bag-of-keypoints" image categorisation: Generative models and pdf-kernels. Technical report, University of Southampton, 2005.

[43] L. Fei-Fei, R. Fergus, and P. Perona. One-shot learning of object categories. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2006.

[44] J. Fisher, S. Hinds, and D. D'Amato. A rule-based system for document image segmentation. In *International Conference on Pattern Reognition*, 1990.

[45] P. Frasconi, G. Soda, and A. Vullo. Hidden markov models for text categorization in multi-page documents. *Journal of Intelligent Information Systems*, 2002.

[46] Y. Gong and S. Lazebnik. Iterative quantization: A procrustean approach to learning binary codes. In *IEEE Conference on Computer Vision & Pattern Recognition*, 2011.

[47] 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, [to appear]*, 2012.

[48] A. Gordo, J. Gibert, E. Valveny, and M. Rusiñol. A kernel-based approach to document retrieval. In *International Workshop on Document Analysis Systems*, 2010.

[49] A. Gordo and F. Perronnin. A bag-of-pages approach to unordered multi-page document classification. In *International Conference on Pattern Recognition*, 2010.

[50] A. Gordo and F. Perronnin. Asymmetric distances for binary embeddings. In *IEEE Conference on Computer Vision & Pattern Recognition*, 2011.

[51] A. Gordo and E. Valveny. A rotation invariant page layout descriptor for document classification and retrieval. In *International Conference on Document Analysis and Recognition*, 2009.

[52] G. Griffin, A. Holub, and P. Perona. Caltech-256 object category dataset. Technical report, California Institute of Technology, 2007.

[53] 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.

[54] P. Heroux, S. Diana, A. Ribert, and E. Trupin. Classification method study for automatic form class identification. In *International Conference on Pattern Reognition*, 1998.

[55] H. Hotelling. Relations between two sets of variates. *Biometrika*, 1936.

[56] P. Indyk and R. Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *ACM Symposium on Theory of Computing*, 1998.

[57] D. J. Ittner, D. D. Lewis, and D. D. Ahn. Text categorization of low quality images. In *Symposium on Document Analysis and Information Retrieval*, 1995.

[58] M. Iwamura, T. Tsuji, and K. Kise. Memory-based recognition of camera-captured characters. In *International Workshop on Document Analysis Systems*, 2010.

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

[60] H. Jégou, M. Douze, and C. Schmid. Hamming embedding and weak geometry consistency for large scale image search. In *European Conference on Computer Vision*, 2008.

[61] H. Jégou, M. Douze, and C. Schmid. On the burstiness of visual elements. In *IEEE Conference on Computer Vision & Pattern Recognition*, 2009.

[62] H. Jégou, M. Douze, and C. Schmid. Packing bag-of-features. In *International Conference on Computer Vision*, 2009.

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

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

[65] T. Joachims. Optimizing search engines using clickthrough data. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2002.

[66] D. Keysers, T. Deselaers, and H. Ney. Pixel-to-pixel matching for image recognition using hungarian graph matching. In *Symposium of the German Association for Pattern Recognition*, 2004.

[67] D. Keysers, F. Shafait, and T. M. Breuel. Document image zone classification - a simple high-performance approach. In *International Conference on Computer Vision Theory and Applications*, 2007.

[68] F. S. Khan, R. M. Anwer, J. van de Weijer, A. D. Bagdanov, M. Vanrell, and A. M. Lopez. Color attributes for object detection. In *IEEE Conference on Computer Vision & Pattern Recognition*, 2012.

[69] F. S. Khan, J. van de Weijer, A. D. Bagdanov, and M. Vanrell. Portmanteau vocabularies for multi-cue image representation. In *Neural Information Processing Systems*, 2011.

[70] J. Krapac, J. Verbeek, and F. Jurie. Modeling spatial layout with fisher vectors for image categorization. In *International Conference on Computer Vision*, 2011.

[71] A. Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

[72] H. W. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 1955.

[73] B. Kulis and K. Grauman. Kernelized locality-sensitive hashing for scalable image search. In *International Conference on Computer Vision*, 2009.

[74] B. Kulis, K. Saenko, and T. Darrell. What you saw is not what you get: Domain adaptation using asymmetric kernel transforms. In *IEEE Conference on Computer Vision & Pattern Recognition*, 2011.

[75] C. Lampert and O. Kromer. Weakly-paired maximum covariance analysis for multimodal dimensionality reduction and transfer learning. In *European Conference on Computer Vision*, 2010.

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

[77] G. Lanckriet, N. Cristianini, L. E. Ghaoui, P. Bartlett, and M. Jordan. Learning the kernel matrix with semi-definite programming. *Journal of Machine Learning Research*, 2004.

[78] 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 & Pattern Recognition*, 2006.

[79] J. Li, Z.-G. Fan, Y. Wu, and N. Le. Document image retrieval with local feature sequences. In *International Conference on Document Analysis and Recognition*, 2009.

[80] J. Liang, D. Doermann, M. Ma, and J. Guo. Page classification through logical labelling. In *International Conference on Pattern Reognition*, 2002.

[81] L. Likforman-Sulem, P. Vaillant, and F. Yvon. Proper names extraction from fax images combining textual and image features. In *International Conference on Document Analysis and Recognition*, 2003.

[82] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 2004.

[83] J. Lu, G. Wang, and Y.-P. Tan. Multilinear locality preserving canonical correlation analysis for face recognition. In *International Conference on Information, Communications and Signal Processing*, 2011.

[84] S. Marinai, E. Marino, and G. Soda. Layout based document image retrieval by means of X-Y tree reduction. In *International Conference on Document Analysis and Recognition*, 2005.

[85] T. Mensink, J. Verbeek, and G. Csurka. Learning structured prediction models for interactive image labeling. In *IEEE Conference on Computer Vision & Pattern Recognition*, 2011.

[86] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. V. Gool. A comparison of affine region detectors. *International Journal of Computer Vision*, 2005.

[87] G. Nagy. Twenty years of document analysis in PAMI. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2000.

[88] G. Nagy and S. Seth. Hierarchical representation of optically scanned documents. In *International Conference on Pattern Recognition*, 1984.

[89] M.-E. Nilsback and A. Zisserman. A visual vocabulary for flower classification. In *IEEE Conference on Computer Vision & Pattern Recognition*, 2006.

[90] D. Nistér and H. Stewénius. Scalable recognition with a vocabulary tree. In *IEEE Conference on Computer Vision & Pattern Recognition*, 2006.

[91] A. Oliva and A. Torralba. Modeling the shape of the scene: a holistic representation of the spatial envelope. *International Journal of Computer Vision*, 2001.

[92] M. Perdóch, O. Chum, and J. Matas. Efficient representation of local geometry for large scale object retrieval. In *IEEE Conference on Computer Vision & Pattern Recognition*, 2009.

[93] I. Perea and D. López. Syntactic modeling and recognition of document images. In *International Workshop on Structural, Syntactic, and Statistical Pattern Recognition*, 2004.

[94] F. Perronnin. Universal and adapted vocabularies for generic visual categorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2007.

[95] F. Perronnin, Z. Akata, Z. Harchaoui, and C. Schmid. Towards good practice in large-scale learning for image classification. In *IEEE Conference on Computer Vision & Pattern Recognition*, 2012.

[96] F. Perronnin and C. Dance. Fisher kernels on visual vocabularies for image categorization. In *IEEE Conference on Computer Vision & Pattern Recognition*, 2007.

[97] F. Perronnin, Y. Liu, J. Sánchez, and H. Poirier. Large-scale image retrieval with compressed fisher vectors. In *IEEE Conference on Computer Vision & Pattern Recognition*, 2010.

[98] F. Perronnin, J. Sánchez, and Y. Liu. Large-scale image categorization with explicit data embedding. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

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

[100] M. Raginsky and S. Lazebnik. Locality-sensitive binary codes from shift-invariant kernels. In *Neural Information Processing Systems*, 2009.

[101] A. Rahimi and B. Recht. Random features for large-scale kernel machines. In *Neural Information Processing Systems*, 2007.

[102] N. Rasiwasia, P. J. Moreno, and N. Vasconcelos. Bridging the gap: Query by semantic example. *IEEE Transactions on Multimedia*, 2007.

[103] M. Rastegari, C. Fang, and L. Torresani. Scalable object-class retrieval with approximate and top-k ranking. In *International Conference on Computer Vision*, 2011.

[104] Y. Rubner, L. Guibas, and C. Tomassi. The earth mover's distance, multi-dimensional scaling , and color-based image retrieval. In *ARPA Image Understanding Workshop*, 1997.

[105] G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. *Information Processing Management*, 1988.

[106] G. Salton and M. J. McGill. *Introduction to Modern Information Retrieval.* McGraw-Hill, 1983.

[107] P. Sarkar. Image classification: Classifying distributions of visual features. In *International Conference on Pattern Recognition*, 2006.

[108] E. Saund. A graph lattice approach to maintaining dense collections of subgraphs as image features. In *International Conference on Document Analysis and Recognition*, 2011.

[109] M. A. R. Schmidtler and J. W. Amtrup. Automatic document separation: A combination of probabilistic classification and finite-state sequence modeling. In *Natural Language Processing and Text Mining*. Springer, 2007.

[110] P. Schonemann. A generalized solution of the orthogonal procrustres problem. *Psychometrika*, 1966.

[111] F. Sebastiani. A tutorial on automated text categorisation. In *Argentinian Symposium on Artificial Intelligence*, 1999.

[112] G. Sharma and F. Jurie. Learning discriminative spatial representation for image classification. In *British Machine Vision Conference*, 2011.

[113] C. Shin, D. Doermann, and A. Rosenfeld. Classification of document pages using structure-based features. *International Journal on Document Analysis and Recognition*, 2001.

[114] K. Simonyan, A. Vedaldi, and A. Zisserman. Descriptor learning using convex optimisation. In *European Conference on Computer Vision*, 2012.

[115] J. Sivic and A. Zisserman. Video google: A text retrieval approach to object matching in videos. In *ICCV*, 2003.

[116] S. Sonnenbur, G. Rätsc, C. Schäfe, and B. Schölkop. Large scale multiple kernel learning. *Journal of Machine Learning Research*, 2006.

[117] A. L. Spitz and A. Maghbouleh. Text categorization using character shape codes. In *Symposium on Electronic Imaging Science and Technology*, 1999.

[118] N. Stamatopoulos, B. Gatos, and T. Georgiou. Page frame detection for double page document images. In *International Workshop on Document Analysis Systems*, 2010.

[119] X. Tang. Texture information in run-length matrices. *IEEE Transactions on Image Processing*, 1998.

[120] R. Tibshirani and G. Hinton. "Coaching" variables for regression and classification. *Statistics and Computing*, 1995.

[121] A. Torralba, R. Fergus, and W. Freeman. 80 million tiny images: a large dataset for non-parametric object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2008.

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

[123] J. van Beusekom. Document layout analysis. Diploma thesis, Technische Universitt Kaiserslautern, 2006.

[124] J. van Beusekom, D. Keysers, F. Shafait, and T. Breuel. Distance measures for layout-based document image retrieval. In *International Conference on Document Image Analysis for Libraries*, 2006.

[125] A. Vedaldi and A. Zisserman. Efficient additive kernels via explicit feature maps. In *IEEE Conference on Computer Vision & Pattern Recognition*, 2010.

[126] G. Wang, D. Hoiem, and D. Forsyth. Learning image similarity from flickr groups using stochastic intersection kernel machines. In *International Conference on Computer Vision*, 2009.

[127] J. Wang, S. Kumar, and S.-F. Chang. Semi-supervised hashing for large scale search. In *IEEE Conference on Computer Vision & Pattern Recognition*, 2010.

[128] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong. Locality-constrained linear coding for image classification. In *IEEE Conference on Computer Vision & Pattern Recognition*, 2010.

[129] K. Weinberger and L. Saul. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 2009.

[130] Y. Weiss, A. Torralba, and R. Fergus. Spectral hashing. In *Neural Information Processing Systems*, 2008.

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

[132] J. Weston, S. Bengio, and N. Usunier. WSABIE: Scaling up to large vocabulary image annotation. In *International Joint Conference on Artificial Intelligence*, 2011.

[133] I. Witten, A. Moffat, and T. Bell. *Managing Gigabytes: compressing and indexing documents and images*. Morgan Kaufmann Publishers, 1999.

[134] K. Y. Wong, R. G. Casey, and F. M. Wahl. Document analysis system. *IBM Journal of Research and Development*, 1982.

[135] X. Zhou, K. Yu, T. Zhang, and T. S. Huan. Image classification using super-vector coding of local image descriptors. In *European Conference on Computer Vision*, 2010.

[136] J. Zobel and A. Moffat. Exploring the similarity space. In *ACM SIGIR Forum*, 1998.