Interactive Object Retrieval using Interpretable Visual Models

Ahmed-Riadh Rebai

UNIVERSITÉ PARIS-SUD 11
Faculté des sciences d'Orsay
N° Ordre: 2011PA112054

P H D T H E S I S

Interactive Object Retrieval using Interpretable Visual Models

Submitted for the degree of "docteur en sciences"
of the University Paris-Sud 11
Speciality: Computer Science

By

Ahmed Rebai

May 2011

INRIA Paris-Rocquencourt, Imedia Team

Thesis committee:

Reviewers:	Fred Stentiford	-	Prof. at University College London (UK)
	Sylvie Philipp-Foliguet	-	Prof. at Université Cergy/Pontoise (FR)
Director:	Nozha Boujemaa	-	Director of the INRIA-Saclay Center (FR)
Advisor:	Alexis Joly	-	Researcher at INRIA-Rocquencourt (FR)
Examinator:	Michel Crucianu	-	Prof. at CNAM (FR)
President:	François Yvon	-	Prof. at Université Paris-Sud 11 (FR)

Abstract: This thesis is an attempt to improve visual object retrieval by allowing users to interact with the system. Thanks to the advances in technology, content-based image retrieval has gained greater maturity over the last few years, and there have been a number of improvements in terms of image description and large scale real-time efficient search. However, current search engines haven't yet reached the point where they are able to correctly formulate and answer a user's mental query. This is the case, for example, when we wish to retrieve images that contain an object with specific characteristics. Unfortunately, and in practice, the results returned by state-of-the-art visual concept detectors are often difficult to interpret from a user's point of view. The visual models produced by statistical machine learning methods are indeed highly dependent on the training data and might convey a different semantic than the words used to describe the originally targeted concept. This often makes users uncomfortable with these technologies since they do not get what they expected from the textual description of the trained concept. It could well be of interest, therefore, to build systems that retrieve these concepts according to human perception. Our solution lies in constructing an interactive system that allows users to define their own visual concept from a concise set of visual patches given as input. These patches—which represent the most informative clues of a given visual category—are trained beforehand with a supervised learning algorithm in a discriminative manner. Then, and in order to specialize their models, users have the possibility to send their feedback on the model itself by choosing and weighting the patches they are confident of.

The real challenge consists in how to generate concise and visually interpretable models. Our contribution relies on two points. First, in contrast to the state-of-the-art approaches that use bag-of-words, we propose embedding local visual features without any quantization, which means that each component of the high-dimensional feature vectors used to describe an image is associated to a unique and precisely localized image patch. Second, we suggest using regularization constraints in the loss function of our classifier to favor sparsity in the models produced. Sparsity is indeed preferable for concision (a reduced number of patches in the model) as well as for decreasing prediction time. To meet these objectives, we developed a multiple-instance learning scheme using a modified version of the BLasso algorithm. BLasso is a boosting-like procedure that behaves in the same way as Lasso (Least Absolute Shrinkage and Selection Operator). It efficiently regularizes the loss function with an additive L_1-constraint by alternating between forward and backward steps at each iteration. The method we propose here is generic in the

sense that it can be used with any local features or feature sets representing the content of an image region. In addition, we extended the initial version (using single image features to describe image patches) to a geometrically consistent version using feature sets as patch descriptors. Quantitatively, our method achieves similar performances as current state-of-the-art systems but outperforms them when training very small objects in highly cluttered images. Qualitatively, the interpretability allows users to construct their own model from the original set of learned patches, thus allowing for more compound semantic queries. Furthermore, we developed a GUI that allows interaction with users and we showed how results might be improved. Finally, it is worth mentioning that our method can be applied to various multimedia sources including text, audio and video documents.

Résumé : L'objectif de cette thèse est d'améliorer la recherche d'objets visuels à l'aide de l'interactivité avec l'utilisateur. Grâce aux avancés technologiques durant les dernières années, la recherche d'images par le contenu a gagné davantage en maturité. En effet, il y a eu beaucoup d'amélioration concernant la description des images et la recherche à large échelle en temps-réel. Toutefois, les moteurs de recherche actuels n'ont pas encore atteint le point où ils peuvent facilement permettre de formuler et de répondre à une requête mentale de l'utilisateur. Il s'agit ici de chercher des objets avec des caractéristiques spécifiques. En pratique, les résultats retournés par les détecteurs de concepts visuels état-de-l'art sont souvent difficiles à interpréter de point de vue utilisateur. Les modèles visuels produits sont en effet fortement liés aux données d'apprentissage et peuvent par la suite apporter une sémantique différente des mots qui ont été utilisés pour décrire le concept visuel d'origine. Par conséquent, les utilisateurs sont souvent insatisfaits de cette technologie qui ne répond pas à leurs attentes à cause de l'infidélité à l'égard de la description textuelle. Il se trouve donc judicieux de fournir un système capable de retrouver un concept selon la perception humaine. Notre solution est de construire un système intéractif permettant aux utilisateurs de définir leurs propres concepts visuels à partir de certains mots clés visuels. Ces mots clés visuels, qui en théorie représentent les mots visuels les plus informatifs liés à une catégorie d'objets, sont appris auparavant à l'aide d'un algorithme d'apprentissage supervisé et d'une manière discriminative. Par la suite, pour personnaliser davantage leurs modèles, les utilisateurs ont la possibilité d'intéragir avec le modèle en choisissant en en pondérant les mots visuels auquels ils font confiance.

Le challenge est de construire des mots clés visuels concis et interprétables. Notre contribution repose sur deux points. D'abord, contrairement aux approches existantes qui utilisent les sacs de mots, nous proposons d'employer les descripteurs locaux sans aucune quantification préalable. Par conséquent, chaque composante du vecteur multi-dimensionnel utilisé pour décrire une image est associée à un patch unique et précisément localisé dans l'image. Deuxièmement, nous proposons d'ajouter une contrainte de régularisation à la fonction de perte de notre classifieur pour favoriser la parcimonie des modèles produits. La parcimonie est en effet préférable pour sa concision (nombre de mots visuels réduits) ainsi pour sa diminution du temps de prédiction. Afin d'atteindre ces objectifs, nous avons développé une méthode d'apprentissage à instances multiples utilisant une version modifiée de l'algorithme BLasso. BLasso est une forme de boosting qui se comporte similairement au lasso (Least Absolute Shrinkage and Selection Operator). Cet algorithme régularise

efficacement la fonction de perte avec une contrainte additive de type L_1 et ceci en alternant entre des itérations en avant et en arrière. La méthode proposée est générique dans le sens où elle pourrait être utilisée avec divers descripteurs locaux voire un ensemble structuré de descripteurs locaux qui décrit une région locale de l'image. Notamment, nous avons étendu la version initiale (utilisant un descripteur local unique pour chaque patch décrit) à une version géométriquement consistente qui utilise un ensemble de descripteurs locaux pour chaque patch décrit. Quantitativement, notre méthode est comparable à l'état de l'art du point de vue des performances mais elle est meilleure quand il s'agit d'apprendre des petits objets dans des images fortement encombrées. Qualitativement, l'interprétabilité permet aux utilisateurs de construire des modèles personalisés à partir des patchs appris, favorisant ainsi la définition de requêtes sémantiques composées. Par ailleurs, nous avons développé une interface graphique qui permet d'intéragir avec le système et nous avons montré comment les résultats pourraient être améliorés. Nous notons enfin que notre méthode pourrait bien être appliquée à différente sources de données multimédia comme le texte, l'audio et la vidéo.

إنّما تنجح الفكرة إذا قوي الإيمان بها ، وتوفّر الإخلاص في سبيلها ، وازدادت الحماسة لها، و وجد الاستعداد الذي يحمل على التّضحية والعمل لتحقيقها

ملخّص

تعالج هذه الأطروحة مسألة البحث عن الأشياء في الصّور الثّابتة و هي محاولة لتحسين نتائج البحث المنتظرة عن طريق تفاعل المستخدم مع النّظام . شهد مجال البحث عن الصّور الثّابتة باستعمال محتواها تطوّرا ملحوظا في السّنوات القليلة الفارطة ، فقد سجّل تحسّنا في نوعيّة وصف الصّور كما أنّه أصبح بالإمكان القيام ببحث آنيّ عبر قاعدات البيانات الضّخمة . على الرّغم من التّطوّر المنجز ، لا زالت محرّكات البحث الحاليّة تشكو من عدم القدرة على صياغة و من ثمّ الاستجابة للطّلبات الذّهنيّة للمستخدم . مثال على ذلك هو البحث عن جسد معيّن يحتوي على خصائص دقيقة . في واقع الأمر ، تلاقي النّتائج المعروضة من قبل مكتشفي المفاهيم المرئيّة في الوقت الرّاهن صعوبة في الفهم من قبل المستخدمين و ذلك نظرا لأنّ هذه النّماذج المتأتّية من برامج التّعلّم الإحصائيّة مرتبطة غاية الارتباط بالمعطيات المتاحة ممّا يجعلها تحمل في بعض الأحيان معنى مختلفا عن المعنى الذّي تعبّر عنه الكلمات الأساسيّة المستعملة في تحديد المفهوم المستهدف . يتولّد عن غياب النّتائج المنتظرة عدم ارتياح و قلق إزاء هذه الأنظمة . لذلك توجّب تصميم تقنيّات جديدة تسمح بالبحث عن المفاهيم و عرضها حسب إدراك الإنسان . يتمثّل الحلّ المقترح من خلال هذه الأطروحة في تصميم نظام تفاعليّ يتيح للمستخدم صياغة مفهومه المرئيّ عن طريق مجموعة مقتضبة من أجزاء صغيرة للصّور هي عبارة عن كلمات مفاتيح قد تمّ تعلّمها سابقا عن طريق تعلّم آليّ استنتاجيّ . يمكن للمستخدم حينئذ تخصيص أنموذجه أوّلا بالاختيار ثمّ بترجيح الأجزاء التّي يراها مناسبة .

يتمثّل التّحدّي القائم في كيفيّة توليد نماذج مرئيّة مفهومة و مقتضبة . نكون قد ساهمنا في هذا المجال بنقطتين أساسيّتين تتمثّل الأولى في إدماج الواصفات المحلّيّة للصّور دون أيّ تكميم ، و بذلك يكون كلّ مكوّن من ناقلات الميزات ذات الأبعاد العالية مرتبط حصريّا بمكان وحيد و محدّد في الصّورة . ثانيا ، نقترح إضافة قيود تسوية لدالّة الخسارة من أجل التّحصّل على حلول متفرّقة و مقتضبة . يساهم ذلك في تقلّص عدد هذه الأجزاء المرئيّة و بالتّالي في ربح إضافيّ لوقت التّكهّن . في إطار تحقيق الأهداف المرسومة ، قمنا بإعداد مشروع تعلّم قائم على تعدّد الأمثلة . يرتكز هذا المشروع أساسا على نسخة محوّرة لخوارزميّة لإسو و التي تعتبر طريقة ناجعة لحلّ مشكل لإسو . يمكن تصنيف خوارزميّة لإسو ضمن خوارزميّات بوستينغ (التّعزيز) و لكنّها تختلف عنها لاعتمادها خطوات أماميّة و ورائيّة أثناء كلّ مرحلة و ذلك قصد تقييد دالّة الخسارة . يمكن توظيف هذا العمل باستخدام نوع أو عدّة أنواع من الواصفات المحلّيّة للصّور ، و قد تمّ تطوير النّسخة الأولى التي تستعمل واصفة لكلّ جزء من الصّورة إلى نسخة أشمل تستعمل عدّة واصفات لكلّ جزء و ذلك قصد إضافة المقابلة الهندسيّة . من النّاحية الكمّيّة ، أظهرت النّتائج أنّ الطّريقة المقترحة موازية لطرق أخرى معتمدة حاليّا للبحث عن الأشياء في الصّور و لكنّها تتميّز بتفوّقها الملحوظ عندما تكون هذه الأشياء صغيرة جدّا أو متواجدة بين عدّة أشياء أخرى متكاثفة . من النّاحية الكيفيّة ، أتاح فهم أجزاء الصّور التي قد تمّ تعلّمها مزيدا من الحرّيّة للمستخدمين في بناء نماذجهم المرئيّة مفسحة بذلك المجال لخلق المزيد من الطّلبات المركّبة و الذّات معنى . بالإضافة إلى ذلك ، تمّ تجسيم هذا العمل بتصميم واجهة مستخدم رسوميّة كمثال لتبيين تحسّن نتائج البحث عن طريق التّفاعل . نشير في الأخير إلى إمكانيّة تطبيق الطّريقة المقترحة على مصادر مختلفة من الوثائق المتعدّدة الوسائط مثل النّصوص ، المقاطع السّمعيّة و أشرطة الفيديو.

To my dear parents

Preface

Only two things are infinite, the universe and human stupidity,
and I'm not sure about the former.
Albert Einstein

I sometimes wonder how fast and accurate machines will be in the future. Starting from the fact that computers can already carry out arithmetic operations very much faster than a human being can, and that humans are able to quickly and precisely recognize objects (including detecting their position and segments), how fast are machines going to be able to operate once an efficient object recognition algorithm has been developed?

It has recently been claimed that the human brain is much more powerful than we had supposed, and that by exploiting the capabilities of the subconscious mind, we might be able to make use of this potential in any number of ways. One example is that of reading: in addition to the traditional method, which is taught in schools, we now have speed-reading, photo-reading and quantum-reading. These last two are extremely fast techniques which, if the claims are to be believed, allow people to "read" an entire book in less than five minutes. Of course, the word "read" may not be entirely appropriate here, but it seems that people who have mastered these techniques are able to recall whatever information they are looking for. Every day, we are learning more about human mental capabilities, and even if we don't feel like we have super powers, perhaps, in fact, we do!

Thus, human brain power is increasing, side by side with the power of computers. Nowadays, we talk about intercloud computing and the possibility of moving data from one cloud to another—a notion that would, literally, have been considered 'castles in the air' just a few years ago. The next ten years will undoubtedly be revolutionary regarding this topic. At the same time, researchers are constantly trying new techniques to build more "intelligent"

machines and, in this direction, a number of achievements have been made in computer vision. Yet exploring new ways to exploit collaboration between humans and computers through interaction may well be an equally promising approach to improving computer applications and making our lives easier.

Carrying out research can sometimes be very stressful but, at the same time, it can be a lot of fun—depending on how much time a researcher has to achieve the objectives. In fact, even when the current objectives have been met, the research continues. Working on this dissertation has been the longest writing process I have ever undertaken. Dear Reader, whichever kind of reader you may be, I hope you enjoy it...

Paris, May 18th, 2011
Ahmed Rebai

Acknowledgements

Manners maketh man.
William of Wykeham

This thesis would not have seen the light of day without the various contributions of many people who supported me financially, morally and scientifically. First, and from the bottom of my heart, I thank my beloved parents for their continuous encouragement. The distance which separated us didn't stop them from praying for me and giving me kind advice. I also treasure the kindness of my brother and sisters and all the members of my wider family, including my nephews and nieces.

My sincere gratitude goes to my main supervisor Nozha Boujemaa, director of the center of research of INRIA Saclay and head of IMEDIA team. She gave me the opportunity to work in her lab with considerable freedom and independence. I also greatly appreciated her human qualities. Her numerous responsibilities never prevented her from providing help whenever I needed it. I would also like to thank my advisor Alexis Joly, researcher at INRIA Rocquencourt, for his consistent support, continuous encouragement and the fruitful discussions we had. I heartily appreciate his advice during the few years we worked together. Many thanks to both engineers Souheil Selmi and Mondher Khadraoui, for their collaboration and help. I also thank all IMEDIA members, past and present, for the enjoyable work atmosphere they created: in alphabetical order, Amel, Anne, Asma, Esma, Hervé, Hichem, Itheri, Jean-Paul, Joost, Julien, Laurence, Laurent, Marin, Mathieu, Mehdi, Michel, Mohamed, Nicolas, Olfa, Raffi, Riadh, Saloua, Sofiene, Vincent and Wajih.

I am very much indebted to Richard James, English teacher at INRIA, for helping me to correct my mistakes in English. My PhD research was a long process throughout which I learned new ideas, enjoyed some new events and

experienced some difficulties. Words are nothing but a medium to express my gratitude to my friends who were always there for me and those from whom I learned so much. I mention Bassem, Hichem, Mounir, Oussama, Sofiane, Wassim and Younes. I finally thank God Almighty for everything He has given me.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

As far as the laws of mathematics refer to reality, they are not certain; and as far as they are certain, they do not refer to reality.
Albert Einstein

1.1 General introduction

HAVE you ever woken up and started naming objects: "this is a clock, and I think that might be a chair over there. Oh, wait a second, indeed it is, I can sit on it! Well done, I'm on fire today!" As ridiculous as it sounds, people don't need to make much effort to recognize objects around them. It is trivial , straightforward and sometimes unconscious. In fact, we hardly ever question our recognition system: when we look, we see, and when we see, we recognize.

Although it is as simple as breathing for us, the recognition process is by far more complicated for machines, as is reflected by the fact that, object recognition by computers has been an active area of research for almost five decades. One of the many reasons that researchers have given so much attention to object recognition and detection is that we need machines to help us perceive information that we don't necessarily capture in the first place. For instance, fingerprint, face and iris identification have gained sufficient maturity to allow the correct match to be made from large image databases. Moreover, some applications in information retrieval cannot afford to spend time and effort on manual and textual annotations due to data deluge. For other applications, annotations are just nonsense. Consider, for example, a company that wants to collect statistics about the number of times its logo appears in web images. Another good reason lies in the fact that we need to interact with computers to analyze stored images. Nowadays, it is necessary to make the machine see what a human sees. Faced with the massive amount of information produced

by the media every day, we can have no choice but to rely on computers to highlight the information we are looking for.

Since the previous decade, a number of applications have emerged such as Internet search engines, adult content filtering, face detection and recognition, video archiving and management, teaching and the management of self produced content. This thesis focuses on object-based image retrieval. An object is viewed as a tangible concept such as a flag, a car, a house, etc. Unlike traditional image retrieval systems CBIR where users search for *looking-like* images using global descriptors, in this PhD, the goal of object retrieval is to look for object instances or objects that semantically belong to the same category. or *precise* objects using a local description, An object can occupy the whole image or a small region of the image. In addition, some images may only include a few parts of the whole concept. Despite these variations, search engines must be effective and time-efficient.

1.2 Object retrieval issues

Unlike object recognition which consists in understanding the image scene in depth in order to know whether or not a given object is present, object retrieval aims at ranking the images that are most likely to contain the object in question. Retrieval is performed using object models which are learned *a priori* or online. Moreover, an object model must truly represent all of the different possible instances of the object.

Despite diverse studies on the subject [Sivic 2003, Lowe 2004, Fergus 2005, Lazebnik 2006, Opelt 2006, Philbin 2007, Joly 2009, Wu 2009, Boureau 2010, Pineda 2010], current search engines have not yet reached the point where they are able to correctly formulate and answer a mental user query. The first level of difficulty arises from the fact that computers look at visual content as digits (cf. figure 1.1). The theoretical formulation and design of object models is also far from obvious. In fact, these models are complex to understand and manipulate. They usually involve optimization, kernels and different parameters that need to be properly tuned.

Object retrieval involves many challenges starting with the definition of the training dataset and the choice of the visual descriptors, moving to the

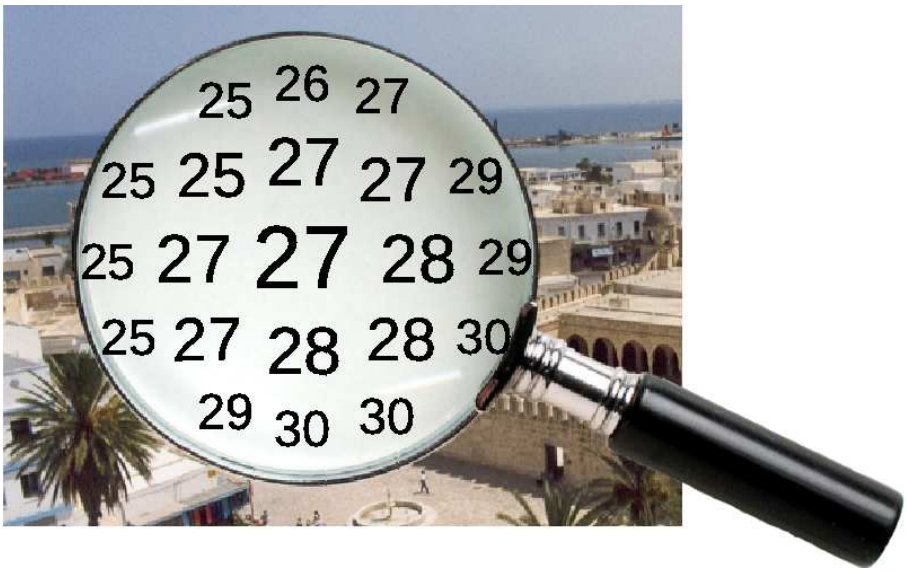


Figure 1.1: How computers look at images.

way the computer learns a reliable classifier. Some researchers prefer to use a bottom-up approach, tracing the very low information in the signal and trying to interpret it so as to get powerful models. Others find that it is more intuitive to use a top-down approach [Agin 1973, Fischler 1973, Cass 1997]. They try to characterize the signal by exploiting their knowledge of what the objects are. It is, however, unclear which of the two methods is the best or whether one should try a combination of both. This raises issues about visual stimulus and how we humans recognize things [Tarr 1995]. For example, is the contextual information always useful? Does it help recognition when scale change occurs or does it make the learning error-prone in the presence of occlusion and clutter? Is it useful in the case of deformable objects?

Among the most important questions in object retrieval is how we define the global similarity between two objects. This question is *application-dependent* and it is also related to the choice of the descriptors, to the distance function and to the model's construction. For this purpose, statistical machine learning offers powerful tools to deal with training data. However, these tools are statistical and consequently assume that the distribution of the training set is statistically similar to the prediction dataset.

In addition to that, the generated models are somehow fixed and largely

dependent on the training set which makes it difficult to apply them on various data. In addition, they might convey a different semantic than the words used to describe the originally targeted concept. Therefore, and in practice, the results returned by state-of-the-art visual concept detectors are often difficult to interpret from a user point of view. This often makes users uncomfortable with these technologies since they do not get what they expected from the textual description of the trained concept. Moreover, the generated models are likely to be abstract. This may lead to confusion among researchers when they are faced with bad prediction performances because they don't really understand where the bad results come from.

Machine learning is also concerned by the computation time but it is not that important compared with the prediction time needed for retrieval. In fact and for a wide variety of applications, learning is done offline, or online with very little training data. Apart from some professional and research fields, all user-friendly online applications need to be as fast as possible. This includes object retrieval search engines. In this perspective, exhaustive windowed searches over location and scale is inappropriate.

1.3 Contributions

The major contribution of this thesis is to make a step forward in generating interpretable models for object retrieval. In doing so, we create a link between the numerical representation of objects and our visual representation of them. The proposed idea finds its roots in text documents. Just as when browsing a text file we obtain a lot of information simply by reading keywords, this could, by analogy, be applied to images. In fact, keywords in text can give an idea about the document type, the field it covers and an overview of the material you are going to process. They are also used to query search engines. With the spread of this technology, users are getting more familiar with formulating compound queries using a few keywords. This very basic principle also works with objects. Visual keywords can indeed give a great deal of meaning to visual content. Take for instance any object you want—say a keyboard or a salamander, and think for a moment about the characteristics that make it what it is. You will soon find out that, in most cases, even

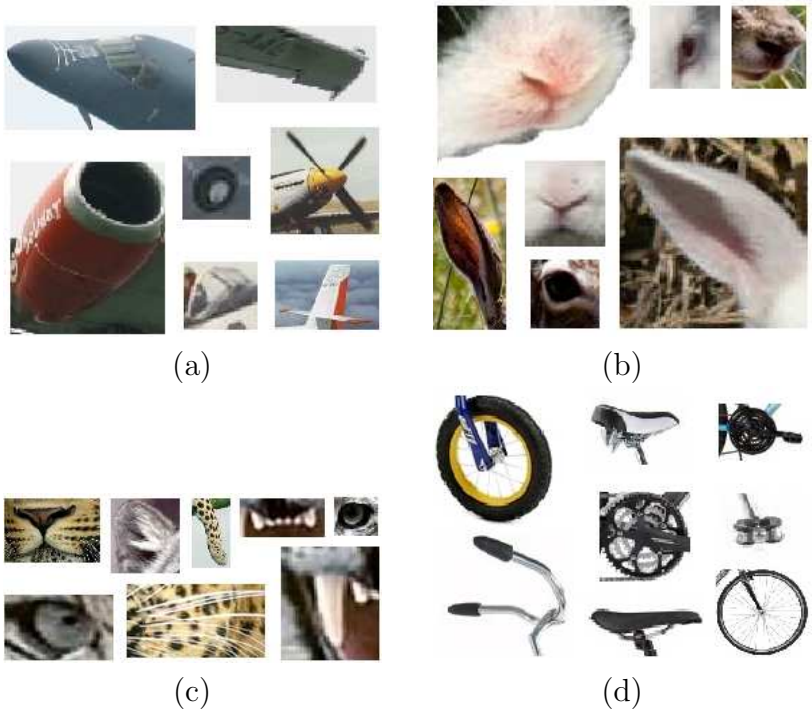


Figure 1.2: Example of four object categories represented by visual keywords.

without any geometric relations between the parts you named, you can still recognize that object. Figure 1.2 presents four objects in order to illustrate this principle.

It is of interest to build systems that retrieve visual objects according to human perception. Our solution lies in constructing an interactive system that allows users to define their own visual concept from a concise set of visual patches given as input. These patches—considered as *visual keywords*—are trained beforehand with a supervised learning algorithm in a discriminative manner. They provide users with a better understanding of what this model is able to retrieve. Then, and in order to specialize their models, users have the possibility to return their feedback on the model itself by choosing and weighting the patches they are confident of.

The real challenge consists in how to generate concise and visually interpretable keyword-based models. Our contribution relies on two points. First, in contrast to the state-of-the-art approaches that make massive use of the bag-of-words model, we propose to embed local visual features without any

quantization. Therefore, each component of the high-dimensional feature vectors used to describe an image is associated to a unique and precisely localized image patch. Second, we use regularization constraints in the loss function of our classifier to emphasize the sparsity of the models produced. Sparsity is indeed preferable for concision (a reduced number of patches in the model) as well as for decreasing prediction time. To meet these objectives, we developed a multiple-instance learning scheme using a modified version of the BLasso algorithm. BLasso is a boosting-like procedure that behaves in the same way as the lasso (Least Absolute Shrinkage and Selection Operator). It efficiently regularizes the loss function with an additive L_1-constraint by alternating between forward and backward steps at each iteration. The method we propose here is generic in the sense that it can be used with any local features or feature sets representing the content of an image region. Notably, we extended the initial version (using single image features to describe image patches) to a geometrically consistent version using feature sets as patch descriptors. Quantitatively, our method achieves similar performances as current state-of-the-art methods but outperforms them when training very small objects in highly cluttered images. Qualitatively, the interpretability allows users to understand and construct their own model from the original set of learned patches, thus allowing more compound semantic queries. Users can perform object retrieval in large image collections which may contain heterogeneous data from different sources. For this purpose, we developed a GUI (Graphical User Interface) that allowed interactivity and we showed how results might be improved. Finally, it is worth mentioning that our method can be applied to various multimedia sources including text (e.g. topic discovery), audio and video documents.

1.4 Thesis outline

This thesis is organized into three parts, each of two chapters. The first part reviews some of the past and current methods used for training object models in still images. The second part covers the contributions we propose for object retrieval. Finally, the third part presents and discusses our experiments. From basic computer vision notions to our contributions in the domain, we

have striven to make each of the chapters as self-contained as possible.

The next chapter begins with a review of local image features, and gives a brief introduction to machine learning and its applications. This chapter also explains some key notions used in the computer vision domain. Chapter 3 explores the state-of-the-art in object recognition as a combination of low feature image processing and machine learning techniques.

Chapter 4 starts by explaining what motivated us to undertake this work, and then presents our new method for training interpretable models consisting of visual keywords. Chapter 5 provides an in-depth account of interactive retrieval. Here, we discuss the prediction phase and the model representation, as well as the user interface.

Chapters 6 and 7 describe our experiments. In particular, the first one judges the overall performance, while the second one shows to what extent user interactivity improves retrieval quality.

Finally, chapter 8 sets out the conclusions and suggests new ways to achieve possible improvements.

Part I

Visual Objects in Still Images

Chapter 2

Computer Vision: the Basics

Honesty is the best image.
Tom Wilson

OBJECT recognition and scene understanding in still images is a computer science discipline that analyzes the digital image content in order to learn and understand what and where the objects in the image are. It has been widely studied in the field of computer vision [Roberts 1963, Grauman 2005b, Wang 2006] for over five decades, and methods that have been proposed are continually re-visited. They are essentially based on two approaches that are not necessarily independent: low level image processing and visual model construction. From the outset of object recognition and identification, there was a specific interest in geometric modeling [Hu 1962]. Many reasons can justify the use of geometry. Geometric description is indeed powerful because of its invariance to viewpoint and illumination. In addition, it is a well developed theory and there are many man-made objects that can be described by primitive geometric elements. For years, people attempted to model objects within a theoretical framework by applying formal mathematics. The idea was to simplify object representations to standard geometric shapes so as to be able to model objects under different perspectives with projective transformations. Among the most advanced and powerful systems developed at the time were the system proposed by [Roberts 1963] and the generalized cylinders originated by [Binford 1989]. After that, researchers abandoned the methods based on geometric simplifications because there were simply too many assumptions, and they moved on to tackle objects in real world scenes. With the development of artificial intelligence and machine learning tools, there was ever-increasing wish to explore methods based on statistical learning and appearance features. This

relies upon image signal processing and basic image feature extraction and description.

2.1 Low level image processing

An image contains a large number of pixels arranged in a 2D plane. The underlying information available is very rich due to the structure generated by the points. However, the information is not located in every pixel but rather in the way they are arranged (cf. figure 4.2). Added to that, it is neither effective nor efficient—in terms of processing—to consider all the pixels. It is therefore necessary to provide some statistical measures to describe and summarize the image content. One possible form of these measures (also called image signatures) is histograms that code some of the many image characteristics. Image signatures are meant to be invariant to a number of image transformations. Now the two eternal questions are what quantity to measure and where it should be applied. Even though they appear separate, these questions are usually dependent.

In general, the three main aspects that can be measured are color, shape and texture. Color histograms were the first descriptors used to summarize the content of images [Swain 1991]. In the literature, various descriptors were proposed to deal with the degree of invariance sought. They mainly differ in the color system used and in the feature space quantization [Fauqueur 2004, Ferecatu 2005]. Color correlograms were proposed by [Huang 1998] as an attempt to improve description by considering the spatial distribution of color. The second main aspect that can be measured is shape information. This characterizes geometrical structures in images and mostly relies on edges and image gradients [Jain 1996, Ferecatu 2005]. Besides color and shape, texture is generally used to describe what is left. It relates to small scale parts which are roughly uniform entities having approximately the same dimension and which are repeated over a region. In reality, there is no universally accepted definition for texture. Some definitions, however, include the notion of order which consists in the non-random arrangement of the elementary parts. Texture can be described by Markovian analysis and spatial frequency techniques such Fourier transform, Gabor filters and

wavelets [Ferecatu 2005, Newsam 2003, Zhu 2005].

Color, shape and texture have been identified as the main low-level aspects that describe images *globally*. In other words, for a given image aspect, only one compact histogram is formed from the raw information that the image contains. These global histograms are still used in today's applications [Ellouze 2010, Hamzaoui 2010, Vinh 2010] particularly in large-scale content based image retrieval where users need to search for globally similar images. In fact, global histograms carry, on the one hand, a global statistical summary of the image content and on the other hand, they are concise which makes them well suited to efficient search. Although global approaches have proved their validity in some applications, they are not capable of solving other computer vision issues since they have low-level semantic content. In fact, the quality of the low-level descriptors used in a given system is dominant over any other component. For applications like automatic image annotation and object recognition, a richer description is needed. To this end, the visual content of an image is not described by a single signature but rather by a set of local signatures that pertain to various image regions. This precise description allows us to improve the overall *effectiveness* since small objects need to be described separately so as to be retrieved or learned more easily. In general, local features computation goes through interest features (i.e. points, regions) localization, support region definition and feature extraction.

2.1.1 Interest points detection

Interest points are effective in highlighting details for various types of images. In fact, the detection process is adapted to many contexts and gives good results with different scenes. This fact makes interest points well suited to locating specific objects and more generally, they are good at searching a specific region selected from the image. These local features gained popularity and were the solution to various matching problems (image retrieval [Schmid 1997], object recognition [Lazebnik 2004, Lowe 1999], wide baseline matching for stereo pairs [Baumberg 2000], robotics [Folsom 2004], and motion tracking [Etiévent 1999]). Using suitable local descriptors on these points gives a summary of the image representation, and so, only a limited number of points concentrates the most relevant information in the image.

Obviously, the greater the number of points, the richer the description generated.

Detecting interest points is not a trivial task since they have to be the most relevant features in the image. In fact, there is no absolute definition of an interest point. To be considered as an interesting feature, a point has to be

robust: to scaling, rotation, shift, viewpoint and illumination changes.

informative: to carry enough information of the local neighborhood so that descriptors will be discriminative.

From previous related work [Schmid 1996, Loupias 2000], we can classify the detectors of interest points into two categories

- Detectors using contours and curvatures.
- Detectors using feature representation. We distinguish here two different approaches; the first approach is based upon the information obtained directly from the signal content and the second approach deals with theoretical models.

Many existing techniques use the first category. Some detectors extract contours first and then detect points with maximal curvature. Others use a polygonal approximation in order to locate salient points. The method by [Asada 1986] is essentially based on plane curves. They classify curvature changes into categories while using a multiscale approach. Mokhtarian and Suomela [Mokhtarian 1998] use inflection points. Other studies were carried out in this field [Medioni 1986, Horaud 1990, Deriche 1990]. The two major difficulties related to this type of detectors is that on the one hand, edge extraction is very sensitive to noise and, on the other hand, the contour chaining may not be well done in cluttered scenes. This is why these methods do not seem to be the most effective. After having been largely abandoned during the 90s, they regained some interest at the beginning of this century. In [Mikolajczyk 2003], Mikolajczyk et al. localize salient edge points through the maximization of a scale-normalized Canny edge detector function. The scale of the interest points is determined by maximizing the Laplacian. Jurie and Schmid [JUR 2004] propose an edge-based region detector that determines salient local space convexities over space and scale.

Among the second category of detectors, the most popular is the Har-

ris detector [Harris 1988]. It computes an autocorrelation function using the first derivative signal. This function locates local changes in the signal. The Harris detector is robust to image rotation, illumination changes, addition of noise and small scale variations [Schmid 2000]. This detector was improved in [Gouet 1998] by extending the unique luminance information into a multi-channel scheme, namely color. It was proved that it is more robust to image transformations [Gouet 2000]. This result was also confirmed in [Stöttinger 2007] where Stöttinger *et al.* show that using color information and boosting salient colors results in improved performance in retrieval tasks. Here, a new color scale selection method was presented and different color spaces were evaluated. [Rohr 1992] proposes an approach modelling the intersections of many lines to detect corners. His technique gives accurate detection. Also dealing with the corner concept, Smith and Brady [Smith 1997] propose the SUSAN detector. They assume that there are two different regions in the local neighborhood of a given point: one region is similar to that point (comparing illumination) and the other is different. So the interest value attributed to the pixel depends on the ratio between the two regions. After that, a threshold is applied to fix the maximum number of interest points to be retained. Here, the simplicity of the detection principle makes the running very fast compared to other detectors. Bres and Jolion [Bres 1999] suggest a detector based on variations in the image contrast which computes the luminance difference between the local neighborhood and the background. The choice of the window size that defines the neighborhood depends on the scale of the image details, which is why a multiscale approach is used. Lowe [Lowe 1999, Lowe 2004] proposes a scale invariant detector called DoG. This detector locates keypoints using scale-space extrema in the difference-of-Gaussian function convolved with the image. It compares each point to its eight neighbors in the current image and to the nine neighbors in the scale above and below. A point is selected only if it is a maxima or a minima. The DoG operator is actually a blob detector approximating the Laplacian of the Gaussian. Yet it is faster. For more details on blob detection, the reader can refer to the work by [Lindeberg 1998]. The aforementioned DoG operator is invariant to four out of six parameters of an affine transform. Recently, Morel and Yu [Morel 2009] extended the previous work

to achieve full affine invariance by varying the two camera axis orientation parameters (latitude and longitude). Mikolajczyk *et al.* [Mikolajczyk 2004] extended the Harris detector to obtain a scale and affine invariant detector, the basic idea being to compute multiscale representation for the Harris detector. Subsequently, the algorithm selects only the points whose Laplacian is maximal over scales. This provides a scale invariant set of interest points each one characterized by its own scale. Using this scale, the affine shape of a point neighborhood is estimated, hence the affine invariance. The method described earlier was also applied by replacing the Harris detector by the Hessian detector. In the same direction, and using a fast Hessian detector, Bay *et al.* [Bay 2006, Bay 2008] propose a method called Speeded-Up Robust Features (SURF). The rapidity of this detector comes from the approximation of the Hessian matrix by convoluting the image with Haar wavelets, which in turn are approximated by Gaussian second order derivatives. Here, the extraction of interest point locations and scales is done at the same time by maximizing the determinant of the scale normalized Hessian matrix. Also using wavelets and curvelets, Tonnin and Gros [Tonnin 2004] propose convolution kernels for detecting points with point and edge continuities. Still with wavelet methods, Sebe *et al.* [Sebe 2000, Tian 2001] use an iterative procedure to determine the highest wavelet coefficients of the same region from coarser to fine scale. Kadir and Brady [Kadir 2001] proposed another type of scale-invariant detector which is basically a salient region detector. This approach finds its roots in information theory where the entropy measure is used. Salient points are indeed centers of regions with high entropy. The goal is to determine local maxima in the affine transformation space from the entropy of pixel intensity histograms whose supports are elliptical regions. Another method based on region detection was proposed in [Matas 2002] under the name of Maximally Stable Extremal Regions (MSER). This algorithm uses a watershed segmentation that starts from local intensity extrema then connected regions grow over pixel intensity variations. A faster implementation of this algorithm was published in [Nistér 2008]. It provides exactly identical results in true worst-case linear time according to the number of pixels. Other methods simply use a random sampling on a regular grid [Hervé 2009]. For a comparison and evaluation of some of the aforementioned detectors, the

reader may refer to [Mikolajczyk 2005a, Mikolajczyk 2005b, Salgian 2006].

Some research on interest point localization focussed on the biological aspect of detection and how interesting these points are according to human vision. In their seminal work, Loy and Zelinsky [Loy 2003] developed a fast radial symmetry for detecting the centers of small objects. Since our world is full of symmetric objects and since these objects have various sizes, Rebai *et al.* [Rebai 2006, Rebai 2007] extended the previous work, first, to detect circular objects and estimate their corresponding radii, and second, to discover new types of symmetries and interest features including vanishing points. The detection of vanishing points makes it possible to infer the 3D structure from 2D images. That is why, when existing in a scene, a vanishing point belongs to the best interest points in that image, and so, it has a high visual interpretability. In this context, the approach developed by Stentiford [Stentiford 2006] is based on a model of human visual attention. It computes the scarcity of a point according to its neighbors. A point that rarely occurs (i.e. unusual) will be attributed a high attention score.

2.1.2 Local feature descriptors

A multitude of local feature descriptors have been developed. However, there is no descriptor known to be suitable for all computer vision domains. Each proposed method is generally limited to a particular range of applications. Local descriptors are computed within support regions which pertain to interesting image primitives (i.e. interest points). Moreover, each key location helps to define one or many support regions. Some local descriptors are completely independent from the interest points detectors used. Other descriptors, however, exploit the underlying information driven by the detectors and incorporate it in the final description. Each technique provides different levels of robustness and the choice to be made is application dependent. According to existing algorithms, we can classify local descriptors as follows: distribution based descriptors, differential and moment descriptors and spatial frequency descriptors.

Distribution based descriptors are similar to global descriptors in that they are histograms that encapsulate the image characteristics related to shape and/or appearance. The only difference is the support region: rather than

using the whole image, statistical measures are taken locally. For instance, Fauqueur *et al.* [Fauqueur 2004] use a coarse segmentation to extract regions and apply a finer color description afterwards. The advantage of this method is to modify the color space quantization for each region (Adaptive Distribution) in order to provide a more accurate description, in contrast to the traditional color space subdivision which is useless in this case. Based on pixel intensities, [Lazebnik 2003] proposed a histogram on point positions in the immediate neighborhood of an interest point. This representation is called *spin image* and was first suggested in [Johnson 1997] in a 3D object context. The spin image descriptor is built from a two-dimensional histogram on the radius separating a point from the central interest point and its corresponding intensity value. Shape context is another type of a structured histogram proposed in [Belongie 2002] to describe edge distribution. This descriptor is mainly used in applications where images contain very stable edges. In fact, edges are first extracted using the Canny detector [Canny 1986] and then, locations are quantized into bins of a log-polar coordinate system. Another shape descriptor called the Directional Fragment Histogram (DFH) was suggested in [Yahiaoui 2006], and was used to retrieve images in botanical databases. The DFH characterizes the outline of the shape of a plant by assuming that the contour is a succession of elementary components (i.e. segments), each of which has information on its length and direction. Ling and Jacobs [Ling 2005] defined a deformation invariant local descriptor named GIH (Geodesic-Intensity Histogram). Unlike usual descriptors, GIH automatically detects its support region and does not need an invariant deformation detector. The idea is to treat a deformation as a homeomorphism between two images, which means that pixel locations change but not their intensity values. They first find a deformation invariant neighborhood using geodesic distances and after that, they build a descriptor based on the intensities of sampled points. Cheng *et al.* [Cheng 2008] propose another local image descriptor that is robust to image deformations. They suggest using multiple support regions around an interest point and then concatenate the feature vectors computed for each support regions. These features are histograms of the gradient directions.

Lowe [Lowe 2004] proposed the SIFT descriptor (Scale Invariant Feature

Transform) and he applied it on DoG points (cf. section 2.1.1). It can, however, be used with other point types. In its basic form, it is a 128-histogram storing the directions of local gradients weighted by their magnitudes. SIFT is constructed from 16 blocks (a 4×4 grid centered with respect to the interest point). Each block quantizes gradient directions into 8 bins. A smoothing Gaussian function is added, on the one hand, to emphasize the information situated in the immediate neighborhood of interest points, and on the other hand, to diminish errors due to side effects. To ensure robustness against rotations, the grid is adjusted according to the main direction of the interest point being described. Besides, SIFT makes a trilinear interpolation in order to eliminate errors due to the abrupt changes in the frontier of two adjacent blocks. This descriptor has been a success in the object recognition domain thanks to the rich information in gradient locations and orientations which makes it robust to small geometric distortions. Yet, the main drawback of SIFT is its relatively high dimensionality which increases the computation time in the matching step. To overcome this problem, Lowe [Lowe 2004] proposed the use of the best-bin-first method in the matching process. Ke and Sukthankar [Ke 2004a] applied PCA on the gradient image resulting in a 36-dimensional descriptor called PCA-SIFT. Although faster in matching, this descriptor is slower to compute and proved to be less distinctive than the original SIFT. In order to increase the robustness and the distinctiveness of SIFT, Mikolajczyk *et al.* [Mikolajczyk 2005a] presented the GLOH descriptor (Gradient Location-Orientation Histogram). Here, SIFT descriptors are computed for a log-polar location grid with 3 bins in radial direction and 8 bins in angular directions for only the 2^{nd} and 3^{rd} radial bins, which results in 17 location bins. Knowing that gradient orientations are quantized into 16 bins, the resulting histogram contains 272 bins that are reduced with PCA to 128 bins. Some comparison results showed that GLOH might be more robust and distinctive than SIFT, but slower in computation. SIFT descriptors are robust to rotations, translations, scale changes and to a large range in viewpoint changes. However, they are not completely affine invariant. In order to achieve full affine invariance, Morel and Yu [Morel 2009] derived an algorithm named ASIFT. They basically keep the same mechanism for building SIFT descriptors, but, they apply them on DoG points extracted

from multiple views of the same image. These views are obtained by varying longitudes and latitudes in order to simulate variations in the camera axes. Matching with ASIFT was shown to outperform the original SIFT as well as MSER [Matas 2002], Harris-Affine and Hessian-Affine [Mikolajczyk 2004].

Among various differential descriptors, we can cite the work by Koenderink and Doom [Koenderink 1987] who studied the properties of local derivatives called "Local Jet". Various refinements on this basic scheme have been proposed. Gool *et al.* [Gool 1996] proved that combining the components of the Local Jet can achieve invariance to photometric changes, image rotation and affine transformations. For instance, Florack *et al.* [Florack 1994] derived differential invariants (a kind of combination of the Local Jet components) to obtain rotation invariance. Also dealing with invariance to rotation, Freeman *et al.* [Freeman 1991] use steerable filters. Given the Local Jet's components, these filters steer the derivatives in a particular direction. Other methods using complex linear filter were presented in [Baumberg 2000, Schaffalitzky 2002]. Carneiro *et al.* [Carneiro 2003] proposed a phase-based local feature. It constitutes a complex representation of local image data obtained through the use of steerable quadrature filter pairs. They also used phase correlation to define a similarity measure between their local features.

Joly [Joly 2007] uses dissociated dipoles to build 20-dimensional normalized features. Dissociated dipoles are non local differential operators which are constructed from a pair of Gaussian lobes. Comparison with SIFT shows that the method performs better in web images search in terms of MAP while keeping a low matching time.

Generalized color moments can also be a good alternative to deal with some image transformations thanks to their properties of combination of shape and color information. Such moment invariants are explored in [Mindru 2003].

Finally, local descriptors can be constructed by exploiting the spatial-frequency information. To do so, one needs to transform the pixel values to some other domains by means of transformation kernels as wavelets, Fourier, DCT, Gabor, etc. Bay *et al.* [Bay 2006, Bay 2008] use Haar wavelets to build a 64-dimensional descriptor called SURF (i.e. Speeded Up Robust Features). In a way similar to SIFT, this descriptor splits the support window into 4×4

square sub-regions. Results showed that SURF outperformed all GLOH, SIFT and PCA-SIFT descriptors.

2.1.3 Feature matching

Local features carry two precious pieces of information: point coordinates and local descriptors. In general, each description method is associated with an adapted similarity measure that could be used to compare local signatures. Finding correct point matches is *not* as easy as finding the shortest distance. In fact, many interest points end up with no correct match because they are not common to the images being compared or because they simply belong to background clutter. Therefore, an appropriate matching scheme should be used.

In his paper [Lowe 2004], Lowe showed that it is not sufficient to use a predefined global threshold on the distance between two SIFT features because, on the one hand, some descriptors are more discriminative than others, and on the other hand, false matches may occur due to the high dimensionality of the feature space. Rather, he suggested—as a global threshold—the ratio between the first closest distance to the second closest distance. This formulation seems independent of the description type. To be considered as a correct match, any ratio must be less than or equal to the predefined threshold. With a PDF graph on the distance ratio, he showed that a value of 0.8 eliminated 90% of false matches while discarding 5% of correct matches. Figure 2.1 is an example of matching two images with this simple strategy using ASIFT descriptors [Morel 2009]. It demonstrates how powerful local descriptors are, even for matching deformed scenes.

In the literature, other matching methods that do not include any spatial registration have been proposed. We will refrain from discussing them here, but the interested reader may refer to [Ke 2004b, Mikolajczyk 2005b, Boughorbel 2005, Grauman 2005a, Zhang 2007].

The above-mentioned techniques use only descriptor information and discard interest point locations. Yet some of these methods add another spatial consistency step to check for local coherence. In fact, even rejecting many false matches using the previous techniques might still lead to "incorrect" matches due to the presence of other valid objects. [Lowe 2004] used, for ex-

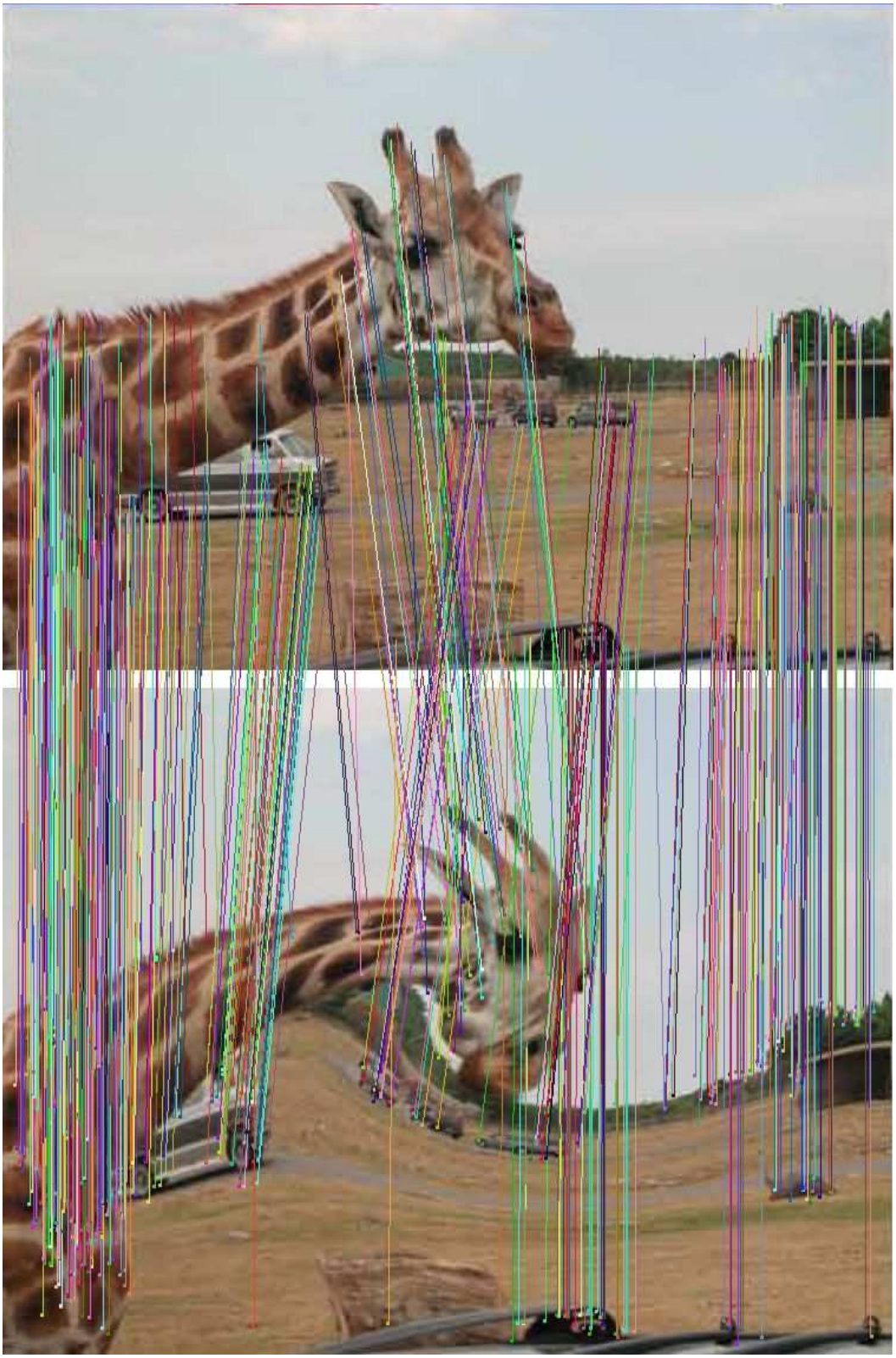


Figure 2.1: Simple image matching with Lowe's strategy.

ample, the Hough transform to cluster features in pose space. When clusters of features are found to vote for the same pose of an object, the probability of the interpretation being correct is much higher than for any single feature.

Other research work [Ke 2004b, Brown 2005, Philbin 2007] have been based upon the RANSAC (RANdom SAmple Consensus) algorithm [Fischler 1981]. Here, an affine transformation model (homography) with 6 degrees of freedom is estimated through an iterative process that selects a subset of the observed interest points as *inliers* and considers the rest of the points as *outliers*. Besides RANSAC which performs robust estimation, Joly and Buisson [Joly 2009] add *a contrario* normalization of geometric consistency scores to further improve matching results.

2.1.4 Bag-of-features

Bag-of-features (BoF) is a technique that can be considered as an embedding of local features distribution into a single histogram. In fact, each image is described by one histogram vector which is usually high-dimensional and sparse, and whose bins are quantized positions belonging to a local feature-space. A local descriptor is then associated with the region of feature space to which it belongs. By analogy with text, the quantized positions in the feature-space are often referred to as *visual words*. The set of them all represents the visual vocabulary or codebook. Thus, an image descriptor characterizes the number of occurrences of the visual words contained in the image disregarding their underlying spatial structure. The term bag-of-features is sometimes referred to as bag-of-visual-words [Yang 2007, Tirilly 2008] or simply bag-of-words [Botterill 2008, Wu 2009] by analogy with the term used in the text retrieval community. [Boureau 2010] presents a comprehensive study for constructing mid-level features.

BoF evolved from *texton* methods where textons, here, are the visual words. A texton (which is derived from the word texture) simply refers to one of the centers of clusters obtained after clustering the responses of linear texture filters. The various approaches proposed mainly differ in the local description or the clustering algorithm used. The codebook can be constructed using interest points [Agarwal 2004, Fergus 2003, Fergus 2005] or dense sampling [Leung 2001, Winn 2005, Agarwal 2006]. For clustering, the two most

commonly used algorithms are k-means [Leung 2001, Winn 2005, Varma 2002, Bai 2010] and agglomerative clustering [Lazebnik 2003, Pineda 2010].

2.2 Machine learning: Theory and applications

Machine Learning (ML) has grown over the past five decades alongside computer technologies. It can be defined as the process of gaining understanding through some observed data so as to construct models that help for further predictions. Scientists are still exploring new ways for computers to learn to play games, to recognize speech and visual objects and to do a variety of other tasks. This field stands at the crossroads of computer science and statistics.

2.2.1 Need for machine learning

ML is used when human expertise does not exist or when humans cannot formulate their expertise, as in speech recognition. It is also useful for solving problems where the optimal solution changes in time or when the solution needs to be adapted to particular cases. ML should definitely not be confused with logic, artificial intelligence and rule-based inference. It is divided into some canonical categories, each of which having its own specific problem to solve. Supervised learning techniques deal with knowledge extraction, typically, when we do know the desired outputs from the inputs and we want to generate rules to predict future cases. For instance, when we want the machine to decide whether an E-mail a spam or not or when we want it to recognize handwritten characters. Supervised learning also includes classification problems, regression and time series prediction. On the other hand, unsupervised learning is less well defined as there is no output (i.e. labels). The task consists in automatically discovering some features, representation or structure from the input data. For example, in marketing, we want the system to classify customers according to their preferences and similarities. Unsupervised learning also deals with association, clustering and density estimation. Fundamental categories of ML also include semi-supervised learning

and reinforcement learning. Semi-supervised learning is proposed to alleviate the tedious task of labeling data. In fact, some of the data used is labeled and some of it is unlabeled. In reinforcement learning, there is no supervised output but delayed rewards or punishments. Here, the computer can take specific actions that affect the world and the next inputs. This kind of learning is used, for example, in game playing.

The key issue in ML is how to represent information numerically. Data can be organized as vectors, matrices, strings or as structured objects such as graphs. It is crucial to decide *a priori* which representation to choose because each one might bring its own problems. It is then of interest to find regularities or structure composing the data. To this end, taking the simplest hypothesis consistent with data seems a good idea. Generally speaking, applying ML algorithms in computer vision without considering the nature of images could lead to undesirable results. For example, consider the task of face registration. If we assume that the image window representing a face is just a two dimensional matrix without taking into account possible interpolation between pixels, then the system is bound to fail because not only does the face's appearance change when moving muscles, but also the task involves registration under different viewpoints, lightening conditions, etc.

2.2.2 Learning methods

In the computer vision domain, both supervised and unsupervised learning are considered to be the most important learning strategies. In this section, we focus on some notions used in supervised learning. For more in depth detail, the reader may refer to [Sebe 2005, Bishop 2007, Hastie 2009b].

Object models in supervised learning are built upon two main families: generative and discriminative. Generative methods try to model the prior probability $\mathrm{p}(image|object)$ within a maximum likelihood framework. In other words, they express the likelihood of an image given the object model. On the other hand, discriminative methods do not learn probabilistic class models but rather try to find boundaries between the categories to be learned. These boundaries, or rather the model parameters, are determined in such a way as to minimize a loss function. Because learning a mapping function is considered to be easier than estimating the prior probability, these methods can

formulate complex relationships between the observed and target variables and have been shown to generally outperform generative methods in classification tasks [Mutter 2007, Subramanya 2007, Bar-Hillel 2008]. Simple methods are sometimes the most effective. Among the early powerful methods that were developed, we should mention the LDA (Linear Discriminative Analysis) approach [Fisher 1936] which finds the best linear transformation (discriminant function) that provides the optimal separation between two object categories. The use of kernel machines was suggested later. The kernel concept itself was introduced in pattern recognition by Aizerman [Aizerman 1964]. The success of sparse kernel machines has been observed during recent years, mostly with the use of support vector machines (SVM).

A typical problem that may occur during learning is *overfitting*. This simply means that the relationships, which the algorithm suppose to be statistically significant, are noise. Consequently, the generated model will not be able to generalize to deal with similar data. To alleviate this problem, regularization is needed. Tibshirani [Tibshirani 1996] proposed an additive constraint with the L_1-norm to shrink the model and favor sparsity. He termed the method *lasso*. Subsequent work [Meinshausen 2009, Xu 2010] has shown the robustness properties and sparsity characteristics of lasso. In addition to these results, Xu *et al.* [Xu 2010] proved a theorem stating that sparsity and algorithmic stability contradict each other. Therefore, lasso is not stable. We should emphasize here that stability differs from robustness. Stability means that when slightly varying the training data or the regularization amount, the algorithm will output a slightly different (i.e. similar) model. On the other hand, robustness means that applying such variations would not affect the prediction performance on test data (regardless of whether the model is slightly or completely altered). In reality, this result—proved by Xu *et al.*—was somehow revealed by Meinshausen and Büehlmann in their paper 'stability selection' [Meinshausen 2009]. They addressed the problem of determining the proper amount of regularization for variable selection in high-dimensional data when the number of variables significantly outnumbers the number of observations—which is generally the case with local image features.

Variable selection, also known as feature selection or feature reduction, is a common ML task required when the number of variables is very large. It

is achieved in order to highlight the most reliable data and could be done as a preprocessing stage in order to improve the performance of a learning algorithm. Some methods, however, can identify interesting features at the same time as training classifiers.

Supervised learning mostly relies on labeled data. Some problems in real life cannot provide exact information about the true labels either because it is technically impossible or because collecting labels demands a great deal of effort. In order to overcome this issue, Dietterich *et al.* [Dietterich 1997] came up with a multiple-instance learning (MIL) scheme while investigating drug activity prediction. Since then, MIL has enjoyed success in many applications including image retrieval and categorization [Maron 1998, Andrews 2003, Zhou 2006, Li 2009, Fu 2009]. Rather than labeling each training sample, MIL only labels a group of samples (also *instances*) known as *bags*. A bag is positive if and only if it contains at least one positive instance and the task consists in predicting the labels of *unseen* bags.

Chapter 3

Object Recognition: the State-of-the-Art

Art is the imposing of a pattern on experience, and our aesthetic enjoyment is recognition of the pattern.
Alfred North Whitehead

IN different applications, there is a need to bring together objects that share the same characteristics or function. We talk about categorical object recognition or object categorization. A category can encompass objects that have no boundaries, objects that are purely conceptual but which are still tangible. In this chapter, we give key-notions on how objects are defined and dealt with in the literature. We provide an overview on how specific tasks are tackled and we explore some of the-state-of-the-art methods used for modeling and learning.

3.1 Objects: What and Where?

3.1.1 Object types

Depending on the far-end application, the definition of the concept to recognize may vary. Objects can be classified into bounded/unbounded, rigid/flexible. Bounded and rigid objects can have various shapes (e.g. car, chair), a fixed shape with different appearances (e.g. rugby ball, CD), or a fixed shape and appearance (e.g. landmark). On the other hand, bounded and flexible objects have various appearances (e.g. an open eye vs. a closed eye).

Dealing with objects that have a limited volume is still a formidable challenge. These are the objects that have been the most studied in the literature.

The easiest categories might be those that define rigid objects with a fixed shape. Yet, even with a shape that doesn't vary much from one instance to another, a deformable object easily adds another level of difficulty either by changing the pose, appearance or point of view (e.g. flag, face). Having a rigid object with different instances is another example that makes categorization difficult. The task can be even trickier when recognizing objects with different shapes and that can be easily deformed (e.g. carpet, handwritten digits).

Dealing with rigid objects, it is sufficient to use a robust interest point detector and descriptor coupled with an efficient matching algorithm. For example, object recognition performed in [Lowe 2003, Morel 2009] uses a similarity matching algorithm. First, each keypoint is associated with its nearest candidate match (the closest neighbor according to the Euclidean distance). Then, a comparison to the second closest distance is performed. Applying a threshold on the ratio defined by the first closest distance over the second closest distance turns out to be a very useful means to discard many inconsistent matches. In addition, this ratio threshold is beneficial because it facilitates, in terms of efficiency, further processing steps. In fact, and for more accurate object recognition, a Hough transform is used in order to cluster the keypoints with same characteristics. After identifying at least three entries in a bin, an affine verification procedure is performed for additional geometric consistency.

Categorizing unbounded objects is certainly difficult. For example, scene categorization has been of interest in many research studies [Lazebnik 2006, Jégou 2007, Gemert 2008]. Scenes can be classified into indoor/outdoor, day/night, urban/forest, etc. More specific topics can also be applied (e.g. class-room, bedroom). Recently, Xiao *et al.* [Xiao 2010] have developed a database of 899 categories and 130,519 images that is used for scene recognition. They gave it the name SUN (Scene UNderstanding). Within this same context, we can talk about *event* search. In fact, object classes may consist of specific events. Each event is defined by a place and a limited period of time (e.g. music concert, wedding, football match). Although searching for events is not well covered in the literature, it is an area of interest particularly for users who want to collect images for the events they pertain to.

3.1.2 Challenges

Object recognition tasks involve detection, classification and retrieval. Since our work focuses mostly on classification and retrieval, we just give a brief overview on detection, moving on after that to present some state-of-the-art methods on the subject.

3.1.2.1 Detection

Object detection aims at identifying the position of any instance representing the object class. In other words, detection must provide answers to the two following questions: How many objects are there? Where are they?

Detection may be achieved by indicating a bounding box in which the object is localized. For example, Laptev [Laptev 2006] uses a rectangular window search over positions and scales. Then, he clusters positively classified sub-windows in order to eliminate multiple detection. A similar approach based on an exhaustive scan of rectangular features is used by Viola and Jones [Viola 2004] for face detection. Osadchy *et al.* [Osadchy 2007] suggest a more precise delimitation and representation of faces by estimating relative pose. They used an oriented box as well as two axes: a "vertical" axis passing through the nose and a "horizontal" axis showing the position of the eyes. On the other hand, Tuzel *et al.* [Tuzel 2008] use ellipses to represent detected pedestrians.

Since a bounding box might omit a part of the object or might be imprecise by including a great deal of context around the object, it is considered to be the easiest way for localization. A better way is to specify a segmented region which defines the boundaries of the object itself. For this purpose, Marszalek and Schmid [Marszałek 2007] make use of shape masks to accurately locate object instances by defining outlines. Shape masks are considered to be a natural generalization of discrete binary segmentation masks.

Forssén and Moe [Forssén 2006] learn the *appearance* of objects. The goal is to allow robots to detect objects with little or no texture. In this case, detection results were evaluated by drawing a small symbol on the object's center. In order to locate an object, another technique consists in labeling the local image features (interest points) [Ulusoy 2006]. This method can be viewed as a classification of each local feature.

3.1.2.2 Classification and retrieval

Classification, also called *object categorization*, consists in classifying images into categories by giving them the labels of the objects they contain. It is only based on the presence or absence of the object. For each object category, images are attributed the value 1 or 0 depending on whether or not they contain the object.

Following the same idea, we can perform object retrieval. Users can query the system to retrieve images which contain the objects they are looking for. This should be based upon an efficient and fast prediction to ensure a reasonable response time, particularly when dealing with large databases and/or complex models.

Nowak and Jurie [Nowak 2007] propose a framework where the notion of modeling an object category is absent. They compare never seen objects. The training set comprises pairs of images all weakly labeled as being *same* or *different*. These are the only clues that are available for the learning algorithm. Given two test images, the task consists in labeling this pair of images as "same" if they contain the same object and "different" otherwise. The method is based on learning a similarity measure from *equivalence constraints*. The algorithm aims to learn the similarity between small patches as well as to characterize their local differences. This method is practical to tell whether two faces are different or not even under different appearance. It can also be used to cluster a bunch of images according to their similarity. On the other hand, all the experiments shown used images containing objects that mostly cover the whole image area and, it is unclear to what extent this method is still applicable when objects become smaller (with respect to the image size).

Although they represent an unstructured set of data, bag-of-features methods turn out to be very discriminant and they are considered to be effective tools for classification. The main choices that have to be made are about how to choose interesting local image regions, how to describe them and how to quantize the feature space. Several authors have attempted to use specific interest operators to extract image local features. However, Nowak *et al.* [Nowak 2006] demonstrated an increase in performance when using a random sampling. They added that random sampling gives better classifiers and that interest operators are not suitable because they cannot provide enough

patches. The more patches used, the more robust the classifiers will be. The reader can refer to [Nowak 2006] and the references therein for a more in depth explanation.

Visual words carry much more information than textual words, hence more structure is needed. Moreover, with a bag-of-features method, the choice of the words is abstracted to the system. In this direction, a few attempts have been made to introduce some spatial consistency to bag-of-features. Philbin *et al.* [Philbin 2007], for example, perform object retrieval by introducing a fast spatial matching. By object, they mean a query region selected by the user. After retrieving images using a bag-of-features scheme, they add a spatial verification stage and only re-rank the top-ranked results. In fact, they estimate a transformation between the query region and each target image using spatial constraints (based on how well feature locations are predicted by the estimated transformation) and apply the LO-RANSAC [Chum 2004] algorithm afterwards. On the other hand, Lazebnik *et al.* [Lazebnik 2006] proposed a spatial pyramid matching in order to encode and bring more spatial structure to the unordered words. They partition the image into sub-regions at different levels, from coarse to fine. Their idea has its roots in the pyramid matching kernel [Grauman 2005b]. PMK represents a weighted sum of histogram intersections and the idea by [Lazebnik 2006] consists in representing the image with multiple histograms each corresponding to a geometric location (i.e. image sub-region). Furthermore, the weights in PMK are computed so as to penalize matches found in larger cells because they involve increasingly dissimilar features. Yang *et al.* [Yang 2009] state that bag-of-features and spatial pyramid matching (SPM) should be used together with a particular type of non-linear Mercer kernels in order to obtain good performance. They added that this result has been empirically proved. Vedaldi *et al.* [Vedaldi 2009] use dense and sparse visual words at different levels of spatial organization. Another work by Philbin *et al.* [Philbin 2008] uses an affine homography as a geometric relation built into a generative latent model. The method is an extension to LDA and was named *g*LDA (Geometric Latent Dirichlet Allocation). It is able to compute a matching score of the spatial consistency as well as an approximation to the transformation between two spatially distributed sets of bag-of-features.

Due to their simplicity and computational rapidity, using approaches that don't model relationships between parts of the object is appealing. For instance, Opelt *et al.* [Opelt 2006] propose training the classifier from weakly labeled images. The object class model is composed of a weighted sum of independent local features whose weights represent a degree of confidence in their reliability. Moreover, each local feature is characterized by a discriminative radius. Lin *et al.* [liu 2007] build an MRF graphical model to learn various local ensemble kernels and use an SVM classifier afterwards. Torresani *et al.* [Torresani 2010] introduce a new representation called *classemes* to allow *novel* categories to be recognized. A classeme is a base classifier describing either an object similar to the category to be learned or an object which is often present in that category; the idea is to combine all these base classifiers. Frome *et al.* [Frome 2007] classify visual categories by learning an image-to-image distance function. The objective is to obtain smaller distances between images from the same category than any two images belonging to different categories. The distance function defined is a sum of weighted distances where each distance corresponds to a local patch belonging to the query image. Furthermore, the weights are attributed in a way to emphasize "relevant" features. Seinstra and Geusebroek [Seinstra 2006] address the problem of recognizing objects by robots. They compute 37 local color-based histogram invariants on a hexagonal grid and object recognition is achieved by matching these histograms.

Geometric approaches consider objects as being an association of a set of parts whose relative positions are constrained by the model. Zha *et al.* [Zha 2008] build a model from a multi-label framework where they jointly learn connections between regions and their corresponding labels. Their probabilistic model is the sum of four potential functions. It formulates the association between a given region and its label, the coherence between regions and image labels, the spatial relation between region labels as well as the correlations of image labels. Here, the spatial relationships only take into account the neighborhood as a constraint and model each pair of labels separately. The potential function is a weighted sum of these pair-of-labels power indicators. Fergus *et al.* [Fergus 2003] model objects as a random constellation of P parts. A histogram (of length P) is then constructed in order to associate each

part to the number of its relative instances present in the image. The value 0 indicates that a part is missing. Moreover, these parts are modeled with a probabilistic representation that takes into account shape (Relationships between parts are represented by a shape model.), appearance, occlusion and relative scale. Epshtein and Ullman [Epshtein 2007] construct a probabilistic graphical model which encapsulates various appearances of the object in a semantic hierarchy. The graph is composed of hidden nodes that correspond to object parts. Each hidden node contains two variables specifying the appearance of the object part and the location in the image. Bar-Hillel and Weinshall [Bar-Hillel 2008] describe an efficient method to model object parts taking into account appearance, location, and scale as well as relations between the parts. These relations are related to one another through a hidden center. Li and Zhang [Li 2010] propose a method which is considered to be an improvement to traditional LDA. Unlike [Philbin 2008], their method abandons the notion of bag-of-features and learns directly from SIFT descriptors. The training takes into account the location coordinates of image features and employs prior spatial configurations and affine transformations to characterize spatial information. Amores *et al.* [Amores 2007] represent an image as a constellation of generalized correlograms (GC) that correspond to image parts together with their relative contexts. They use various interest operators to determine the location where to compute the GCs in order not to miss any informative location. A GC estimates a joint distribution of local and relational properties. Apart from characterizing spatial relations, their representation of correlograms also takes into account relations between local parts of the object.

3.1.3 Evaluation

3.1.3.1 Metrics

Various metrics have been developed to compare and judge the effectiveness of existing methods. For classification tasks, the basic measures used to evaluate and to derive, on a second level, other metrics are true positives, false positives, true negatives and false negatives. A true positive is equivalent to a hit while a true negative means a correctly rejected hypothesis. On the

other hand, a false negative is equivalent to a miss (i.e. a falsely accepted negative hypothesis) and a false positive is equivalent to a false alarm. Many researchers plot ROC (*Receiver Operating Characteristic*) curves to report their results. The ROC curve is drawn from true positives as a function of false positives and the area under this curve is interpreted as the probability that a randomly chosen positive image will be ranked higher than a randomly chosen negative image.

For retrieval applications, the most common measure is *precision*. It is defined as the ratio of the number of *correct* answers to the number of the documents retrieved. Unless equal to one, this measure doesn't specify the *order* of good documents with respect to all the retrieved documents. Yet the notion of order is very important in retrieval. One way to encounter this problem is to define a precision at a cut-off level, so that, for instance, we can evaluate the precision for the first 5 documents, then for the first 10 documents, etc. On the other hand, the precision value only evaluates the documents retrieved and doesn't give any clue about the relevant documents that should have been retrieved and were not. Here comes another measure called *recall* which specifies the proportion of the relevant documents. It is defined as the ratio of the number of *correct* answers to the number of all the relevant documents in the database. Obviously, returning all the database documents results in a recall of 1, which is why, recall must be jointly used with *precision* to give a clear idea about the performance. To this end, a *precision-recall* curve can be derived (precision as a function of recall). The closer the area under the curve to one, the better the system is. But more generally, and for two equal areas, the system is better when the precision is higher at an early stage so that good results are shown in the first place.

Although a curve is more informative than any summarizing number, in many cases, it is tedious and time-consuming to compare several performances based on curves. In addition, the evaluation process moves from being quantitative to qualitative. For this reason, it is preferable to rely on one numerical measure to make it possible to rank different algorithms and/or different runs. Consequently, we can think of using the value of the area under the precision-recall curve for the matter. This area can be approximated by the *average precision* measure (AP for short). The AP favors returning relevant docu-

ments first while keeping some knowledge on recall. In fact, it is an average over precisions computed at each level where a relevant document is found. This means that when a relevant document is retrieved after many bad documents, the precision at this level is low and would consequently affect the final measure.

3.1.3.2 Databases and benchmarks

Databases are an essential ingredient to object recognition research. In fact, they play a key role in defining user needs as well as in evaluating existing algorithms. Consequently, appropriate datasets are required. To some extent, certain datasets are qualified as easy and inappropriate for a given challenge, while at the same time, they could represent a good means of evaluation for other tasks. That is why, different datasets may be complementary to one another.

Each far-end application is designed to meet one or many objectives (Video surveillance in a public street, video surveillance in a supermarket, detection of faces in a camera, copyright fraud detection, pedestrian detection when driving a car, retrieval of a specific image region, etc.). Some of these applications focus on speed rather than precision. Other applications may need to minimize false positives or rather, they may not tolerate false negatives. Since each application has its own specific requirements, algorithms should be compared depending on the goal they were designed to. Unfortunately, rare are the databases that are task specific except of some databases which were constructed within benchmarks [1] [2]. Some researchers prefer to build their own dataset. For example, Fergus *et al.* [Fergus 2005] use images directly provided from search engines to build their own training and test set.

3.2 Object characterization

Characterizing an object class is equivalent to making the right choice for description, model types, learning algorithms and similarity measures to name but a few. One of the important decisions is to choose an appropriate

1. http://pascallin.ecs.soton.ac.uk/challenges/VOC/
2. http://www.imageval.org/e_presentation.html

description. The next section summarizes some types of description used in object recognition.

3.2.1 Choosing a good description

Image descriptors are the raw material used to train, build models and predict on unseen images. It is usually necessary to proceed with a local description since objects may be small in size with respect to the image size. Using a local description allows us to capture subtle information in various image locations. A global description would be equivalent to a vague measure of image characteristics. Nonetheless, the question where to measure this information is no by means obvious and it is certainly task-related. The reader may refer to [Salgian 2006] for a brief comparison on local descriptors applied to object recognition.

Interest points are usually more abundant in real images than straight edges and several research studies have used them as the basic image features. [Amores 2007] extract edges and corners (i.e. contour points) and then sample this set by keeping the points that have a maximum distance from each other. After that, and for each interest point, they use a correlogram representation to measure the distribution of neighboring points according to their positions and local properties. A difference-of-Gaussian interest point detector, alongside with the SIFT descriptor [Lowe 2004], are used in [Ulusoy 2006, Jégou 2007, Agarwal 2006, liu 2007, Li 2010]. Harris-Laplace and Affine-Hessian [Mikolajczyk 2004] are respectively used in [Ulusoy 2006] and [Jégou 2007]. [Bouchard 2005] use the Harris detector [Harris 1988] and then apply an agglomerative clustering over 25×25 patches. Similarly, [Weber 2000] use the Föstner detector [Föstner 1987] and apply vector quantization over 11×11 patches for clustering.

Support regions defined for interest points usually have a square shape. For a more accurate description, regions that capture exact shape and object structure are used. For example, Deng *et al.* [Deng 2007] propose a region detector based on the principal curvature. They first extract curvilinear structures using the maximum or minimum eigenvalue of the Hessian matrix at each pixel. Then, they detect structural regions using a watershed transform of the principle curvature across scale-space. The method is fur-

ther refined to avoid over-segmentation (due to small watershed regions) or unstable regions (due to low contrast segments). To capture more structure, research by [Fergus 2003, Bar-Hillel 2008] use the Kadir and Brandy region detector [Kadir 2001]. This is an entropy-based operator which finds regions that are salient over location and scale. Gosselin *et al.* [Gosselin 2007] exploit a fuzzy-region segmentation approach for object retrieval. They use a watershed algorithm on the image of gradient norms. The cores of fuzzy regions are then deduced from uniform areas. To control the final number of regions, a merging step is performed afterwards. Possible overlaps between regions is stabilized through feature computation, which takes into account the membership degrees of the pixels.

In contrast to region detectors, [Forssén 2006] presented a method that computes invariant frames from color contour description. The method tackles the problem of learning texture-free objects and is robust to scale change. First, it extracts line segments and ellipses as the image primitives. Then, and for each of these primitives, it constructs an invariant frame defined by a similarity invariant transform. Feature vectors are computed afterwards by sampling the image orientations according to the similarity transform. [Shotton 2005] proposed a method for detecting objects using local contour features. The feature detector is learned using a very small set of segmented images then it is applied to a larger training set. The model forms a constellation. It comprises many contour fragments arranged around a centroidal point. These models were compared using the chamfer distance, which is considered to be tolerant to misalignment in position, scale and rotation. Experiments were carried out on horses, cars (side), faces and motorbikes. However, no results were reported on whether the method would discriminate between objects with similar shapes (e.g. cat, puma, panther, jaguar, etc.). [Haugeard 2009] addressed the problem of window extraction from facades to retrieve buildings. They proposed an accurate detection of contours using the Canny detector by automatically adapting the parameter related to the size of the filter to the correct smoothing scale of analysis. Contour fragments are then matched using a graph matching technique with a kernel that takes into account orientations and proximity of contours in the structure.

Although interest operators have proven to be efficient in locating salient

features, some scientists prefer to use a random or a regular sampling. For instance, [Nowak 2006] stated that random sampling gives equal or better classifiers than sophisticated multi-scale interest detectors. [Viola 2004] noticed that rectangular features provide a rich image representation for effective training. In fact, rectangular features are considered to operate at a coarse level due to their sensitivity to edges, bars and other simple image structures. For face detection, they defined three kinds of features based on Haar basis functions and introduced a novel image representation called integral images for fast computation. In order to cover several parts of the object, [Laptev 2006] proposed using various rectangles in the normalized object window where he computed histograms of gradient orientation. Tuzel *et al.* [Tuzel 2008] made use of integral images representation. They extracted a large number of descriptors with overlapping regions and showed that overlapping significantly contributes to increasing detection performance. Their method was applied to pedestrian detection. A region is represented by a covariance-matrix of image features and a human is represented by several covariance matrices of overlapping regions. Within this representation, the feature space created is viewed as a connected Riemannian manifold. Kokkinos and Yuille [Kokkinos 2008] convert image scalings and rotations into translations by combining grid log-polar sampling and spatially varying smoothing. Then, scale invariant descriptors (dim=128) are built using phase, orientation and amplitude features from the Fourier transform.

An alternative to using samples from images is to use the available information provided by all the pixels. Osadchy *et al.* [Osadchy 2007] propose a system that maps raw images to points in a low-dimensional space, where a face manifold is defined. The idea consists in mapping face images with known poses to points in that manifold and non-face images to points far away from the manifold. The training images used are 32×32 pixel-sized. Along with descriptor location, some research work has attempted to optimize the description generated through learning the parameters of the descriptors [Winder 2007] or combining different feature representations [liu 2007].

3.2.2 Modeling and learning

3.2.2.1 Object modeling

Based on the search or classification method, objects are usually represented with a model-free representation, as an entire entity or with reference to their parts.

A model-free representation tries to characterize global statistics about objects in a way that allows the machine to guess the presence or absence of a particular object, regardless of relations between the object's parts. This representation is also called geometry-free. In fact, there is no related information to the positions of the characteristics being learned. The advantage of such methods is that they don't require a *strongly* labeled training set where the positions of the object, and probably its parts, are annotated. Computationally, they are generally the most efficient. On the other hand, it is unlikely they can successfully provide location. Some of these methods include, but are not limited to, geometry-free bag-of-features [Nowak 2006, Opelt 2006, Frome 2007, Nowak 2007].

Among geometry-free modeling, considerable attention has been given to the bag-of-features technique. Despite its simplicity, it shows very good performance. The success of this method may be due to its flexible representation which allows a wide range of visually different classes to be covered. The method essentially codes local image patches independently using statistical appearance models. Originally, bag-of-features mimicked text retrieval engines by using the concept of word search. In fact, the method consists in defining "visual" words from the feature space of description. Images are then described according to these words by building a relatively large histogram containing occurrences of each observed word (each local feature will be assigned to its closest visual word).

Some methods model objects as whole entire entities that are inseparable. Such methods are mostly used in localization. Marszałek and Schmid [Marszałek 2007] define a shape mask function which is a generalization of the discrete binary segmentation mask. Using a bag-of-keypoints representation, the classifier is trained to distinguish between objects and background and the resulting model is a featured shape mask globally representing the object. Philbin *et al.* [Philbin 2008] try to discover particular

objects by first clustering similar images and then modeling objects along with their location and shapes. In order to detect humans, Tuzel *et al.* [Tuzel 2008] use covariance descriptors [Tuzel 2006]. They apply a classifier at all possible sub-windows in the image. Lowe [Lowe 2004] uses distance matching followed by a Hough transform to determine consistent features with the object geometry. For rigid objects, [Philbin 2007] and [Joly 2009] consider objects as image regions (i.e. query regions).

The third group of methods for forming object models is based on detecting object parts, including common shapes, and then assembling these local parts according to geometric constraints. These part-based methods usually model parts' appearance as well as invariant relations of location and scale between all parts. They can be good for dealing with intra-class variability but they may, however, be sensitive to missing parts. Shotton *et al.* [Shotton 2005] use contour fragments as raw object parts and exploit their spatial position to build a class dictionary from segmentation masks. The model they propose admits a centroidal point that pulls together the different object parts. Bar-Hillel and Weinshall [Bar-Hillel 2008] use a simple *star-like* Bayesian network whose parameters are learned through a discriminative optimization. Moreover, the model has a central hidden node describing objects' location and scale. Bouchard and Triggs [Bouchard 2005] code visual object categories as a loose hierarchy of parts with probabilistic spatial relations linking parts to subparts. Similarly, [Epshtein 2007] constructs a semantic hierarchy of object parts. Here, part detection is obtained by a bottom-up top-down cycle. [Fergus 2003] models objects as flexible constellations of parts. A probabilistic representation is used for all aspects of the object including shape, appearance and relative scale.

3.2.2.2 Object learning

Learning objects should be independent from the object design. Yet a learning strategy might be more suitable for certain models rather than for others. Under any circumstances, one has to consider the computational problems involved in using a particular method. This section addresses some of the state-of-the-art algorithms used for learning object models in a supervised manner. They are divided into two categories: generative and discriminative.

Generative-based approaches build a probabilistic model learned by maximizing likelihood. The likelihood ratio test is then used to classify new images. [Fergus 2003] uses an expectation-maximization (EM) algorithm in a maximum-likelihood setting in order to estimate the model parameters. [Epshtein 2007] uses a probabilistic graphical model where the conditional distribution p($Image|Class$) is modeled as a hierarchy of features and sub-features, and the conditional distribution p($Image|NonClass$) is modeled as a Naive Bayes model. In [Philbin 2008], Philbin *et al.* develop a generative latent model with geometric relations at its core. The model is called gLDA as it brings geometry to traditional LDA. Similarly, Li and Zhang [Li 2010] incorporate the spatial structure of local features into LDA. They term the model the Affine Invariant Topic Model (AITM). The method models visual words with latent affine transformations as well as latent topics. Moreover, classification is performed using the Bayesian decision rule.

Discriminative methods try to extract or to separate useful information from the whole set: positive and negative images. Here, the term "useful information" refers to any image feature that helps to correctly classify a given object class, regardless of whether or not this feature is part of the background. One advantage of such methods is that the task lies in minimizing a loss function. Nevertheless, the statistical distribution of the test database must be the same as the statistical distribution used in the training set. [Nowak 2007] uses a set of extremely randomized binary decision trees to compare never seen objects. This method focuses on speed and the random character involved decreases the risk of overfitting. Pham and Cham [Pha 2007] seek to balance the skewness of the labels presented to the discrete-valued weak classifiers learned with AdaBoost in an online manner. [Laptev 2006] also uses AdaBoost in its offline version. Based on AdaBoost, Viola and Jones [Viola 2004] define a method that combines increasingly more complex classifiers of simple features in a cascade. The cascade is viewed as an object specific focus-of-attention since it quickly discards background regions of the image. Similarly, Zhu *et al.* [Zhu 2006] combine multiple classifiers, learned with AdaBoost, to form a rejection cascade such that if any weak hypothesis is negative, then it is considered to be a negative example. Tuzel *et al.* [Tuzel 2008] keep the cascade mechanism but use LogitBoost instead. At each boosting step k, the

best classifier corresponds to the classifier that minimizes the negative binomial log-likelihood to the cascade level k. [Osadchy 2007] detects faces and their poses using the minimum energy machine framework [Huang 2004]. The method consists in mapping raw images to points in a low-dimensional space where a 'face manifold' is pre-defined. Images that contain faces with known poses are mapped to the face manifold. Inversely, non-face images are mapped to points that are far away from the manifold. The image-to-manifold mapping function uses a convolutional network as the basic architecture. To classify *classemes*, Torresani *et al.* [Torresani 2010] combine 13 kernels using the LP-β kernel combiner [Gehler] and try four different strategies for multi-class learning: multi-class SVM, neural networks, decision forests and nearest-neighbor classifier. Marszałek and Schmid [Marszałek 2007] evaluate their shape masks using a non-linear SVM with χ^2 kernel. Kumar and Sminchisescu [Kumar 2007] advocate the transition from the use of SVMs to SKMs (Support Kernel Machines). SKM models estimate the parameters of a sparse linear combination of kernels as well as the parameters of a discriminative classifier. Lin *et al.* [liu 2007] carry out the recognition task with adaptive ensemble kernel machines. They construct a number of kernel matrices, each of which corresponds to a specific type of image feature. Then they fuse these features through kernel alignment [Cristianini 2002]. Yang *et al.* [Yang 2009] use the pyramid matching kernel for image classification. To remedy the problem of algorithm complexity during training (which is $O(n^2 \sim n^3)$) and prediction ($O(n)$), they develop an extension of the SPM method by generalizing vector quantization to sparse coding followed by multi-scale spatial max pooling. In addition, they propose changing the non-linear classifier by a linear classifier using a linear SPM kernel based on SIFT sparse codes. This approach reduces the training complexity to $O(n)$ and prediction complexity to a constant. For object detection, Vedaldi *et al.* [Vedaldi 2009] provide a cascade classifier made up of three stages including linear, quasi-linear and non-linear kernel SVMs. Rather than specifying a pre-defined kernel, they aim to learn a combination of given base kernels. Gemert *et al.* [van Gemert 2010] introduce another analogy to BoF models with text visual words for modeling ambiguous quantifiers such as "some", "much" and "-ish". They incorporate visual word ambiguity in the codebook model by softly assigning continuous

image features to discrete visual words. Furthermore, and for classification, they use an SVM with a histogram intersection kernel.

Some methods combine both generative and discriminative strategies. In [Ulusoy 2006], Ulusoy and Bishop argue that neither a generative approach nor a discriminative approach, alone, is sufficient for large scale object recognition. In addition, their results state that both approaches have complementary strengths and weaknesses. Bar-Hillel and Weinshall [Bar-Hillel 2008] propose a relational model that generatively models object classes using a simple Bayesian network. They use a loss function in order to optimize the parameters of the model.

3.2.3 Similarity learning

Similarity measures play a decisive role in the success or failure of building and using an object model. In high dimensional spaces, usual distances may fail to discern between features. This phenomenon is known as the *curse of dimensionality*. Yu *et al.* [Yu 2008] suggest learning the appropriate distance function to alleviate the problem of having feature elements from heterogeneous sources and which may have different influences on similarity estimation. They experiment the method on various applications (image retrieval, stereo matching and motion tracking) and benchmarks. Nowak and Jurie [Nowak 2007] propose learning a similarity measure that embeds domain specific knowledge. They state that standard distance functions, like the Euclidean distance in the original feature space, are often too generic and fail to encode this information. Jegou *et al.* [Jégou 2007] present a contextual dissimilarity measure (CDM) that takes into account the local distribution of the vectors and iteratively estimates distance correcting terms. In contrast to the ε-search framework where the distance is symmetric, in a k-NN framework, it is not. Here, the CDM tries to improve the symmetry of the k-neighborhood relationship such that the average distance of a vector to its neighborhood is almost constant. Frome *et al.* [Frome 2007] determine similarity between images by learning local distance functions and then choosing the globally consistent distance functions to use them for comparison at test time.

3.3 Scalability and prediction efficiency

Image search techniques and image object retrieval in particular aim to answer a user query in a reasonable time (less than two seconds) when querying large collections of image databases (up to several million images). However, rare are the datasets that address this issue (e.g. the ImageNet database with over $12,000,000$ images); most of them contain up to several thousand images (e.g. SUN (Scene UNderstanding) database with around $130,000$ images). The main problem, in reality, is the construction of a reliable ground truth for such a huge number of images.

Providing a relatively quick answer requires suitable access methods and scalable index structures. While text search engines can cope with very large databases, content-based image search hasn't yet gained enough maturity and is still facing new challenges, mostly because of the 2D-nature of images. In fact, tasks vary from retrieval to classification and detection, which provides an additional complexity to resolving similarity queries. Exhaustive search is the primitive technique to answer these queries; needless to say it is not efficient.

Most of the sophisticated techniques avoid a sequential scan of databases. Rather, they compress the feature space and use specific search methods. Two common types of queries are used: k-NN and range queries. A k-NN query consists in finding the k nearest neighbors of a given query object according to a given similarity measure. On the other hand, a range query consists in finding all elements within a given threshold (with respect to the query object).

Whether vectorial or metric, index structures usually rely on a *tree-like* method. Although they will not be dealt with here, some references are given instead for guidance [Comer 1979, Henrich 1989, Berchtold 1996, Henrich 1998, Berchtold 1998, Li 1999, Manolopoulos 2003, Datar 2004, Zhang 2004, Wang 2010]. Note that building an index structure has two types of cost: computation time and the disk size needed.

Part II

Contributions

Chapter 4

Learning Interpretable Models

A fact is a simple statement that everyone believes. It is innocent, unless found guilty. A hypothesis is a novel suggestion that no one wants to believe. It is guilty, until found effective.
Edward Teller

As previously stated, our thesis proposes a new supervised object retrieval method, called LARK for "LAsso-Regularized Keywords", based on discriminative visual keywords trained through a LASSO-regularized boosting algorithm. This chapter focuses on the learning stage (i.e. the supervised selection of relevant visual keywords from a set of labeled images). The following chapter focuses on the retrieval stage once the visual keywords of a given object class have been trained.

4.1 Towards interpretable visual keywords

4.1.1 Visual keywords

We consider a training set S of N images $(I_i)_{1 \leq i \leq N}$ provided with associated labels $l_i \in \{-1, 1\}$. A training image is labeled as a whole sample. It takes the label $+1$ if it contains the targeted visual object, and -1 if not.

By analogy with text documents (and as originally suggested by [Sivic 2003]), each image I_i is supposed to be described by N_i visual words. The definition of the visual vocabulary will be discussed later.

Our objective is to train a discriminative subset of visual words that are supposed to be the most representative words of the targeted visual object. We refer to these automatically selected visual words as *visual keywords*.

4.1.2 Equivalence between textual and visual keywords

Textual keywords carry information about the type of matter and the subject the document deals with. However, they don't express the writer's point of view. Similarly, our claim is that it suffices to determine a few visual keywords (of a given category) that allow the content of an image to be *interpreted* in order to correctly classify it. The task here is neither to determine the number of instances nor their relative poses and interaction with the rest of the other objects (In text documents, that would be equivalent to avoiding analyzing the writer's point of view and conclusions.). It is rather to label an image positively or negatively according to a visual category (In text documents, this is equivalent to knowing the document type and subject.).

It is certain that words in text documents have less structure than those in images. This shouldn't be a problem since our objective is retrieval. Therefore, there is no need to model relationships between object parts. Modeling these relations is an additive cost which is more appropriate for part identification and localization tasks.

4.1.3 Need for interpretability

Computer models are usually too abstract for users to understand where bad results might come from. By generating interpretable models, we try to create a link between the numerical representation of objects and our visual representation. Not only does interpretability enhance our understanding of output results, but it is also a very effective tool for user interactivity. It allows users to comprehend what the generic model is composed of and to choose, in different situations, the visual patches [1] that best match their needs. Therefore, interpretability is a means to achieve genericity. Users can perform object retrieval in large database collections which may contain heterogeneous data from different sources.

4.1.4 Requirements for interpretable visual keywords

As is the case with usual keywords (used for indexing and retrieving text documents), we would like our visual keywords to be easily interpretable by

1. The expression "visual keyword" and the word "patch" will be used interchangeably.

humans. We therefore introduce three requirements guiding the design of our method:

Readability: each visual keyword must be displayable, i.e. it has a uniquely defined visual representation that can be displayed in a GUI (typically as a thumbnail). The analogy to textual keywords would be that keywords must be readable regardless of whether they are understandable or not.

Conciseness: the set of the selected visual keywords must be as concise as possible. Conciseness is important for two reasons: first, it helps users to get a global overview from the very first glance and second, it increases the system's efficiency.

Disambiguation: each visual keyword must be as unambiguous as possible. Clearly, having a unique semantic meaning for each keyword is not realistic. Textual words themselves are known to be ambiguous (the same word having different meanings). Nonetheless, reducing the ambiguity of the visual keywords produced should remain a crucial objective towards interpretability.

4.1.5 Using common visual words

The first intuitive approach to solving our problem would be to rely on commonly used visual words. The related methods usually involve vector quantization of the visual space as a preprocessing stage that aims to reduce the visual vocabulary. This formalism may not satisfy our readability and disambiguation requirements because it discards the local geometric positions of the features being learned. In fact, and with such a representation, the only local information preserved is the centers of clusters formed from the feature space. From a user point of view, this information is not useful because they have no idea whether these centers pertain to tangible parts of the objects (i.e eye, tooth, finger, etc.) or if they are just a statistical combination of some of these parts; hence the non-readability. Moreover, and *assuming* that the readability condition is satisfied, each visual word might refer to distinct parts in distinct images, consequently increasing its ambiguity.

Commonly used visual words should be considered as syllables rather than words. Cluster centers then represent the common root of the words. For

example, and by analogy with text, the syllable "tor" can be the root of the words: torso, torsion and torpido,

Using such visual words as querying keywords was explored by Fauqueur [Fauqueur 2003] but with different local features and quantization (segmented regions, clustering). The centers of clusters are presented to users who are able to perform semantic queries through a Boolean composition. Logical queries are built by selecting—from the initial set—consistent regions and canceling out non-relevant regions. Besides non-readability, conciseness is not satisfied since all the centers of clusters are presented to users. In addition to that, these regions are generally textured which is considered to be a hindrance to clear interpretation.

Instead of using vector quantization, in our method, we choose to keep all local features as visual word candidates. Our representation is described in detail in section 4.2.

4.1.6 Using popular classifier models

To classify objects, many current state-of-the-art techniques use the bag-of-features method jointly with an SVM classifier [Lazebnik 2006, Nowak 2006, Yang 2007, Tirilly 2008, Dardas 2010, Raza 2010]. This method has proved to be very effective and achieved high classification scores, mostly because of the sparse representation it uses. However, it does not highlight the most interesting features that an object has. It discriminates objects by focusing on optimal separations in the feature space. In fact, the decision function of SVM classifiers is usually written as

$$f(x) = \sum_i \alpha_i \kappa(x_i, x) + b$$

where $\kappa(\cdot, \cdot)$ is the kernel function used to make a non-linear feature map. Moreover, generating a global descriptor for each image can be regarded as a weak point because it gives *global statistics* on the image scene. There is indeed no clear separation between an object and its context. Therefore, the model is stiff and cannot be used interactively. Furthermore, it is very likely that the performance drops when using test images with a different context than the context used in training (cf. figure 4.1).

(a) Example of a training set (b) Example of a test set

Figure 4.1: Images taken from the BelgaLogos dataset http://www-rocq.inria.fr/imedia/belga-logo.html (*Nike* category).

(a) Original image

(b) Zoom + crop

Figure 4.2: Contextual information enhances recognition ability

4.1.7 Contextual information

Before introducing the details of our method, we discuss here the added value of using contextual information. In fact, the object context sometimes plays a primordial role in recognition. Figure (4.2-a) gives an insight into human recognition ability. Thanks to the context, it is easy to tell that there is a plane in the sky. It is even possible to guess the pose and the principle direction. The same image is shown in figure (4.2-b) but with a zoom effect and a crop around the plane. Due to JPEG compression, the aliasing effect is now noticeable. It is obvious that the object which we have already interpreted is hardly recognizable. This shows that the information is not localized in the pixels themselves but rather in the manner they are grouped together and that the context is vital for recognition. As a rule of thumb, using varied backgrounds during training improves the generalization ability of the classifier [Ponce 2006].

4.1.8 Multiple-instance learning

Exploiting contextual information is possible by using a weak learning approach (also called *multiple-instance learning*: MIL). Indeed, each training image is labeled as a whole sample. MIL deals with uncertainty of instance labels (i.e. individual instances take no labels). An image is viewed as a bag of multiple features which are the local visual words. The bag will have only one label according to whether or not it includes at least one positive instance. It follows that it is only certain for a negative bag that there are no objects.

(a) Original (b) Resized

Figure 4.3: Importance of scale information

Using an MIL approach has the luxury of unsupervised learning (i.e unlabeled data). It gives more freedom to the algorithm to select contextual information whenever it turns out to be useful to characterize the category. Furthermore, labeling images with a detailed hand segmentation is very labor intensive. So, using a weak learning approach also helps to train images without much knowledge about the objects inside.

4.1.9 Local description and multi-features

There are various reasons why one should rely on local features. First of all, variation of small patches is more robust to a multitude of image transformations. Second, the foreground is usually accompanied by background clutter and it is necessary to find the useful information. Relying on local features that are more related to human vision should normally favor both interpretability and classification. Therefore, it is preferable to use local detectors that preserve scale information. Figure (4.3) is an example which illustrates how important scale information is: a person would not be able to easily identify the coral snake with a low scale.

Despite the fact that the method we propose is generic—in the sense that it could be used with any local features or patterns, in our work, we preferred

to use interest points (or structured sets of interest points) as the image primitives. In fact, multi-scale keypoint detectors have proven to be effective in matching applications [Nowak 2006]. They are preferred over edges or other larger features [Kerr 2008]. In addition, they can be extracted under severe transformations. On the other hand, contour techniques are usually used with images involving man-made objects (characterized by known geometric shapes) or with images that don't vary much in viewpoint (e.g. a horse taken from a side view). A shape-based detector will also fail to discriminate between categories originating from the same family as felidae within animals.

We believe that good recognition comes with a good description, specially one that uses multi criteria such as texture, scale and color. Image descriptors are indeed the raw material and the basic data for learning. In order to cover the difference in the nature of the objects to be learned and at the same time the intra-class variability of the same object, a multiple description scheme is needed. Moreover, the usefulness of a feature is often object-dependent and the best visual feature combination for classification could vary from object to object. It is then up to the learning algorithm to choose a descriptor or a combination of many descriptors that best suits a given category. Results reported by [liu 2007] show that the combination of various types of description, after an appropriate learning method, performs better than each single description alone. Similarly, [Xiao 2010] showed that the combination of all features outperform the state-of-the-art.

4.1.10 Feature appearance and model specialization

Our work can be linked to [Epshtein 2007] where a semantic hierarchy was constructed for object parts. The results showed that using varied appearances of the same object part significantly decreases the error rate. However, rather than modeling a hierarchy of all possible appearances, we rely on the learning algorithm to select appropriate visual keywords based on sparsity. Note that part-based approaches are usually sensitive to missed part detection. We want here to change this weak point into a rather powerful clue. In fact, users, generally speaking, tend to have some preferences about the images they are looking for. When querying the retrieval engine, they might want to emphasize the presence of some details (like a specific object part, its relative

scale, its context, etc.) and might be indifferent to other details. Allowing users to specialize their own model based on a variation of appearances will certainly provide better results.

To diminish the burden of computation complexity, the image representation we propose can be looked at as being part-based but with no geometry to model relationships between the parts. Our image representation is discussed in the next chapter.

4.2 The image representation proposed

For the sake of interpretability, the image representation is important because it is the key to building the components of the model. To reduce ambiguity, we consider the complete set of visual features included in all the training images *without* any quantization. We refer to our representation as bag-of-raw-visual-words (BoRW) to emphasize that all raw visual features are considered as visual words (without quantization).

A visual word σ_k is a structured set of local features: $\sigma_k = \{F_u, 1 \leq u \leq \varsigma\}$. A feature F_u is a vector signature characterizing a local area of the image. σ_k can be reduced to a singleton. In this case, we just refer to it as F_k. For now, and without loss of generality, we consider that each visual word is made up of one single local feature. Structured sets are discussed in section 4.4.

Our image representation satisfies the readability requirement and it is well suited to the disambiguation requirement. In fact, visual words are mapped to their exact geometric locations in the training images. Therefore, the models generated represent true real entities of what is described and they are not a vague approximation of the image content. Each image I_i is represented by a vector of size M features: $(F_k)_{1 \leq k \leq M}$ (cf. Figure 4.4). Note that the vector V may contain heterogeneous features, that is, features obtained with different types of description and/or with different dimensions thus resulting in various feature spaces. Even so, the method we propose is still applicable.

An object model is a weighted sum of our visual words. The models generated are extensible if we ever want to use additional training data. They are also shrinkable and can be modified according to the needs of a human operator. Users can query the retrieval engine using only the visual keywords

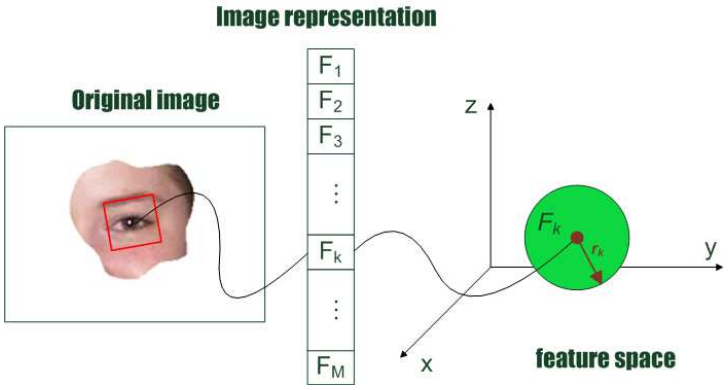


Figure 4.4: Image representation.

that they think are the best for their purpose. The next section discusses how these visual keywords are trained in order to satisfy our second requirement: conciseness.

4.3 Training discriminative visual keywords

4.3.1 Specificity of discriminative training

The choice of using a discriminative training approach over a generative approach stems from the observation that, in practical applications, objects are neither infinite nor isolated. Thus, they can be modeled in relation to one another. Take for example two human languages you have never heard before. And for a while, listen carefully to each language from a given audio sequence. After that, it is not usually difficult to correctly classify some *new* excerpts from these same languages. This proves that, during listening, people do not try to understand what is going on but rather try to find some characteristics (pitch values, repeated patterns, phonemes, etc.) that could help them to distinguish between the languages in question.

The same schema can be applied to object retrieval. In fact, we might be tempted to know the characteristics that the machine judged as being salient or object-related rather than considering them as an unknown black box. The next paragraph discusses how it is possible to bridge the semantic gap that could exist between human knowledge and the computational representation

of the models learned.

4.3.2 Keywords conciseness and model sparsity

Our claim is that conciseness—the faculty of being brief and informative—could be a result of using a sparse representation. Sparsity is also preferable because it reduces the complexity of the model and subsequently the prediction time. Furthermore, producing sparser solutions helps both researchers and users to understand what the model is composed of.

Constraining the loss function with an additive term, also known as regularization, is an effective tool to generate sparse models. Let $\beta = (\beta_1, \cdots, \beta_j, \cdots)^T$ be the vector of parameters to estimate (i.e. the weights of the visual words, initially zeros). We denote by $S_k = (I_k, l_k)$ a training image I_k labeled with $l_k \in \{-1, 1\}$. $S = \{S_1, \cdots, S_N\}$ represents the set of all the training data (N is the total number of the training images). The general form of the regularized loss function L_R can be written as

$$L_R(\beta, \lambda) = \sum_{n=1}^{N} L_C(S_n, \beta) + \lambda \cdot ||\beta||_\alpha \tag{4.1}$$

where L_C is a convex loss function, $\lambda \geq 0$ is a tuning parameter and $|| \cdot ||_\alpha$ is the α-norm.

To obtain sparse solutions, one should consider setting $\alpha \leq 1$. In fact, minimizing eq. 4.1 is equivalent to minimizing $\sum_{n=1}^{N} L_C(S_n, \beta)$ subject to $||\beta||_\alpha < t$ with $t \geq 0$. From a geometric perspective, the smaller α, the higher the chance that the solutions occur in intersections between the regularization volume and the space axes, leading to zero coefficients. Figure 4.5 presents unit balls for some values of α and figure 4.6 gives some insight into this idea.

In his thesis, Mairal [Mairal 2010] states that, to encourage sparsity, a natural choice would be to take $\alpha = 0$. Here, L_0 is the *pseudo*-norm originally proposed by Donoho and which counts the number of non-zero elements in β. He added that this solution is intractable because it necessitates the use of greedy algorithms. Note that the choice $\alpha = 1$ gives the smallest possible convex so it is a good compromise between generating sparse solutions and

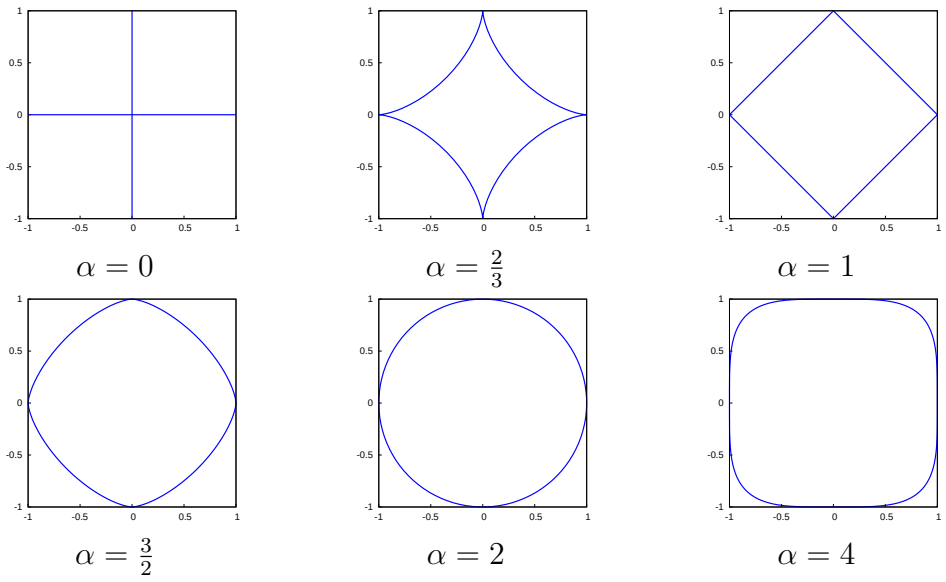
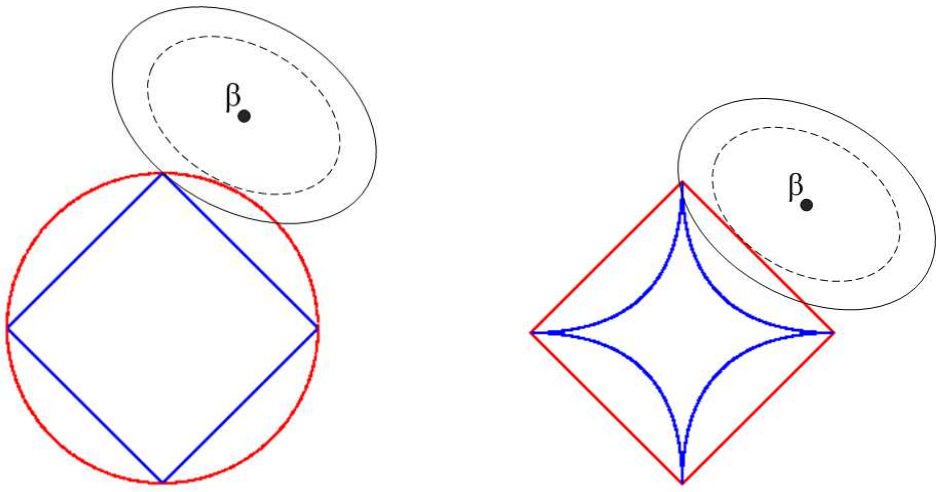


Figure 4.5: Unit balls for some values of α.



(a) An example where $\alpha = 1$ gives better shrinkage than $\alpha = 2$.

(b) An example where $\alpha = 0.5$ gives better shrinkage than $\alpha = 1$.

Figure 4.6: Illustration of the fact that it is preferable to choose a smaller α to get sparser solutions. The schemas are geometric representations in two dimensions showing a least square problem. Solutions occur where ellipses first hit the regularization area. The dashed ellipses show where the solutions would occur if a greater value of α was used. With a smaller α, some coefficients are set to zero.

keeping the minimization problem convex.

4.3.3 Background: training sparse models with boosting

Our method has its roots in boosting. The boosting mechanism was proposed by Schapire [Schapire 1990] in 1990. Since then, many algorithms have emerged [Freund 1997, Friedman 2000, Singer 2000, Tieu 2004, Opelt 2006, Saffari 2010] and boosting has become one of the most successful machine learning techniques. It was basically designed for classification tasks but it is also suited to regression. The theoretical foundations of boosting, including multiple references on the matter, can be found in [Meir 2003].

The underlying idea of boosting is to combine many weak classifiers—called hypotheses—in order to obtain one final "strong" classifier. Boosting is an additive model which builds up one hypothesis after another by reweighting the data for the next iteration—increasing the weights of misclassified images and decreasing those of well classified ones. This concept helps to generate different hypotheses, putting emphasis on misclassified examples, typically those located near the decision boundary in the feature space. In addition, boosting is able to build a model containing hypotheses of different natures in one learning stage. That is, the feature selection mechanism can process features which belong to different image descriptors. By the term "feature selection", we mean the process of selecting the most discriminant local signatures of the image.

Boosting has been considered as a stagewise gradient descent method in an empirical cost function. In particular, AdaBoost uses the exponential loss function [Friedman 2000, Rätsch 2001]. Nonetheless, this view is continuously brought into question. For example, Wyner [Wyner 2003] presents a boosting algorithm which performs empirically like AdaBoost while stabilizing the exponential loss to a constant. The reader may also refer to the discussion in [Mease 2008].

Although it is an intuitive algorithm, boosting may overfit the training data, particularly when it runs for a large number of iterations T in high dimensional space and with noisy data [Rätsch 2001, Grove 1998]. Moreover, a large value of T implies a long prediction time. On the other hand, setting

T to a small value may lead to underfitting. Therefore, the model may be non-discriminant, inconsistent and might not cover the variability inside the category itself. The boosting procedure can also be qualified as oblivious as it always functions in a forward manner aiming to minimize the empirical loss. Although the concept of re-weighting is interesting, at an iteration $t+1$, we have no idea whether the t previous generated hypotheses are good enough or not versus the model complexity.

4.3.4 Training sparse classifiers with LARK

In this section, we explain our method LARK in detail. We use the BoRW with a multiple-instance learning scheme that regularizes the loss function through Lasso.

4.3.4.1 Lasso

Tibshirani observed that the ordinary least squares minimization technique is not always satisfactory since the estimates often have a low bias but a large variance. In 1996, he came out with Lasso [Tibshirani 1996] which shrinks or sets some coefficients to zero. Lasso stands for Least Absolute Shrinkage and Selection Operator. The idea has two goals: first to gain more interpretation by focussing on relevant predictors and, secondly to improve the prediction accuracy by reducing the variance of the predicted values. The Lasso loss function Γ can be deduced from eq. 4.1 by setting α to 1 and L_C to the L_2 loss function. It is given by

$$\Gamma(\beta, \lambda) = \sum_{n=1}^{N} L_2(S_n, \beta) + \lambda \cdot ||\beta||_1 \tag{4.2}$$

Using the L_1 norm shrinks some coefficients and sets others exactly to zero, putting the emphasis on the most important features. As discussed in section 4.3.2, L_1 regularization is the minimal possible convex that can lead to sparse solutions, and keeps, at the same time, the optimization problem convex. This lies at the heart of the method's success.

Recall that $\lambda \geq 0$ is the parameter controlling the amount of regularization applied to the estimate. In order to obtain sparse solutions with an efficient

shrinkage tradeoff, λ usually takes a moderate value since a large one may set these coefficients to exactly zero, leading to the null model. On the other hand, setting λ to zero reverses the Lasso problem to minimizing the unregularized empirical loss. The general Lasso estimate $\hat{\beta}_\lambda$ is defined by:

$$\hat{\beta}_\lambda = \min_\beta \Gamma(\beta, \lambda) \tag{4.3}$$

Lasso minimizes the L_2 loss function penalized by the L_1 norm on the parameters. This is a quadratic programming problem with linear inequality constraints and it is intractable when the vector of parameters is very large.

In the literature, some efficient methods have been proposed to solve the exact Lasso namely the least angle regression by Efron *et al.* [Efron 2004] and the homotopy method by Osborne *et al.* [Osborne 2000]. These methods were developed specifically to solve the least squares problem (i.e. using L_2 loss). They work well when the number of predictors is small. However, they are not well-suited to nonparametric and classification tasks.

4.3.4.2 BLasso

Since the exact Lasso minimization is not tractable for a very large vector of parameters, Zhao and Yu [Zhao 2007] suggested using a boosting-like procedure called BLasso (Boosted Lasso). This algorithm tries to find the same solutions as Lasso with more cautious steps.

BLasso has been applied to various domains including text classification [Gao 2006], intestinal motility analysis [Igual 2007] and handwritten character recognition [Obozinski 2010]. However, and as far as we know, there have been no results reported on generic object recognition or retrieval.

The success of BLasso comes from its ability to converge to Lasso solutions while having the same computational advantages as boosting. In fact, it can deal with an infinite number of predictors and various loss functions. It can also perform variable selection given multiple image descriptors.

Unlike usual boosting procedures, and in order to approximate Lasso solutions, BLasso adds a backward step after each iteration of boosting. Thus, one is able to build up solutions in a coordinate descent manner and then take a look back at the consistency of these solutions regarding the model

complexity. Forward steps are used to minimize the empirical loss. On the other hand, backward steps minimize the regularization. In fact, at each iteration, a coordinate β_j is selected and updated by a small step size $\pm\varepsilon$ (with $\varepsilon > 0$). It has been shown [Zhao 2007] that it is preferable to choose a very small step size so that BLasso can approximate the Lasso path perfectly. In practice, ε should always be less than 0.1. Algorithm 1 gives an overview of the BLasso mechanism. Our learning algorithm will be reviewed in more detail in section 4.3.4.7.

Algorithm 1 BLasso

1. Initialization: $\beta = 0$
Make a forward step and initialize λ
2. Backward and forward steps:
Find the backward step that leads to the minimal empirical loss.
if the step decreases the Lasso loss **then** take it.
else make a forward step and relax λ if necessary
3. Repeat step 2 until $\lambda \leq 0$.

Note that a forward step consists in minimizing the current empirical loss. It changes one variable in the vector of parameters β by adding a value $\omega = \pm\varepsilon$. On the other hand, a backward step consists in finding the step (i.e. one of the previous forward steps) that leads to the minimal empirical loss. That is, to each non-null coordinate β_i, add the value $-\text{sign}(\omega)\omega$ while keeping all the other coordinates unchanged, then computing the empirical loss φ_i. After processing all non-null coordinates, find the variable that led to the minimal loss.

$$\hat{i} = \arg\min_i \varphi_i \tag{4.4}$$

4.3.4.3 Solving the multiple-instance problem

In our multiple-instance representation, we consider each image as a bag of instances (i.e a bag of local features). Only the image is labeled either positive or negative according to whether or not it contains the object. The instances themselves are not labeled and so we are not sure about their true labels. The task consists in selecting the positive instances because they represent some parts or a global view (in a coarser scale) of the object.

Since we don't know about the instance labels, one way to solve the feature selection problem is by discarding the instances that we are sure don't belong to the object. The idea is to reject any instance that belongs to a positive bag and that happens to exist in a negative bag. Hopefully, a sufficient number of training images would allow us to keep only the object features. Despite its simplicity, this idea cannot be directly applied to object categorization because the instances we are dealing with are visual features. In other words, given an instance in a bag, this same instance can exist in other bags under a different shape or appearance. This fact makes it impossible to identify negative instances; which leads us to the choice of using similarity measures. Knowing that each instance is a multidimensional vector in a high feature space, it is also difficult to define a good threshold of similarity due to the curse of dimensionality.

The solution proposed here is to rank the bags according to their best similarities to a given instance and then to attribute a score to this instance according to the ranking obtained (see the minimal distance matrix *section 4.3.4.5*). Scores must be attributed so as to favor the instances whose positive bags are ranked first. Here, and in order to assert the rejection of doubtful instances, it is necessary to rank a negative image before a positive image if these images have the same similarity score (or the same distance) to the instance.

Rejecting negative instances may theoretically keep several positive instances which results in a huge model. This is not practical if we wish to allow users to interact with the models. To remedy this problem, one must rely on a good feature selection mechanism that doesn't *overtrain* data. Overtraining should not be confused with overfitting. It simply means that the resulting models are very large and that similar prediction performance can be obtained with fewer instances.

4.3.4.4 Membership function

To decide whether or not a visual word F_k belongs to a given image, we must define a membership function. An intuitive choice would be to consider a usual distance function like the Euclidean distance. The distance between any feature F_{ki} belonging to the image I_i and any image I_j of the training set

is defined by:

$$\mathrm{d}(F_{ki}, I_j) = \min_{1 \leq k' \leq M_j} \mathrm{d}(F_{ki}, F_{k'j}) \tag{4.5}$$

For each F_{ki}, the images I_j are then sorted with increasing distances.

The choice of the minimal distance in eq. 4.5 is open to debate, but it is, in fact a good measure because it takes into account objects detected in a very low scale (i.e. the object is only described by one or two image features). However, it is not robust in the presence of outliers, noisy data and/or when the descriptors used are not good enough.

Besides common distances, we can define other membership functions that *don't have to satisfy* the requirements of a distance, namely non-negativity, symmetry, triangle inequality and the assertion that $\mathrm{d}(a, b) = 0$ iif $a = b$). In fact, we can think of any function that attributes higher positive scores to higher similarities. In this case, and for the sake of the genericity of the method, we can multiply each score by -1 to be able to rank images in an increasing order (according to best similarities). This kind of membership function is used in section 4.4. Nonetheless, in the following sections, and without loss of generality, we denote any membership function by $\mathrm{d}(\cdot, \cdot)$.

4.3.4.5 Definition of a weak classifier

The final classifier (also called *strong* classifier) is a weighted sum of the weak classifiers learned during training. A weak hypothesis h_k (i.e. *weak classifier*) represents a coordinate of base learners. Its weight is strictly positive if it was chosen at least once during the boosting process and remains zero if not. In the context of object retrieval and categorization, the weak hypothesis h_k is a visual word F_k with a particular description ρ and which has an optimal radius r_k (Opelt *et al.* [Opelt 2006]). h_k is viewed as a hypersphere centered on the local image feature F_k (cf. figure 4.4). For a test image x, h_k will output $+1$ if the distance between x and F_k is less than r_k and -1 otherwise:

$$h_k(x) = \mathrm{sign}\big(r_k - \mathrm{d}(x, F_k)\big) \tag{4.6}$$

Choosing hyperspheres over other representations like hyperplanes and decision stumps gives an easier interpretation to image features. It also guarantees a fast classification since we just need to compare the distance $\mathrm{d}(x, F_k)$

to r_k.

The radius r_k of a hypersphere is computed such that the classification error of F_k is as small as possible. In order to compute the optimal r_k, we need to precompute a minimal distance matrix between any image feature and the training images themselves. Let $\Omega = \{e_1, \cdots, e_E\}$ be the set of all types of descriptions used and $V_{i,\rho} = \{F_{i,\rho,k}; 1 \leq k \leq M_{i,\rho}\}$ the set of the local signatures belonging to the image I_i according to the description e_ρ. Now, $\forall i \in \{1, \cdots, N\}$ and $\rho \in \{1, \cdots, E\}$ $\forall k \in \{1, \cdots, M_{i,\rho}\}$

- **Minimal distance matrix:** $\forall j \in \{1, \cdots, N\}$ compute the minimal distance $d_{i,\rho,k,j}$ between $F_{i,\rho,k}$ and the image I_j (i.e. the closest distance to I_j)

$$d_{i,\rho,k,j} = \min_{1 \leq k' \leq M_{j,\rho}} \mathrm{d}(F_{i,\rho,k}, F_{j,\rho,k'}) \tag{4.7}$$

- **Sorting:** Let s be a permutation such that

$$d_{i,\rho,k,s(1)} \leq \cdots \leq d_{i,\rho,k,s(N)} \tag{4.8}$$

  Consequently, images are sorted increasingly according to their distances to $F_{i,\rho,k}$
- **Radius computation:** Select the index $\hat{\eta}$ where the sum of image labels is maximum

$$\hat{\eta} = \arg \max_{1 \leq \eta \leq N} \sum_{j=1}^{\eta} l_{s(j)} \tag{4.9}$$

  The hypothesis radius $r_{i,\rho,k}$ is then given by:

$$r_{i,\rho,k} = \frac{d_{i,\rho,k,s(\hat{\eta})} + d_{i,\rho,k,s(\hat{\eta}+1)}}{2} \tag{4.10}$$

The distance matrix may sometimes be too large. This usually happens when the training set comprises many images and/or when there are many descriptions used. Two major problems are generated in this case: an excess use of the main computer memory and a slowness in the learning algorithm to determine the best hypothesis at each boosting iteration. One solution consists in reducing the size of the distance matrix by getting rid of the "bad" entries that we know will never be selected in the learning stage. For example, this is the case when the closest neighbor to a query feature belongs to a

negative training image. It is certain that this kind of visual word will never generate good hypotheses. In addition, and as a second filter, we can apply the distance ratio proposed by [Lowe 2004] to discard bad matches. The method consists in rejecting a feature when the ratio between the distance to its first closest neighbor and the distance to its second closest neighbor is above some threshold. However, since a good match can be found in many positive images, the second closest distance is defined as the first closest distance to a negative image. Winder and Brown [Winder 2007] reported that this simple optimization produces a boost in performance. [Mikolajczyk 2005a] showed that the matching gained by the nearest neighbor distance ratio varies according to the description used. In any case, there are fewer false matches and precision is improved.

In our experiments, we used hard weak classifiers as defined by eq. 4.6. We can also define soft weak classifiers. Consider, for example, the expectation $\mathrm{E}\big(h_k(i)\big)$ of the classification of the image i, which measures how likely an image contains the visual word F_k. Take all the radii r_{ki} of F_k computed from the distance matrix. The task is to minimize the probability p_k that the image does *not* contain the feature F_k

$$p_k = \min_i p_{ki} \tag{4.11}$$

where p_{ki} is the probability that the image does not contain F_k within the radius r_{ki} It is given by

$$p_{ki} = \frac{(\nu - \tau\nu)!\ (\nu - \nu^+)!}{\nu!\ (\nu - \tau\nu - \nu^+)!} \tag{4.12}$$

where $0 < \tau \leq 1$ is a constant *repeatability* parameter, ν is the number of features in i and ν^+ is the number of the features in i within the radius r_{ki}. $\mathrm{E}\big(h_k(i)\big)$ is written as

$$\mathrm{E}\big(h_k(x)\big) = 1 - 2 \cdot p_k \tag{4.13}$$

4.3.4.6 Definition of the loss function and weight updates

Since the exact minimization of the loss function with a very large number of base learners is hardly practical, boosting-procedures try to find the solution with an iterative procedure. At each iteration, a weak hypothesis is

chosen such that the strong classifier converges to the optimal solution. For classification tasks, various convex loss functions have been used such as exponential loss, logit loss, binomial deviance, etc. In our method, we use the exponential loss function L_e as in AdaBoost. Let T be the maximum number of iterations. At an iteration $t+1 \leq T$, one minimizes:

$$L_e^{(t+1)}(\beta_j) = \sum_{n=1}^{N} \exp(-l_n \cdot F_{\beta_j}^{(t+1)}(I_n)) \tag{4.14}$$

where $F_{\beta_j}^{(t+1)}(I_n) = \beta_j \cdot h_{t+1}(I_n) + \sum_{k=1}^{t} \beta_k \cdot h_k(I_n)$ is the set of ensembles of base learners.

$$\beta^{(t+1)} = \beta^{(t)} + \beta_j \cdot \mathbf{1}_j \tag{4.15}$$

and $\mathbf{1}_j$ is the j^{th} standard basis vector with all 0's except for 1 in the j^{th} coordinate.

After a boosting iteration, the training data are re-weighted. Initially, all image weights are set to $\frac{1}{N}$ (N is the number of the training images). Weights are updated so as to emphasize the misclassified images. The optimal solution to minimizing L_e is $\hat{\beta}_j$ such that

$$\hat{\beta}_j = \frac{1}{2} \log \frac{1 - \varepsilon_{t+1}}{\varepsilon_{t+1}} \tag{4.16}$$

where

$$\varepsilon_{t+1} = \sum_{\substack{n=1 \\ l_n \neq h_{t+1}(I_n)}}^{N} {w_n}^{(t+1)} \tag{4.17}$$

is the weighted training error (w_n is the weight of the image I_n). AdaBoost runs until it reaches T iterations and stops earlier if $\varepsilon = 0$ or $\varepsilon \geq 0.5$.

Boosting has been interpreted as a gradient descent method. Equation (4.16) gives the optimal $\hat{\beta}_j$ that allows the algorithm converge as fast as possible (i.e. steepest descent). This formulation is utilized in AdaBoost. On the other hand, other varieties of forward stagewise additive modeling algorithms take more steps to converge but usually outperform the steepest descent method in prediction. Forward Stagewise Fitting (FSF, also called e-boosting) is one example. It adds new coefficients to the previous set with an infinitesimal fixed step size $\varepsilon > 0$. Yet it is unclear what criteria FSF

optimizes. At each iteration, a coordinate is chosen and updated by $\pm\varepsilon$. The fact that ε is very small imposes a local shrinkage on the variables. Hastie *et al.* [Hastie 2009a] (section 16.2) showed that forward stagewise can sometimes (but not always) approximate the effect of Lasso. Similar observations were noticed by Zhao *et al.* [Zhao 2007]. Their simulations concluded that FSF local regularization does not converge to the Lasso path in general. They also added that FSF solutions are less sparse than Lasso. To remedy this problem, they introduced the concept of backward steps which minimize the Lasso loss (i.e BLasso). By allowing backward steps, the algorithm goes back and forth in order to optimize the trade-off between penalty and empirical loss.

4.3.4.7 LARK: the algorithm

Our objective is to minimize

$$\Gamma(\beta, \lambda) = \sum_{n=1}^{N} L_e(S_n, \beta) + \lambda \cdot ||\beta||_1 \tag{4.18}$$

and our method is based upon BLasso.

BLasso adds a backward step to take into account the regularization term in equation (4.18). Forward steps, however, are chosen so as to minimize the empirical loss of the training samples. Both of these steps use the same loss function.

Unlike BLasso, LARK's forward steps always add a positive value. Consequently, backward steps are those that shrink the model by subtraction. The choice made here is justified by the fact that the distance matrix defined earlier (cf. section 4.3.4.5) is computed only for the features belonging to the positive examples. Therefore, a selected hypothesis represents a visual feature that contributes to building the model of the object category. It must therefore have a positive weight. The reason for not computing the distances between negative features and the training images is that it does not help either user interactivity (i.e. ambiguity to define what a negative visual word is) or genericity (a negative visual word may be representative of one dataset but could not generalize to other databases). However, it is possible to learn a negative model against the object category. This idea is discussed in section 6.5.

In order to minimize the empirical loss, LARK uses a *weighting scheme* as in AdaBoost. In fact, adopting this strategy is more appealing. First because it is much faster and second, it benefits from the weight change. The slowness to overfitting of AdaBoost has been observed over recent years, and this property is incontestable. One of the reasons is that, at each iteration, AdaBoost gives more attention to the misclassified observations by increasing their respective weights. In order to take advantage of this principle, we decided to keep the same mechanism to select forward steps. At an iteration t, we compute the score $s_{i,\rho,k}$ of the base learner $(F_{i,\rho,k}, r_{i,\rho,k})$ based on the image weights

$$s_{i,\rho,k} = \max_{m} \sum_{c=1}^{m} w_{s(c)} \cdot l_{s(c)}$$

Then, we select the base learner which obtains the highest score. In fact, when dealing with a very large or an infinite number of base learners, it is impractical or even impossible to try to minimize the loss function directly. In their seminal work, Opelt *et al.* [Opelt 2006] used a representation with infinite base learners. The radius $r_{i,\rho,k}$ of each weak hypothesis is not fixed *a priori* but takes into account the weight changes during AdaBoost iterations. It is computed after the algorithm selects the feature $F_{i,\rho,k}$. Equation (4.9) then becomes:

$$\hat{\eta} = \arg \max_{1 \leq \eta \leq N} \sum_{j=1}^{\eta} w_{s(j)} \cdot l_{s(j)} \tag{4.19}$$

In LARK, since each visual word $F_{i,\rho,k}$ contributes to the final model by at most one hypothesis, we averaged out the different radii—of a given visual word—that were computed during forward steps. After LARK stops, the hypothesis radius is given by

$$r_{i,\rho,k} = \frac{\varepsilon}{\beta_{i,\rho,k}} \sum_{t} r_{i,\rho,k,t} \tag{4.20}$$

where $r_{i,\rho,k,t}$ is the radius of $F_{i,\rho,k}$ computed at a previous iteration t.

Algorithm 2 summarizes our proposed feature selection mechanism. It has one input parameter $\xi > 0$ which is used as a tolerance level.

In order to illustrate the principle of BLasso, we present a simplified diagram (cf. figure 4.7) that shows how the procedure works. Without loss of

Algorithm 2 LARK: The proposed learning algorithm

1. Initialization: set $\beta = 0$ and $t = 1$
   - Set $w_n^{(1)} = \frac{1}{N}$ for $n \in \{1, \cdots, N\}$
   - Train the classifier and find the best hypothesis $h_\kappa^{(1)}$ (κ is the index which corresponds to the κ^{th} entry of the vector β)
   - $\hat{\beta}^{(1)} = \varepsilon \cdot \mathbf{1}_\kappa$
   - Calculate the initial regularization parameter

$$\lambda_1 = \frac{1}{\varepsilon}\big(\sum_{n=1}^{N} L(S_n, 0) - \sum_{n=1}^{N} L(S_n, \hat{\beta}^{(1)})\big) \tag{4.21}$$

   - Set the active index set $I_A^{(1)} = \{\kappa\}$
2. Backward and forward steps
   Find the backward step that leads to the minimal empirical loss.

$$\hat{j} = \arg\min_{j \in I_A^{(t)}} \sum_{n=1}^{N} L(S_n, \beta^{(t)} - \varepsilon \cdot \mathbf{1}_j) \tag{4.22}$$

   This step is taken if it helps to decrease the Lasso loss. In other words: If

$$\Gamma(\beta^{(t)} - \varepsilon \cdot \mathbf{1}_{\hat{j}}, \lambda_t) - \Gamma(\beta^{(t)}, \lambda_t) \leq -\xi \tag{4.23}$$

   then

$$\beta^{(t+1)} = \beta^{(t)} - \varepsilon \cdot \mathbf{1}_{\hat{j}} \; ; \; \lambda_{t+1} = \lambda_t$$

   Otherwise, we force a forward step and relax λ if necessary.
   - Update weights

$$w_n^{(t+1)} = \frac{w_n^{(t)} \cdot \exp\big(-\varepsilon \cdot l_n \cdot h_\kappa^{(t)} I(n)\big)}{\tau} \tag{4.24}$$

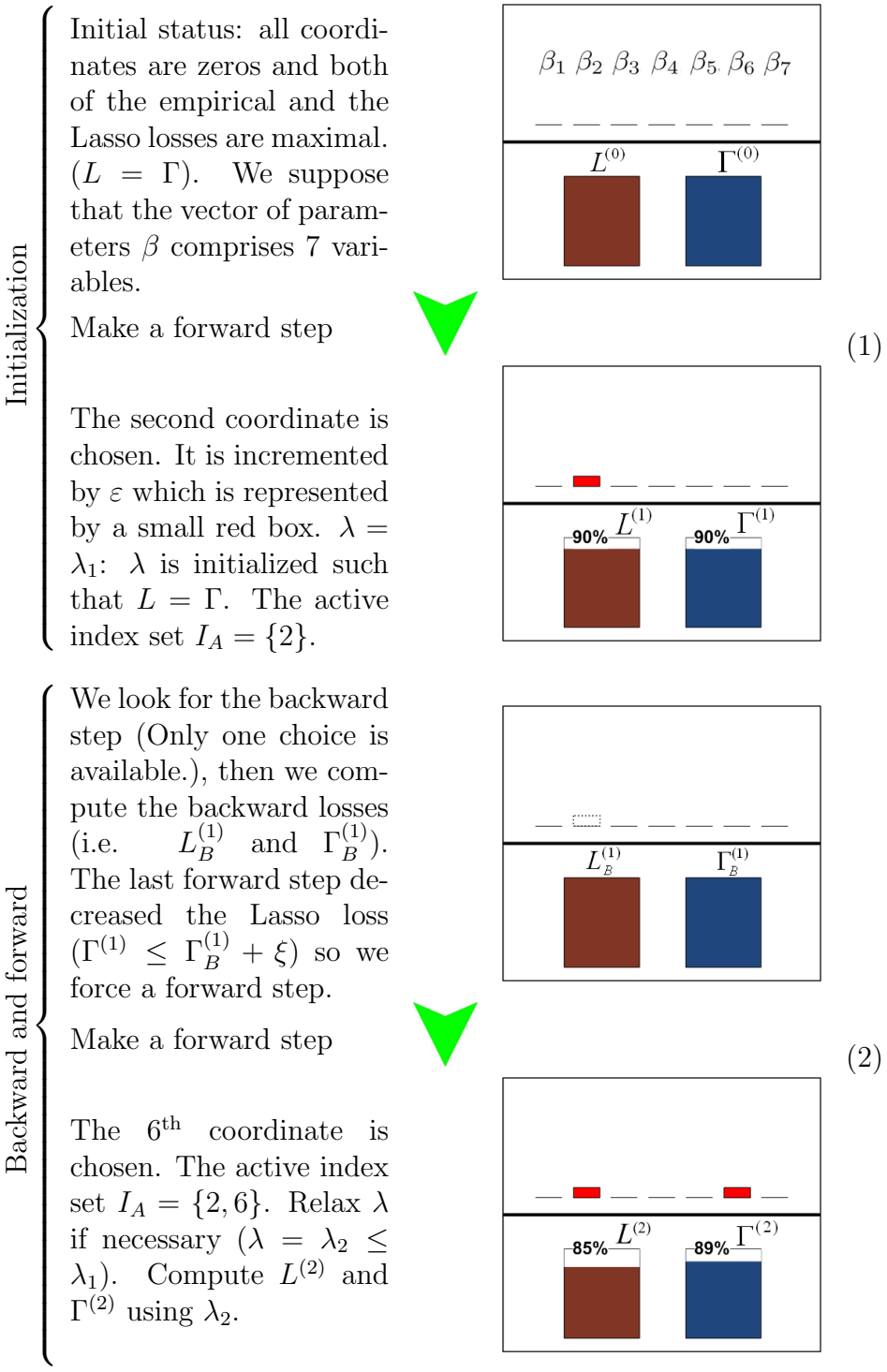
     where τ is a normalization constant such that $\sum_{n=1}^{N} w_n^{(t+1)} = 1$
   - Train the classifier and get the best hypothesis $h_\kappa^{(t+1)}$
   - $\hat{\beta}^{(t+1)} = \hat{\beta}^{(t)} + \varepsilon \cdot \mathbf{1}_\kappa$
   - $\lambda_{t+1} = \min\Big[\lambda_t, \frac{1}{\varepsilon}\big(\sum_{n=1}^{N} L(S_n, \hat{\beta}^{(t)} - \sum_{n=1}^{N} L(S_n, \hat{\beta}^{(t+1)} - \xi)\Big]$
   - $I_A^{(t+1)} = I_A^{(t)} \cup \{\kappa\}$
3. Increase t by one and repeat steps 2 and 3 until $\lambda_t \leq 0$

generality, we suppose that all the coordinates estimated have a positive value (as in LARK). In other words, the forward steps are always positive ($w = \varepsilon$), therefore, a backward step always decreases the coordinate found by a ε. Note that BLasso uses a tolerance parameter $\xi > 0$ to gain more stability. ξ should be set to a very small value. The figure is made up of repeated blocks of backward and forward steps. Each step uses boxes to encapsulate the status of the coordinates $(\beta_i)_i$ as well as the empirical loss L and the Lasso loss Γ. In addition, these losses are shown as a percentage of the initial loss. The numbers given here are not related to any real run with concrete data, but they are chosen in order to illustrate the behavior of BLasso in different situations.

4.3.5 Efficient implementation

The greediest block in the BLasso algorithm is the computation of the loss function. In fact, we need to compute both the empirical loss and the the Lasso loss many times during each iteration. The complexity increases every time BLasso chooses a new coordinate that was not selected before. Luckily, the Lasso loss can be deduced from the empirical loss. It follows that, in order to compute the current Lasso loss and the backward Lasso loss, we just need to compute their respective empirical losses. On the other hand, and at each iteration, only one coordinate is modified while all the other coordinates remain unchanged. Therefore, assuming that memory is cheaper than processing time, the computation of the empirical loss can be speeded up—at an iteration t—by keeping in memory the classification values of the previous state for each coordinate β_j and for each image I_n. The memory in use depends on the number of images in the training set as well as the number of the variables selected by the learning algorithm at a given step. Since BLasso favors sparsity, the number of hypotheses selected will be small enough for this method to be still applicable, even with a large number of images in the training set. Each image I_n will have a storage vector ζ_n. To simplify the notation, we will consider that the j^{th} entry of the vector ζ_n (i.e. $\zeta_n(j)$) refers to the classification value of the variable β_j

$$\zeta_n(j) = \beta_j \cdot h_j(I_n) \qquad 1 \leq n \leq N; j \in I_A \tag{4.25}$$

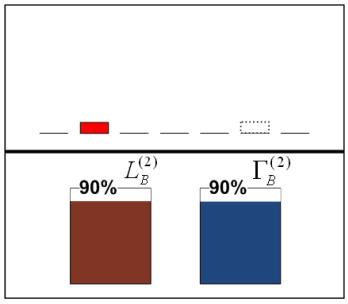


Algorithm 3 summarizes the use of ζ_n for computing the empirical loss.

Thus, when changing a hypothesis h_k in the next iteration, we only need

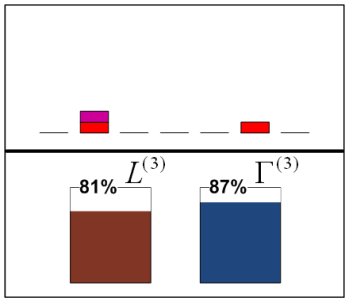
Backward and forward

There are two possibilities for the backward step. Take the one that corresponds to the minimal empirical loss then compute the backward Lasso loss using λ_2 (say the 6^{th} coordinate). We have $\Gamma^{(2)} \leq \Gamma_B^{(2)} + \xi$



Make a forward step

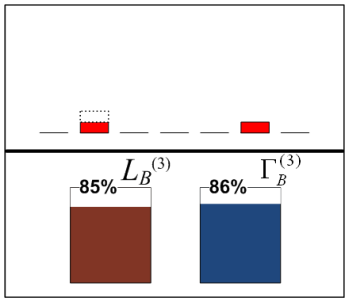
The second coordinate is chosen again. $I_A = \{2, 6\}$. Relax λ if necessary ($\lambda = \lambda_3 \leq \lambda_2$). Compute $L^{(3)}$ and $\Gamma^{(3)}$.



(3)

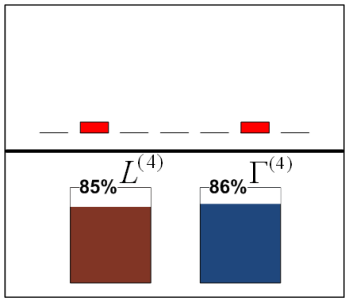
Backward and forward

Find the backward step and compute $\Gamma_B^{(3)}$ using λ_3. It is correct to suppose that that $\Gamma_B^{(3)} < \Gamma^{(2)}$ because $\lambda_3 \leq \lambda_2$ (even though β is the same). The current Lasso loss $\Gamma^{(3)}$ is however greater than the backward loss so we take the step.



Take the backward step

$\lambda_4 = \lambda_3$. The active index set $I_A = \{2, 6\}$. $\Gamma^{(4)} = \Gamma_B^{(3)}$ $L^{(4)} = L_B^{(3)}$



(4)

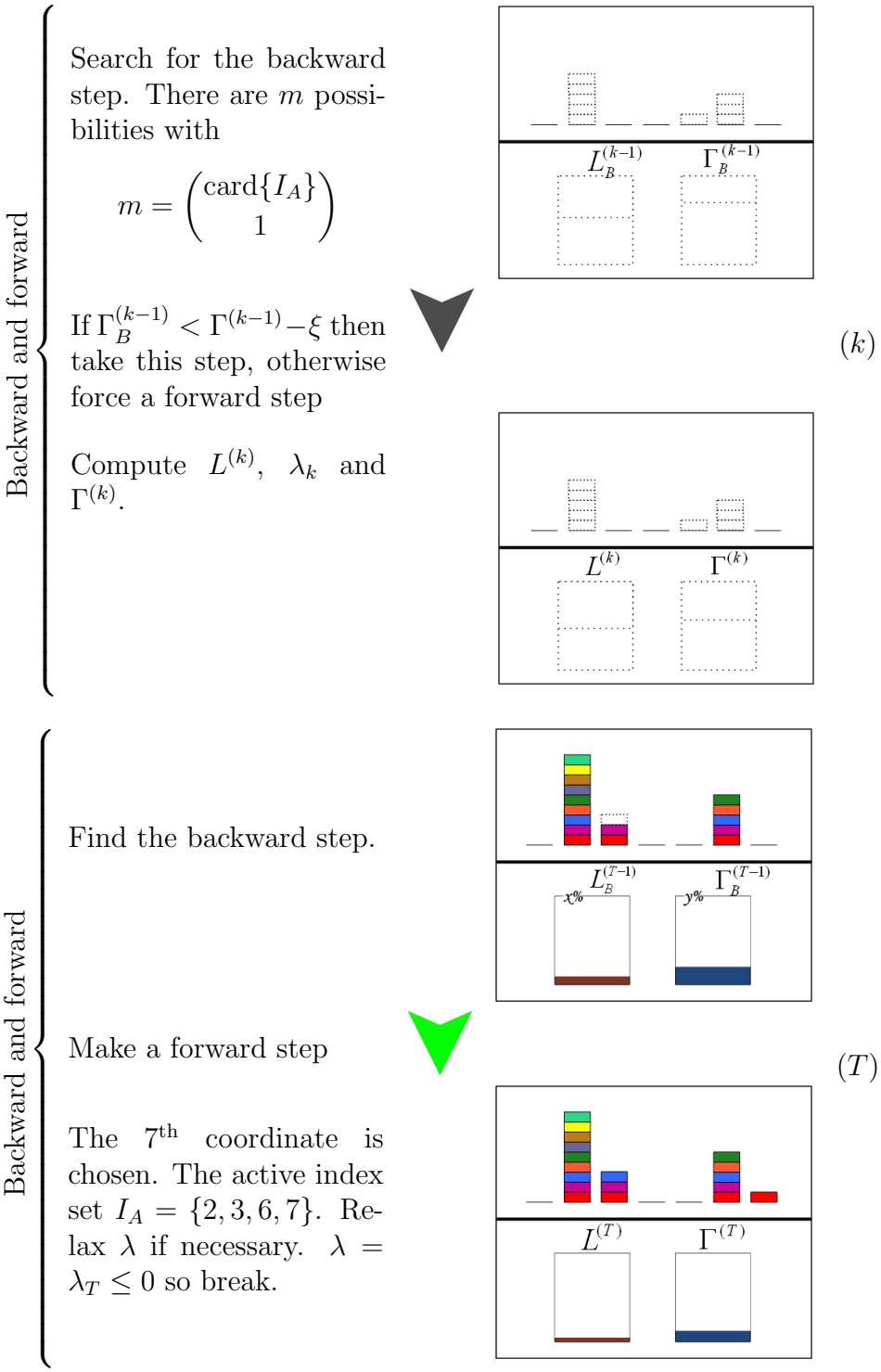


Figure 4.7: Diagram of BLasso.

Algorithm 3 Exponential empirical loss

$C \leftarrow 0$
For n=1 **to** N **do**
  $c_n \leftarrow 0$
  For_each j **in** I_A **do**
    $c_n \leftarrow c_n + \zeta_n(j)$
  Done
  $C \leftarrow C + \exp(-l_n \cdot c_n)$
Done

to update (if β_k has already been selected before) or create new (if the index k is selected for the first time) N classification values. That is, we have to compute $\zeta_n(k)$ for all $1 \leq n \leq N$

The storage of the classification values is more important when searching for the backward step. The empirical loss is not computed just once but at least card$\{I_A\}$ times where card$\{I_A\}$ is the cardinal of the index set (It is computed card$\{I_A\}+1$ times if the coordinate is selected for the first time.) Given an image I_n, for each coordinate j, we need to compute the empirical loss based on the coordinates $\beta - \varepsilon \mathbf{1}_j$. Apart from the coordinate β_j, the other classification values have already been computed and stored in $\zeta_n(k)$ with $k \neq j$. Moreover, the absolute difference between $\zeta_n(j)$ (already computed if the hypothesis is old) and the classification value δ_{nj} that we need to compute is ε

$$\text{if } \begin{cases} \zeta_n(j) < 0 & \Rightarrow \quad \delta_{nj} = \zeta_n(j) + \varepsilon \\ \zeta_n(j) > 0 & \Rightarrow \quad \delta_{nj} = \zeta_n(j) - \varepsilon \end{cases} \tag{4.26}$$

It follows that the classification value of the image I_n (i.e. $\sum_k \zeta_n(k)$) will only change by an absolute difference of ε. We denote by $\overline{\zeta}_n(j) = \sum_{k \neq j} \zeta_n(k) + \delta_{nj}$, that is:

$$\overline{\zeta}_n(j) = \begin{cases} \sum_k \zeta_n(k) + \varepsilon & \text{if } \zeta_n(j) < 0 \\ \sum_k \zeta_n(k) - \varepsilon & \text{if } \zeta_n(j) > 0 \end{cases} \tag{4.27}$$

This formulation helps to locate the backward step quickly. In fact, it only takes a linear time according to the number of the selected hypotheses. After subtracting the value ε from the coordinate β_j, the empirical loss E_j is

computed as follows:

$$E_j = \sum_{n=1}^{N} \exp(-l_n \cdot \overline{\zeta}_n(j)) \tag{4.28}$$

The backward step $\hat{j}$ is then defined by

$$\hat{j} = \arg\min_j E_j \tag{4.29}$$

When the algorithm proceeds and selects a coordinate g at the iteration t, the stored values will be altered as follows. If g is selected for the first time, then

$$\forall n \qquad \overline{\zeta}_n(g) = \sum_{k \neq g} \zeta_n(k) \quad ; \quad \zeta_n(g) = \varepsilon \cdot h_g(I_n) \tag{4.30}$$

and

$$\overline{\zeta}_n^{(t)}(j) = \overline{\zeta}_n^{(t-1)}(j) + \varepsilon \cdot h_g(I_n) \qquad \forall j \neq g \tag{4.31}$$

Now, if g already belongs to the active index I_A, then

$$\zeta_n^{(t)}(g) = \zeta_n^{(t-1)}(g) + \text{sign}(\zeta_n^{(t-1)}(g)) \cdot \varepsilon \tag{4.32}$$

Equation (4.31) is valid for all $j \in I_A$. Note that when $j = g$, this equation automatically takes into account the backward step (i.e the term $-\varepsilon \cdot \mathbf{1}_g$) because it was not added in the first place.

4.4 Including local geometric constraints

Feature selection, at each boosting iteration, is achieved according to the highest score of visual words. These scores are a measure that reflects how reliable a given feature is. They take into account image weights, but more importantly, they are computed with respect to the ranking of the training images (cf. Eq. 4.8) and don't need any additional information. Recall that image ranking is possible thanks to the membership function that computes similarities. Furthermore, each type of description e_ρ may have its own membership function which is the best suited. Hence, and more specifically, various distance functions (L_1, L_2, Mahalanobis, etc.) can be used simultaneously. Also note that, for each membership function, distance computation

can vary from exhaustive to more advanced accelerated techniques.

So far, each visual word has been considered to contain a unique local image feature. In this section, each visual word is a structured set of local image features as defined in section 4.2. Particularly, a visual word contains some interest points located within a neighborhood, and that *preferably*, capture the same structure. This kind of word is very helpful when there exist repeated patterns and/or when dealing with rigid objects.

The question on how to construct a query set from local image features may be resolved in various ways. However, and in order to form a structured set, local image features have to satisfy one or many neighborhood constraints related to the image space (and *independently* from the feature space). Therefore, we can ensure that the visual word concept still holds. Forming bigger image patches may transform these local patches into semi-local or even "global" patches. By analogy with text documents, a word may turn into a phrase or a sentence. This statement stresses that "big" visual words may include excessive information to the point where they become inflexible, hard to manipulate and are deemed to fail if glued together to build a richer semantic concept. In general, if the scale information of a local feature F is available, we think that is a good idea to use a surrounding region defined by a multiple factor of the scale, say three or four times. The query set is then formed from all the features belonging to this region. Otherwise, we may think of considering the kNN features of F. In reality, the scale information is preferable because it captures the local characteristics of a detected feature. It is also a good means to extract the *same* structure from different images.

Rather than using a metric distance for comparison, ranking images is achieved through a similarity matching score that summarizes geometric coherence between the *query set* (i.e. the visual word) and the best set that matches in a particular image x. This score is computed in two steps:

Feature matching Each feature F_{ku} (belonging to a visual word $\sigma_k = \{F_{ku}, 1 \leq u \leq \varsigma\}$) is matched in the database thanks to an efficient approximate similarity search [Joly 2008].

Geometric consistency Using RANSAC, and for each image x_i, we compute a geometric consistency score by estimating an affine transformation model $(\mathbf{A}_{ki}, \mathbf{B}_{ki})$, with six degrees of freedom, between σ_k and the

matched points in x_i.

We denote by $\Psi_k = \{P_{ku}, 1 \leq u \leq \varsigma\}$ the spatial points corresponding to the visual word σ_k and $\Upsilon = \{Q_{ku}, 1 \leq u \leq \upsilon\}$ the spatial points matched. We have $\upsilon \leq \varsigma$. Without loss of generality, we suppose that the point P_{ku} matched the point Q_{ku} for $u \in \{1, \cdots, \upsilon\}$. The matching score Ξ between σ_k and x_i corresponds to the number of inliers in the model. It is given by

$$\Xi(\sigma_k, x_i) = \sum_{u=1}^{\upsilon} \delta\big(||P_{ku} - \mathbf{A}_{ki} Q_{ku} + \mathbf{B}_{ki}|| \leq t\big) \tag{4.33}$$

with

$$\delta(d \leq t) = \text{sign}(t - d)$$

and t is a fixed tolerance threshold ($t = 7$).

Recall here that if two or more images had the same score, negatives would have to be ranked first (see *Solving the multi-instance problem*, section 4.3.4.3).

Chapter 5

Interactive Retrieval

To manage a system effectively, you might focus on the interactions of the parts rather than their behavior taken separately.
Russell L. Ackoff

RETRIEVING visual objects consists in attributing—to *unseen* images—scores according to the likelihood that they contain a particular object. The scores are obtained by comparing the visual keywords composing the object model with each image. Interactivity, however, adds the possibility to adjust the model according to user needs. The model itself has to have a clear view. In the next section, we provide detail about how to compute image scores.

5.1 Prediction

The strong classifier is the predictive function that computes scores. It may involve hypotheses originating from various types of descriptions $(e_{c_i})_{1 \leq i \leq G}$ with $G \leq E$ and can be written as:

$$H = \sum_j w_j h_{j,e_{c_1}} + \sum_j w_j h_{j,c_2} + \cdots + \sum_j w_j h_{j,e_{c_G}}$$

where each $h_{j,e_{c_i}}$ is a weak classifier (cf. *definition of a weak classifier* section 4.3.4.5) associated to one single visual keyword. Note that H is a real-valued classifier that allows images to be ranked according to their scores.

We aim to compute the classification value of a given test image x from a retrieval point of view. There are two different ways to proceed: either by means of range queries or by using an exhaustive search. In any case, we need to compute the output of every weak hypothesis $h_{j,e_{c_i}}$ which is represented by the couple $(F_{k,e_{c_i}}, r_{k,e_{c_i}})$. The cost of the prediction time is based on the

number of distances to compute. Hence, the time complexity is linear with respect to the number of images (since images contain almost the same number of local features).

In the exhaustive mode, we begin by computing the minimal distance (using the membership function of the description e_{c_i}, and which can be either a usual distance or a matching score that takes into account local geometric constraints, cf. sections 4.3.4.4 and 4.4) between x and any $F_{k,e_{c_i}}$. Only the features that belong to x and with the description e_{c_i} are concerned:

$$d_j = \min_{F_{z,e_{c_i}} \in x} \mathrm{d}(F_{z,e_{c_i}}, F_{k,e_{c_i}})$$

The output of $h_{j,e_{c_i}}$ is then given by the formulation in equation (4.6) and the classification value of x is the weighted sum of all the outputs of the weak classifiers. The higher this value, the greater the likelihood that the image contains a given object. This formulation favors retrieving images according to the number of the visual patches they contain. It is, however, difficult to learn an effective threshold that separates object images from non-object images since the model may be specialized by users—implying that the number of visual keywords selected, as well as their confidence measures, may vary. Consider the case where the model comprises two visual keywords, each having the same weight 1. If an image contains both of the patches, its score will be 2 and vice versa. Inversely, if it doesn't contain either, its scores will be -2, and if it contains only one patch, its score will be zero. Now if we take the first case and add to the model eight other visual keywords having the same weight 1, the score of an image containing only two relevant patches is -6. Comparing with the first case when the image has a score of 2, we see the large interval the scores may belong to. Nonetheless, defining a fixed retrieving threshold is possible when considering normalization.

When using an index structure, the same scores are obtained but we proceed differently. Rather than predicting each image separately, all images in the database are predicted at the same time. In fact, we begin by initializing the prediction values of all the test images to zero. After that, for each weak hypothesis $h_{j,e_{c_i}}$, we perform a range query over the database. That is, we query the search engine to retrieve all the images that fall into the hypersphere

$r_{j,e_{c_i}}$ defined by $h_{j,e_{c_i}}$.

$$\Delta = range_{db}(F_{j,e_{c_i}}, r_{j,e_{c_i}}) = \{F \text{ s.t. } \mathrm{d}(F_{j,e_{c_i}}, F) < r_{j,e_{c_i}}\} \tag{5.1}$$

The images belonging to Δ are attributed the value $+1$ as the output of $h_{j,e_{c_i}}$. Consequently, their respective prediction values, initially zeros, will each be incremented by the weight of this weak classifier. On the other hand, the images that don't belong to Δ are attributed the value -1 and their prediction values are each decremented by the same weight of this classifier. After looping through all the weak hypotheses, each test image ends up having a classification value.

The process can be relatively slow depending on the database size and the number of hypotheses constituting the model. For applications that have a fixed image database (not updated online), we can compute *a priori* the distances separating each weak classifier from each image and load them when the search engine starts. This could be achieved with no difficulty because the models used are concise and so will not consume too much computer memory. Moreover, since distances are computed offline, the process can be done either exhaustively or using an index structure. Unlike retrieval with range queries, to answer a given query, images are processed one by one. For each image, the distances to the model defined are obtained through a lookup table and the score is computed by comparing these distances to the corresponding classifier radii.

5.2 Visual keyword representation

The usefulness of the method we propose is that it allows user interaction. After being constructed, a model can be visualized by users as small image patches. Each patch corresponds to a local image feature, particularly, the description region of an interest point or a set of interest points. The visual representation takes into account the window size of the descriptor (in the case of one interest point, and respectively, the region including all the points in the case of a structured set) as well as the principal orientation of the interest point (respectively the principal orientation of the center interest point of the set). However, one may omit rotating the patches to the orientation of interest

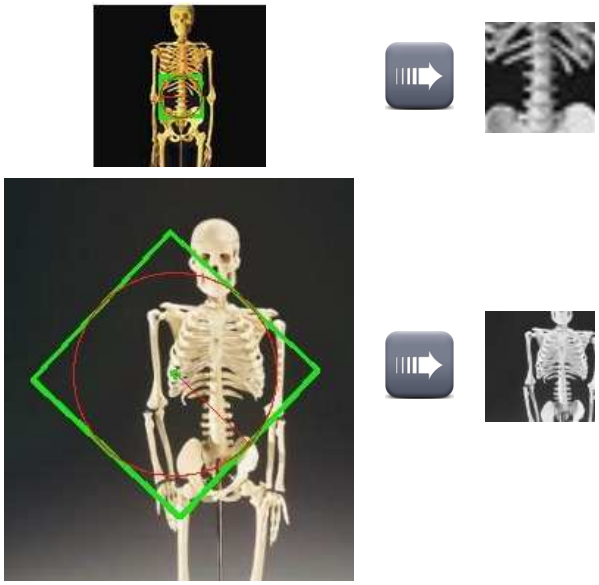


Figure 5.1: Discarding rotation and normalizing the representation of the patches.

points so as to avoid introducing another level of difficulty to users. Indeed, humans are not familiar with upside-down positions, and rotating images adds complexity and slows down the understanding of objects' components. Our primary objective is to keep the visual keywords as interpretable as possible. In addition, and for the representation size, we chose to normalize all the patches to a constant width. Two reasons can justify this choice. First, it is better for users when a software interface presents choices (i.e. object parts) with a certain regularity—from an ergonomic point of view. Second, normalizing sizes has the same effect of a zoom-in or a zoom-out, therefore it preserves the knowledge of the scale of detection. This idea is illustrated in figure 5.1.

5.3 Comprehensibility of a full visual model

Object parts can be grouped together whenever they participate to form an understandable concept which can be easily described by a word or a phrase (e.g. a face includes two eyes, a nose and a mouth). From this perspective, an object model can have various interpretations. In other words, the visual

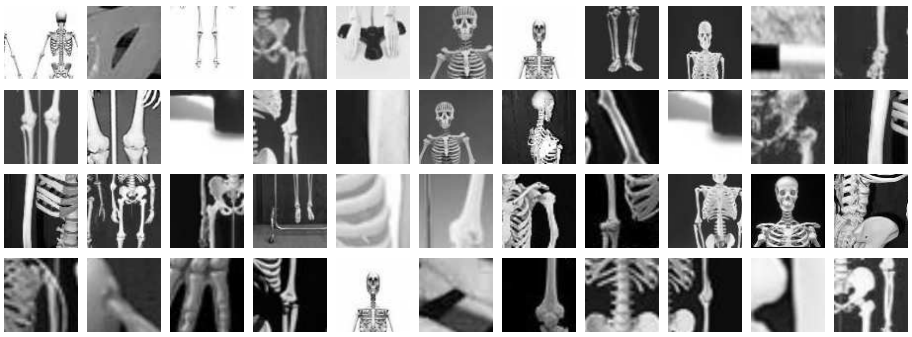


Figure 5.2: Example of an object model: 112.human-skeleton category (`Caltech` dataset). The visual keywords are displayed in decreasing order according to the predictive weights.

components of the model are fixed but different persons may perceive them differently. It is a subjective matter and it depends on the center of interest of the human operator. Figure 5.2 gives an example of a human-skeleton model. The visual keywords are displayed in decreasing order according to the predictive weights. As we can see, the model can be viewed differently (cf. figure 5.3) according to the object parts chosen. This is what interpretability is. The visual representation by keywords, suggested here, is beneficial for both researchers and end-users. For researchers, they can study what the model is composed of. Sometimes, scientists cannot explain their expertise. Take for example a biologist who is looking for the visual relations and similarities that characterize a given species of vegetation or animal. Based on a robust descriptor for the task and assuming that the opposite category in the training set is well defined, there is a high chance that this scientist discovers new interesting visual clues. For computer-science researchers, they might be interested in improving a model of an object category by making it more generic for search engines or more specific for a particular purpose. On the other hand, end-users may profit from the model representation from another perspective. They can query the system only by selecting the most representative visual keywords and with whatever proportion they choose. For instance, and assuming that users are provided with a visual model of cars but with no model of tires, a user who is looking for images containing tires can choose only the relevant patches from the car model (which includes *a priori* some visual keywords of tires).

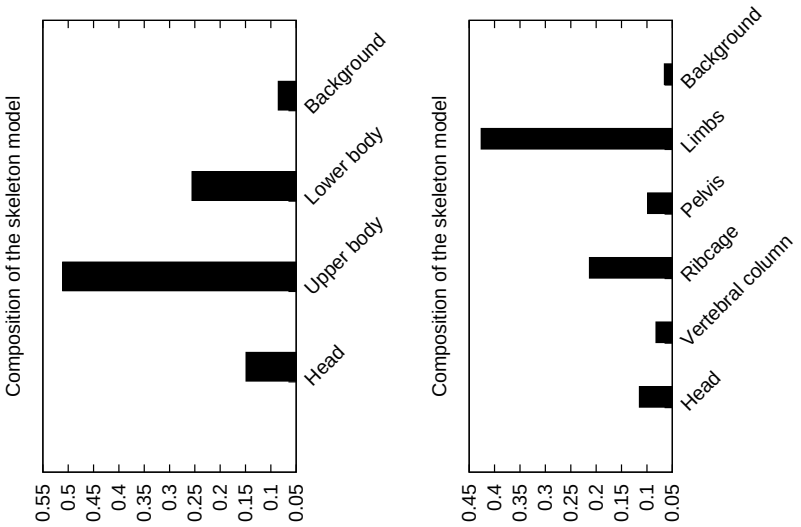


Figure 5.3: Different interpretations of the same model.

5.4 Interaction

Graphical User interfaces (GUI) are a key to interaction success. Advances in technology have made it possible to create friendly interfaces, particularly those that provide screen touch capabilities. Multimedia documents are becoming easy to display, manipulate and organize. Added to that, new options are constantly being suggested and integrated as other multimedia fields release new functionalities.

Figure 5.4 is a snapshot of our user interface. It gives an idea of what our interactive retrieval looks like. It is worth mentioning that, when designing the interface, the primary objective was not to encourage user-friendliness so much as to provide a functional prototype that illustrates the interactivity principle. In this sense, the interface is very basic since it does not include fancy options like a *drag & drop* facility, for example.

5.4.1 Constructing specialized visual models

Let us explore the capacities that the interface does provide. On the left side, users can choose a category from a list of predefined visual categories that have been trained with LARK—and that are available just below the

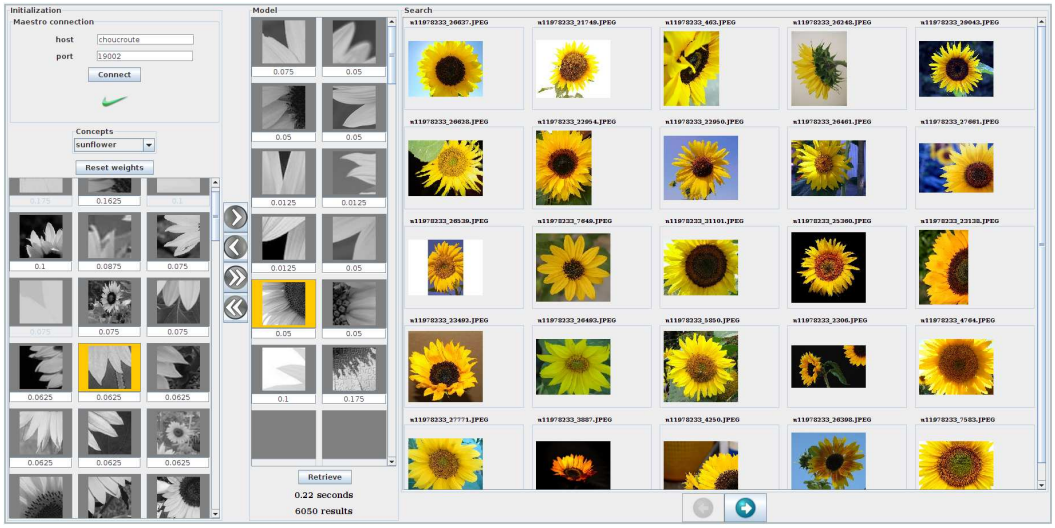


Figure 5.4: Example of an interactive search

word "Concepts". The corresponding model is then loaded. Consequently, users can browse the visual keywords displayed and start forming their own concept. Note that visual keywords are displayed with their relative weights learned during training. However, when building their models, users have the possibility to change these confidence measures according to their own perception. That is, the more relevant a feature is, the higher the weight it will be assigned. Each visual patch chosen as well as its weight are displayed in the middle panel which represents the specialized model. Between these panels we find four keys controlling patch selection. The right arrow allows users to move one or many patches (Multiple selection is possible when holding the *Ctrl* key.) to the middle panel. Inversely, the left arrow moves the selected visual keywords back from the specialized model to their original positions. The double right arrow and the double left arrows are respectively used for adding the whole model to the middle panel and for clearing it. After selecting some visual keywords, the query is ready and the search engine can be launched. The retrieved images are ranked according to the confidence scores given by eq. 5.1 where the weights w_j are either initially trained or manually specialized. These images are displayed in the right panel together with some extra information about retrieval time and the total number of images returned.

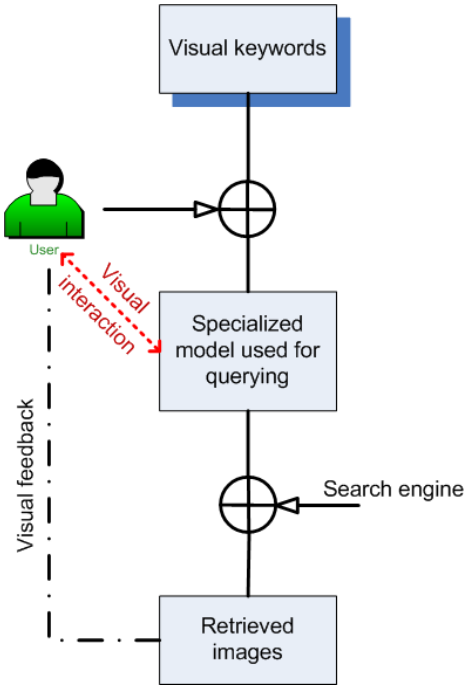


Figure 5.5: Diagram of user interaction.

5.4.2 Refining the visual model

In this section, we give visual illustrations on how user interaction is beneficial. User interaction is mostly based on target focus, context specialization and ambiguous keyword discard. A diagram showing user interaction is presented in figure 5.5.

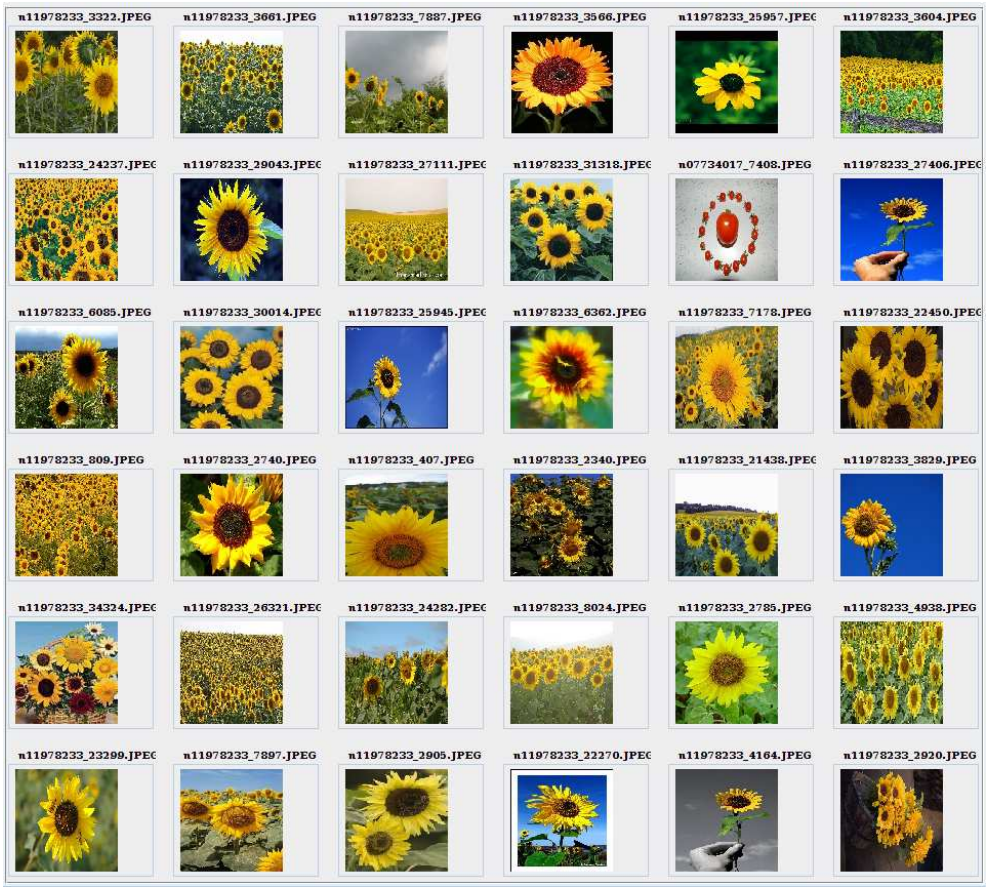
The first example is given in figure 5.6. The model used for retrieval (shown at the top) comprises six patches. The choice to make for displaying patches in a way that truly takes into account the description involved is still challenging. Here, we display the patches in a gray-scale just to point out that the description is not color-related. They were taken from the *sunflower* model and they represent a global view of sunflowers within their context. Next, in fig. (5.6-2), we show the first page search results. As we can see, the retrieved images match the query. They mostly contain sunflowers within a context (field, vegetation, sky). This result is due to the scale information brought by the patches. On the other hand, wa can notice the presence of an image containing tomatoes. Even though it is not appreciated, such a result

explains some limitations of the description used. The image does indeed appear to have the form of a sunflower. Now, let's look at the example shown in figure 5.4 and notice the difference between the first ranked results. The model used is displayed in the middle panel. Unlike the previous example, the images retrieved here tend to occupy all the image area.

As a second example, we present two different queries and their respective results (cf. fig. 5.7 and 5.8). Both queries are related to the zebra category. The first query focuses on the zebra upper-front part (head) while the second query is rather a general and global view of a zebra.

Model specialization may also turn a positive visual keyword into a negative one by attributing a negative weight. This could be helpful whenever the user wants to avoid a particular context, although it may decrease overall retrieval performance.

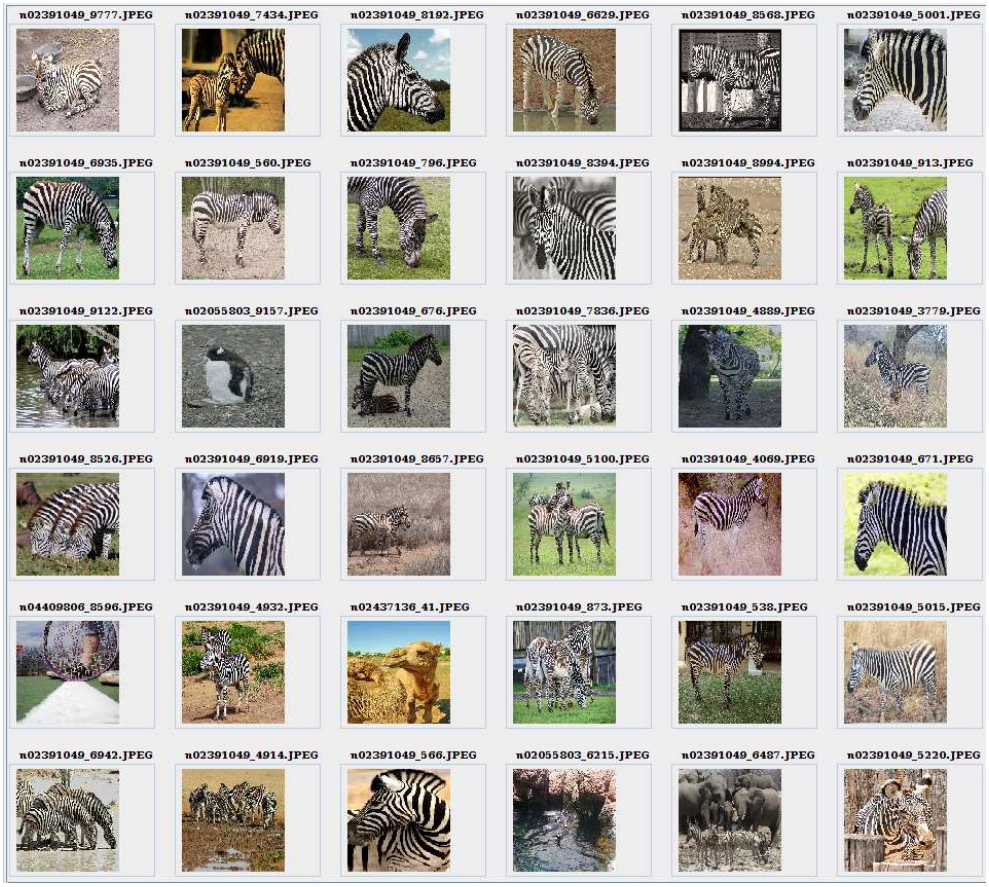
(5.6-1) A specialized sunflower model used for retrieval.



(5.6-2) Search results.

Figure 5.6: Retrieving sunflowers

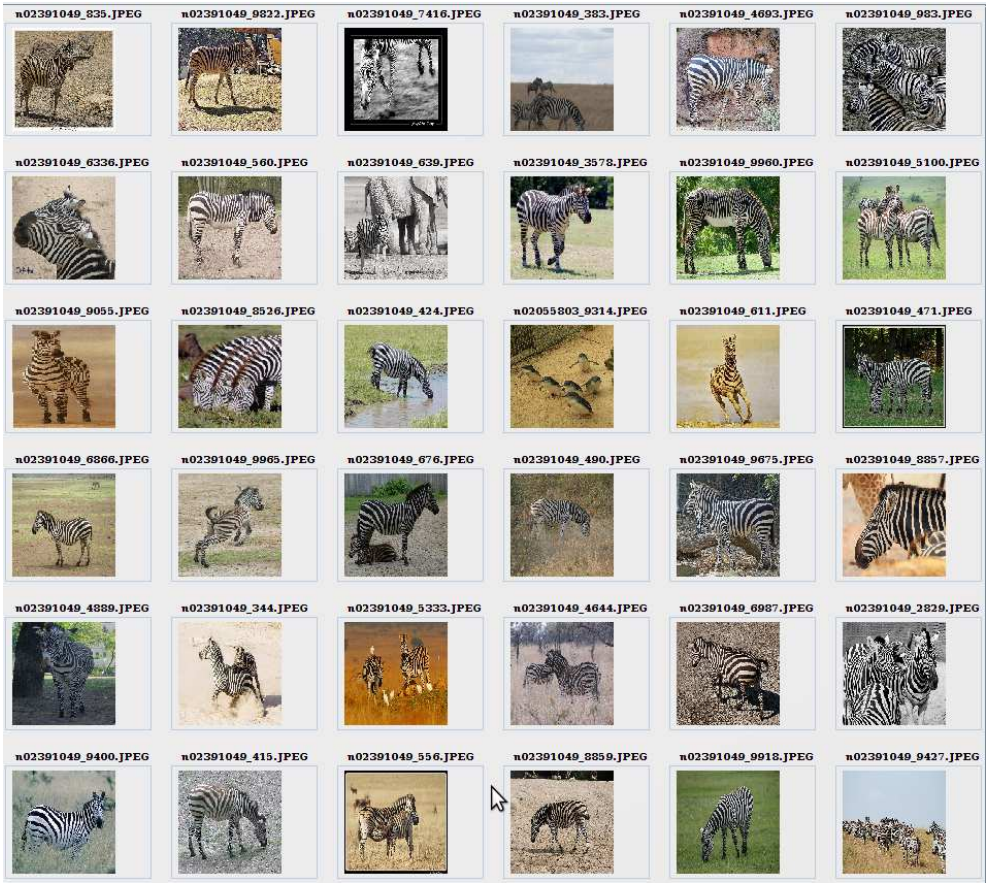
(5.7-1) Zebra specialized model used for retrieval.



(5.7-2) Search results.

Figure 5.7: Example of patches to focus on the upper-front part.

(5.8-1) Zebra specialized model used for retrieval.



(5.8-2) Search results.

Figure 5.8: Example of patches to look for a global view of zebras.

Part III

Experiments

Chapter 6

Performance Evaluation

The great tragedy of science - the slaying of a beautiful hypothesis by an ugly fact.
T.H. Huxley

ESTABLISHING an objective evaluation is not easy since the algorithm's behavior must be analyzed under various conditions. Among the many aspects that must be considered are the types of objects, the types of descriptions, the number of categories involved, search efficiency on a large scale and comparison with state-of-the-art methods. To satisfy these requirements, we carried out our experiments on different datasets and local descriptors. The two questions that have to be answered are:

1. Is the method effective compared to state-of-the-art methods?
2. Are the visual keywords produced interpretable?

In this chapter, we evaluate the retrieval performance. We will answer the first question by comparing our method to Opelt's method [Opelt 2006], which uses the AdaBoost algorithm for feature selection and it outputs readable features. Therefore, it seems suitable for comparison since we also use a boosting-like procedure. Moreover, and apart from retrieval performance, the features selected by Opelt's method can be directly likened to our visual keywords both from model size and interpretability perspectives. The evaluation of user interactivity and visual keywords interpretability will be dealt with in the next chapter.

We begin this chapter by presenting the datasets we used as well as the local features, the algorithm parameters and the evaluation metrics. Then, in section 6.2, we compare our feature selection method with Opelt's method according to the average precision measure and the number of features in the model. After that, section 6.3 shows the extent to which our models can

be generalized. Section 6.4 studies the benefits of combining multiple visual features. In section 6.5, we explore the effect of adding a counter-class model to the object class model. After that, we present the benefits of using the geometric consistency approach in section 6.6. Finally, we stress the merits of using an effective index structure in section 6.7.

6.1 Data preparation and test conditions

6.1.1 Datasets

In order to evaluate our method under different conditions, we carried out our experiments on eight different datasets, some of which correspond to benchmarks and/or are publicly available and some of which were manually constructed. They are `BelgaLogos` [1], `Caltech-256` [2], `ImageNet` [3], `ImagEval` [4], `OxfordBuilding` [5], `PascalVOC` [6], `PlantLeaves_V1` and `PlantLeaves_V2`. Some images belonging to these datasets are shown in figures 6.1 and 6.2.

`BelgaLogos`, `Caltech` and `OxfordBuilding` are publicly available. However, they are provided with no ground truth specifying a clear decomposition between training and test data. Therefore, we managed to randomly split the data with approximately the same number of images used in training as for prediction. We should mention that `OxfordBuilding` contains some *junk* images explicitly identified in the ground truth files it comes with. We discarded all these images. There are also some images that are not annotated. These images contain relevant buildings from the eleven landmarks defined, which is why we chose to exclude them from the composition of the training/test sets.

`ImageNet` is a very large database (containing 12,184,107 images—at the time of writing this thesis—and 17,624 different synsets) indexed with a hierarchy of concepts. Our research does not focus on large-scale concept numbers. From this original dataset, we considered only ten visual categories that in-

1. http://www-rocq.inria.fr/imedia/belga-logo.html
2. http://www.vision.caltech.edu/Image_Datasets/Caltech256/
3. http://www.image-net.org/
4. http://www.imageval.org/
5. http://www.robots.ox.ac.uk/∼vgg/data/oxbuildings/
6. http://pascallin.ecs.soton.ac.uk/challenges/VOC/voc2010/

tersect with some Caltech categories (i.e the same visual concepts). Since ImageNet comprises a large number of images, each training set was constructed from almost 700 images related to the concept. In our experiments, we refer to this dataset as ImageNetSmall.

ImagEval is a French initiative held in 2006 and it included participants from research teams as well as private companies. We used the images provided by this initiative, and which were dedicated to the task of object detection (task 4). The composition of the training/test sets is explicitly defined. Similarly, we used the training/validation images provided by the benchmark PascalVOC 2010 and we excluded the difficult images that are not taken into account in the final evaluation process (i.e. images labeled by 0 in the ground truth).

PlantLeaves_V1 and PlantLeaves_V2 are two manually constructed datasets built from an initial dataset named PlantLeaves which up to now has not been made publicly available. This dataset contains images that are natural scans of some plant leaves and it is constantly increasing thanks to the botanists' social network *Telabotanica*. It is being developed within the Pl@ntNet[7] project. Figure 6.2 shows some samples of this dataset.

The datasets we used differ fundamentally from various points of view, beginning with the number of visual object categories included in each database and the number of images used during training and prediction. Table 6.1 summarizes these differences. Note that the number of training images displayed corresponds to the number of positive images. Moreover, the negative image samples (i.e. counter-class) are randomly chosen from the opposite categories. (Ground truth files are necessary to check for possible overlaps when multiple objects happen to coexist in the same image.) For each object category we trained, we kept around the same number of positive and negative images. This decision is sometimes viewed as problematical since for a small number of positive training images, the number of the negative images will not be sufficient to cover the opposite categories. But in reality, it could be a good criterion to measure the capacity—of a given object class model—to generalize and to judge its discriminating ability. Moreover, and since the learning is completely statistical, maintaining a balance between the number of positive

7. http://www.plantnet-project.org/papyrus.php?langue=en

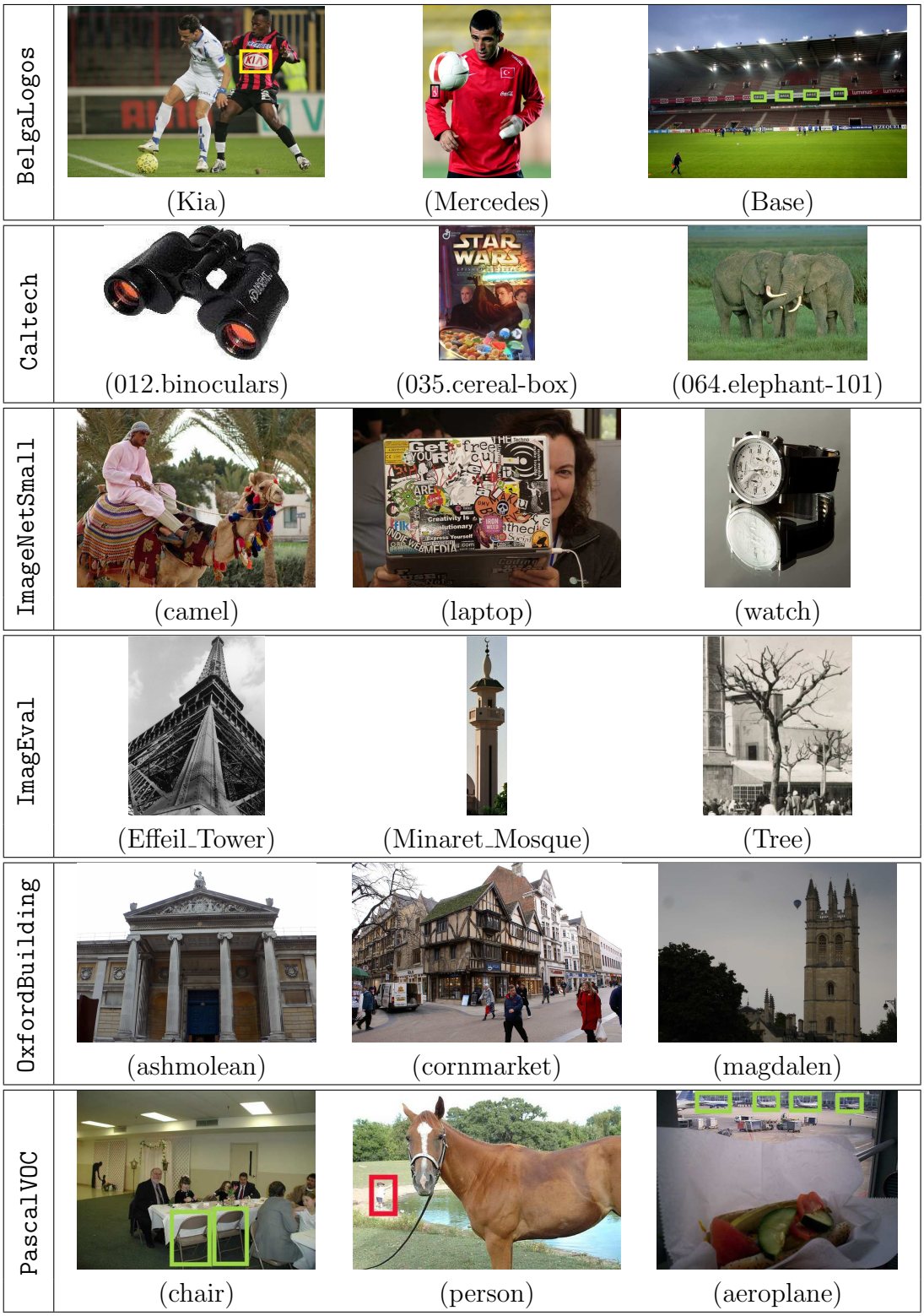


Figure 6.1: Sample images from some categories representing each dataset (Some objects are highlighted using a bounding box).

Figure 6.2: Some images forming the `PlantLeaves` dataset.

and negative images may sometimes help to avoid possible problems which could derive from imbalanced or skewed data.

The other existing differences in the datasets include the types of objects dealt with (rigid or deformable), specificity (of the same appearance as in `OxfordBuilding` which is a collection of landmarks, or generic as in `PascalVOC`), object position and scale variation (`BelgaLogos` has small objects in different image positions while `Caltech` objects have almost the same scale and occupy the center of the image) and end-user type (`PlantLeaves` is mainly intended for botanists while `ImagEval` is aimed at web users).

`Caltech-256` [Griffin 2007] was particularly used in many of our experiments. It has the advantage of comprising many object categories, and thus inter-class variability is very high. Moreover, when dealing with object-based retrieval and for any object category, the average precision measure would be very low according to a random sorting of all the images in the database (i.e. low bias with a random prediction). In this perspective, `BelgaLogos` and `OxfordBuilding` use additional images as distracters in the prediction stage. However, the number of relevant images used in prediction is the same number used in training. For the training, we also included some distracters when constructing the negative sets.

Dataset	# categories	number of images		
		training	prediction	Average per class (training)
`BelgaLogos`	23	579	9476	25.2
`Caltech`	256	15247	15360	59.6
`ImageNetSmall`	10	6830	6833	683.0
`ImagEval`	10	693	8570	69.3
`OxfordBuilding`	11	401	3115	23.6
`PascalVOC`	20	4998	5105	249.9
`PlantLeaves_V1`	12	274	283	22.8
`PlantLeaves_V2`	50	1083	1112	21.7

Table 6.1: Composition of the datasets.

6.1.2 Local features

In our experiments, we used five types of local descriptors. They are: SIFT [Lowe 2004], SURF [Bay 2008], ASIFT [Morel 2009], DIPOLE [Joly 2007] and STD_GLOBAL. Note that the source code we used (whether provided (i.e. publicly available) or developed) is in C++.

SIFT and SURF are two successful state-of-the-art descriptors used in object recognition. The implementation of SIFT is publicly available [Vedaldi 2008]. The source code of SURF was provided by the OpenSURF library [8].

ASIFT is an extension to SIFT. It was proposed to be a fully affine invariant descriptor. The implementation is available at http://www.ipol.im/pub/algo/my_affine_sift/.

DIPOLE refers to *dissociated dipoles* [Joly 2007]. It is mostly used in near duplicate search applications and is robust to many image distortions including partially affine transformation. We computed our descriptors on color Harris interest points [Gouet 1998] using the source code developed by the IMEDIA team (INRIA).

STD_GLOBAL is the concatenation of three global descriptors used here locally: EOH (i.e. a classical edge orientation histogram which is a histogram computed from gradient directions quantized into 8 bins and whose points are located at Canny edge detector), Fourier and Hough based descriptors [Ferecatu 2005] were computed in a fixed window size 65×65 centered with respect to interest points extracted using the Harris detector. These descriptors were also developed by IMEDIA.

Details about the maximum number of descriptors per image used in each dataset and the dimension of the descriptors are given respectively in tables 6.2 and 6.3.

6.1.3 Algorithm parameters

The parameters of our method were tuned as follows

$$\xi = 10^{-6} \quad \text{and} \quad \varepsilon = \frac{1}{80}$$

8. http://www.chrisevansdev.com/computer-vision-opensurf.html

Dataset	Descriptors				
	SIFT	SURF	ASIFT	DIPOLE	STD_GLOBAL
BelgaLogos	1500	-	-	-	-
Caltech	700	700	-	-	-
ImageNetSmall	500	-	-	-	-
ImagEval	1500	-	-	-	-
OxfordBuilding	4000	4000	4000	-	-
PascalVOC	2000	-	-	-	-
PlantLeaves_V1	1000	1000	1000	1000	-
PlantLeaves_V2	-	1500	1500	1500	1500

Table 6.2: The maximum number of descriptors per image used in each dataset.

	Descriptors				
	SIFT	SURF	ASIFT	DIPOLE	STD_GLOBAL
Dimension	128	64	128	20	44

Table 6.3: Descriptor dimensions.

They were empirically determined after several runs on some object categories.

Recall that ε is the forward step attributed to each selected variable. This parameter directly affects the computational complexity: $O(1/\varepsilon)$. That is, the smaller ε is, the larger the number of iterations needed for convergence. Our observations revealed that choosing a small ε didn't necessarily lead to better prediction. However, in theory, choosing a large value for ε would not lead exactly to lasso solutions.

ξ is a tolerance parameter, strictly positive, used for algorithm stability (cf. Eq. 4.23). It is decisive for determining backward steps and should be set to a small value.

For Opelt's method, we fixed the number of iterations of AdaBoost to $T = 500$.

6.1.4 Evaluation metrics

To evaluate the retrieval performance, we used the common measures used in information retrieval: *precision*, *recall* and *average precision* (AP). They

are given by:

$$Precision = \frac{(\#\text{ relevant images}) \cap (\#\text{ retrieved images})}{(\#\text{ retrieved images})} \tag{6.1}$$

$$Recall = \frac{(\#\text{ relevant images}) \cap (\#\text{ retrieved images})}{(\#\text{ relevant images})} \tag{6.2}$$

$$\text{AP} = \frac{\sum_{n=1}^{N} P(n) \cdot \text{rel}(n)}{\#\text{ relevant images}} \quad ; \quad \text{rel}(n) = \begin{cases} 1 & \text{if relevant} \\ 0 & \text{otherwise} \end{cases} \tag{6.3}$$

we also used the number of the visual keywords in the model to measure the reliability of feature selection. On the one hand, this number reflects the complexity of the model and consequently the prediction time. On the other hand, the lower this number is, the easier the interpretability of visual keywords. We discuss interpretability in the next chapter where we will also use the P@n measure to evaluate user interactivity. It is the precision computed at the cut-off rank n (i.e. after n retrieved images). This measure is also used in the formulation of AP. In equation 6.3, it is denoted $P(n)$.

6.2 Comparison with [Opelt 2006]

In order to evaluate the L_1-regularization and to test whether it is of interest in generating sparser models than standard boosting procedures, we compared our method LARK to [Opelt 2006] (hereafter referred to it as Opelt06, and which uses our own C++-based implementation). We carried out various experiments in order to compare performance, efficiency and some of the strong and weak points of each algorithm. We particularly stress the effectiveness vs. the number of features composing the model.

6.2.1 Average precision and prediction time

Table 6.4 gives the results obtained for all `Caltech` object categories but explicitly details twenty-six categories. It also shows the average performance measured over these categories as well as the average over all database object categories. On the right side of the table, we see the number of hypotheses used in each object model for both algorithms. Besides LARK and Opelt06,

we compared the results to a random selector which builds a model from five hundred hypotheses for each category. The weights of these hypotheses are all equal to 1/500 and their respective radii are defined by Eq. (4.10). Note that the performance shown for the random selector is an average over three different runs.

LARK and Opelt06 results are divided into four groups. The first and the second groups are composed of object categories where LARK outperforms Opelt06 in AP, whereas the third and the fourth groups contain object categories where Opelt06 is better. The common point between the first and the third groups is that the number of features selected by LARK is fewer than that selected by Opelt06—which is not the case for the second and the fourth groups. On average, LARK outperforms Opelt06 in AP and selects fewer features to build the model.

The behavior we seek from LARK is illustrated by the first group: higher precision with fewer features. The second and the third groups are rather ordinary because we usually expect a higher performance when the model involves a large number of features. The two first groups also reflect the stop condition problem of AdaBoost. In fact, AdaBoost sometimes stops prematurely because one hypothesis fits the data well (i.e. the classification error is zero). On the other hand, it is sometimes forced to stop after reaching the limit of five hundred hypotheses. The prediction time results of the third group are presented in table 6.5. They highlight our high gain in time compared to the loss in precision shown in table 6.4. Note that the prediction method used here is exhaustive. This choice doesn't favor real-time interactive search applications but it was made here in order to compare the real performances of both algorithms without introducing any bias from index structures and range queries. It is worth pointing out that the prediction time is linear with respect to the model size. Finally, the fourth group (i.e. 043.coin and 177.saturn) behaves worse than Opelt06—with more features selected and a lower AP. This case is very rare (only 15 categories from the 256 categories) which explains the fact that, on average, LARK performs better both in average precision and prediction time. Yet the precision-recall curve, given by figure 6.3, shows that Opelt06 has a slightly better precision than LARK for the first images returned, then LARK becomes better.

	Category	Average precision			Number of features	
		Random	LARK	Opelt06	LARK	Opelt06
Group 1	005.baseball-glove	0.0833	0.2263	0.1692	37	40
	008.bathtub	0.0313	0.0438	0.0373	23	500
	077.french-horn	0.1407	0.1628	0.1037	20	31
	137.mars	0.2509	0.3270	0.2520	12	108
	145.motorbikes-101	0.5296	0.8068	0.7749	76	93
	194.socks	0.0116	0.0160	0.0088	20	134
	234.tweezer	0.0675	0.0808	0.0732	13	77
	235.umbrella-101	0.0225	0.0530	0.0335	25	123
	238.video-projector	0.0054	0.0159	0.0071	4	106
	255.tennis-shoes	0.0083	0.0123	0.0065	20	92
	Average over 93 categories	0.0218	**0.0400**	0.0308	**18.6**	98.5
Group 2	025.cactus	0.0499	0.0247	0.0187	21	18
	037.chess-board	0.4520	0.7293	0.4832	37	7
	053.desk-globe	0.0316	0.0882	0.0611	27	15
	130.license-plate	0.0974	0.4111	0.0940	27	7
	200.stained-glass	0.0164	0.1025	0.0980	39	13
	218.tennis-racket	0.1269	0.2190	0.0758	25	18
	225.tower-pisa	0.0396	0.2561	0.1968	36	21
	237.vcr	0.0476	0.0434	0.0374	29	24
	250.zebra	0.0438	0.2868	0.1912	36	5
	252.car-side-101	0.0159	0.2751	0.2301	33	13
	Average over 35 categories	0.0558	**0.1356**	0.0922	30.3	**15.2**
Group 3	003.backpack	0.0508	0.1294	0.1341	18	33
	011.billiards	0.0131	0.0646	0.0968	40	284
	201.starfish-101	0.0138	0.0140	0.0248	4	23
	232.t-shirt	0.0767	0.1490	0.1523	98	356
	Average over 113 categories	0.0199	0.0317	**0.0499**	**16.5**	102.2
Group 4	043.coin	0.0080	0.0347	0.0486	38	12
	177.saturn	0.5623	0.4741	0.5252	39	25
	Average over 15 categories	0.0836	0.0808	**0.0989**	27.7	**17.9**
All	Average over the 256 categories	0.0292	**0.0518**	0.0516	**19.8**	84

Table 6.4: Comparing LARK with Opelt06 for `Caltech` dataset. A random selector is given as a reference (500 hypotheses per category).

Category	Prediction time (s)	
	LARK	Opelt06
003.backpack	1	1.6
011.billiards	1.3	6.3
201.starfish-101	0.4	1.6
232.t-shirt	6.5	21.8
Average	2.3	7.8
Average over the 256 categories	1.3	5

Table 6.5: Comparing LARK with Opelt06 according to the prediction time (`Caltech` dataset).

After presenting the results of the `Caltech` dataset in detail, we provide a summary of the results of all the datasets in table 6.6. This table shows that LARK AP is better than Opelt06 AP in five datasets. Moreover, in the three other datasets (i.e. where Opelt06 is better), we see that the average number of features in the Opelt06 model is always 500. This means that there were no categories that overfitted the training data. Despite this relatively high number of features, Opelt06 was only better by 1.6%, 1.3% and 1.5% respectively in `ImageNetSmall`, `ImagEval` and `PascalVOC`. Moreover, in these three datasets, the prediction time needed by Opelt06 is ten times the prediction time needed by LARK (on average). Looking closely at the precision-recall curves given in figure 6.4, we can see that apart from the datasets `OxfordBuilding` and `PlantLeaves` where Opelt06 performed very poorly, the retrieval performance of LARK and Opelt06 are almost equivalent with a slight superiority of Opelt06 for 40% recall noticeable in `BelgaLogos` and `ImageNetSmall`. Nonetheless, we judge that feature selection performed by LARK is always better with fewer visual keywords. It was only in `PlantLeaves_V2` that the number of features selected by LARK was greater than the number of features selected by Opelt06. Again, and from a retrieval perspective, it is clear in this case that the Opelt06 model is not optimal with a 40.2% AP less than LARK. Notice also that, on average, LARK AP exceeded Opelt06 AP by 11.1% and the prediction time needed by LARK models is one third the time needed by Opelt06 models.

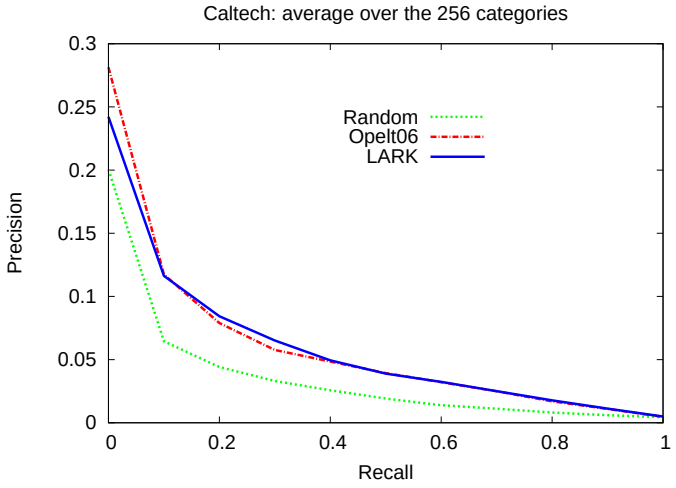


Figure 6.3: Precision-Recall curve: `Caltech` dataset (Average over the 256 object categories).

Dataset	AP		# features		Prediction time (s)	
	LARK	Opelt06	LARK	Opelt06	LARK	Opelt06
`BelgaLogos`	**0.2008**	0.1772	78.4	326.4	317	1,244
`Caltech`	**0.0518**	0.0516	19.8	84	339	1,290
`ImageNetSmall`	0.3979	**0.4142**	60.3	500	44	154
`ImagEval`	0.2177	**0.2306**	24.8	500	54	986
`OxfordBuilding`	**0.6509**	0.3275	211.1	227.8	277	328
`PascalVOC`	0.1285	**0.1437**	61.5	500	46	378
`PlantLeaves_V1`	**0.8689**	0.6513	191.1	250.5	8	11
`PlantLeaves_V2`	**0.6464**	0.2744	466.1	110.8	498	129
Average	**0.3954**	0.2843	139.1	312.4	197.9	565.0

Table 6.6: LARK vs. Opelt06: results summary.

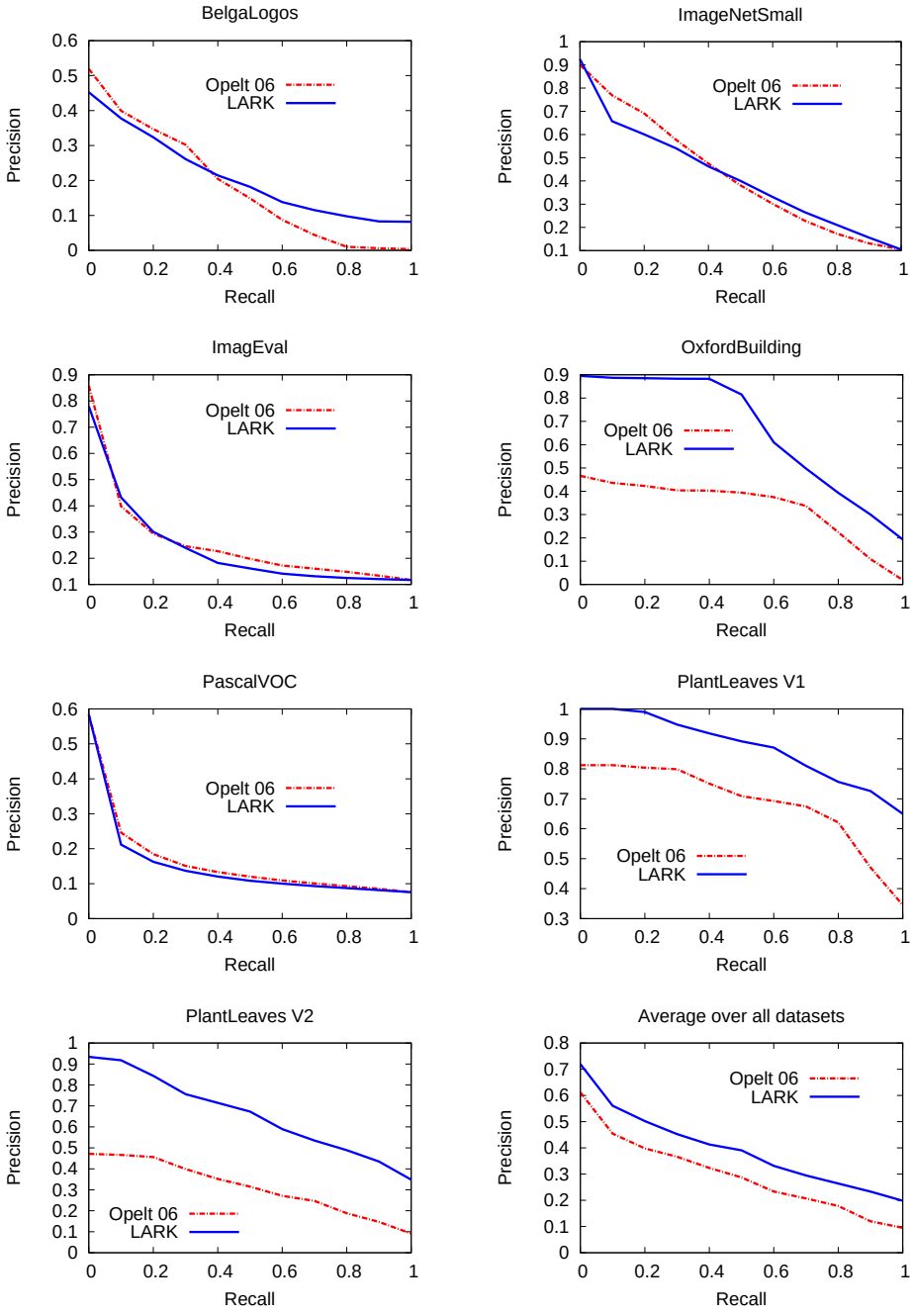


Figure 6.4: Precision-Recall curves.

6.2.2 Overtraining: the effect of increasing the model size

This experiment is intended to measure the increase in performance that will *normally* occur when selecting more visual features in the model. It was carried out on four of our prepared datasets: `Caltech`, `ImageNetSmall`, `ImagEval` and `PascalVOC`. We made several prediction runs each with a limited number of features in the class model. At a given run k, we only kept the *first* k-hypotheses that had the *strongest* weights. We varied k from 1 to 500 and we registered the AP results as well as the number of features m_{ki} used by the category i ($m_{ki} \leq k$; $1 \leq i \leq N_c$ where N_c is the total number of the categories present in the dataset). Note that m_{ki} is equal to k when the model comprises at least k hypotheses and it is strictly inferior otherwise. After that, and for each k, we computed the average number of hypotheses used:

$$q_k = \frac{1}{N_c} \sum_{i=1}^{N_c} m_{ki}$$

This experiment was carried out with both LARK and Opelt06. The results are shown in figure 6.5 for comparison, and the different curves give a global overview of how each algorithm behaves. As expected, the more visual keywords we select, the higher the average precision. In general, AdaBoost continues to select more features even if the gain in AP is very small or insignificant. This is known as *overtraining*. In this situation, however, BLasso stops. This highlights the fact that BLasso judges the overall model consistency after each single step: any hypothesis to be added must *significantly* contribute to increasing the performance.

This experiment, while not precisely giving the optimization path driven by BLasso, demonstrates the smooth behavior of our method. For `Caltech` and `ImagEval`, we notice that Opelt06 is slightly better than LARK in the early stage, and after that, LARK is better. On the other hand, for `ImageNetSmall` and `PascalVOC`, feature selection by LARK is clearly better from the very first visual keywords selected. Moreover, for `Caltech` and `PascalVOC`, LARK stops selecting features just a few steps before Opelt06 performance becomes nearly constant. In `ImagEval`, we remark that the curves of LARK and Opelt06 are very close until LARK stops. At this phase, we notice a sudden change in the

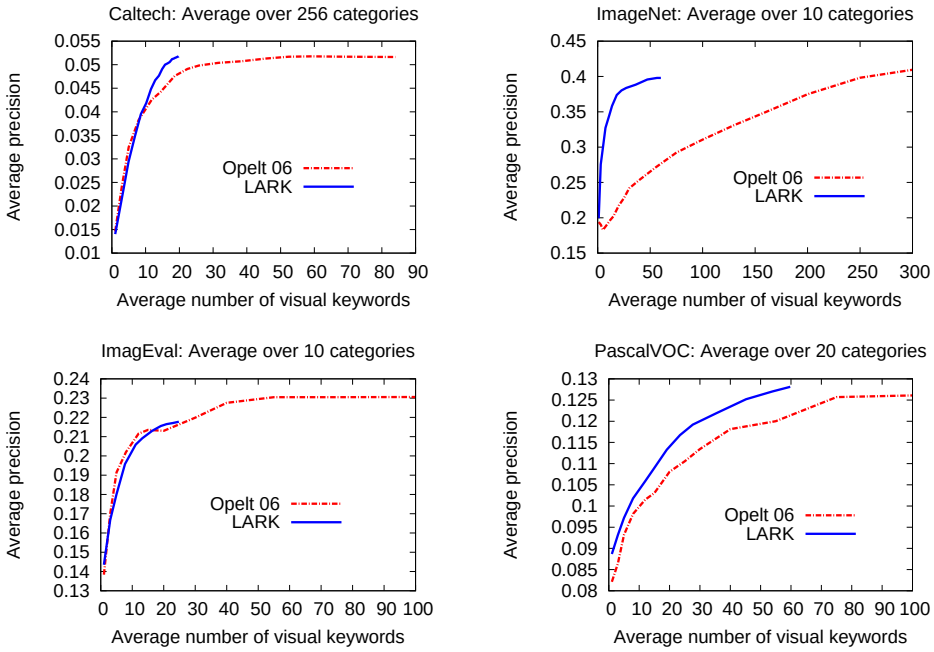


Figure 6.5: Increasing the model size.

Opelt06 curve: it moves from a nearly constant performance (along a segment of eight features) to an increase with a small slope, to become constant again. In `ImageNetSmall`, we see that for a given number of features (in the model), LARK has a far higher performance than Opelt06. The gap takes its highest value 15.6% at the index 14. This observation shows another problematical side of AdaBoost's stopping condition. If we chose $T < 200$, Opelt06 would *underfit* the training data. In other words, the model would not be complete (i.e. small in size and not discriminant enough) and hence would not perform well on test data.

Overall, we see that the LARK selection is always satisfactory, achieving an adequate tradeoff between underfitting and overtraining.

6.2.3 Overfitting phenomenon

Looking back at the precision-recall curves presented in figure 6.4, the superiority in performance of LARK over Opelt06 in `OxfordBuilding` and `PlantLeaves` datasets is intriguing, particularly because their performance is

quite similar in the other datasets. In this section, we will see that this result is due to the overfitting of AdaBoost. Although it was previously observed in `Caltech`, overfitting here is more noticeable.

Table 6.7 is a comparison between LARK and Opelt06 on the `OxfordBuilding` dataset. It shows that for six object categories, Opelt06 stops after selecting only one visual feature. This phenomenon happens every time AdaBoost selects a hypothesis whose classification error is exactly zero. Therefore, it memorizes the training data perfectly rather than learning to generalize. We notice that, in general, when AdaBoost doesn't overfit (500 hypotheses), the number of visual keywords selected by BLasso is relatively small (apart from the "radcliffe_camera" category). This explains the 5.7% increase in performance of Opelt06 (cf. table 6.7). On the other hand, looking closely at the precision-recall curve (cf. figure 6.6-a), we see that both algorithms behave nearly identically for 40% recall. This observation is also true for the `PlantLeaves` dataset. We will limit ourselves here to presenting only the results of `PlantLeaves_V1` (cf. table 6.8) since it has fewer categories than `PlantLeaves_V2`, and hence the results are more readable. In this dataset, and for the categories where no overfitting was observed, Opelt06 outperforms LARK by 4.3%. This increase does not greatly affect the first images to be retrieved. In fact, the precision-recall curve (cf. figure 6.6-b) states that retrieval is the same for 20% recall. We also notice that, as in `OxfordBuilding`, when AdaBoost overfits, BLasso chooses more visual features than it "usually" does. This observation also coincides with using relatively few training images, and so we believe that when AdaBoost *easily* overfits, BLasso tries to generalize by selecting more visual features.

6.3 Generalization capabilities

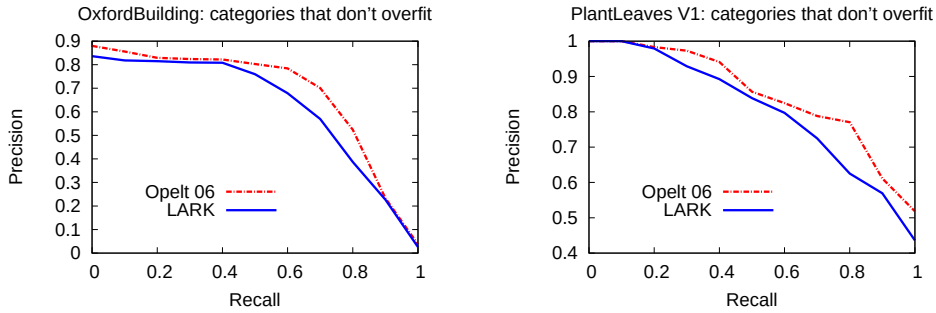
This experiment aims to measure the extent to which the visual keywords generated by LARK can be generic. As we mentioned earlier, the selection of `ImageNetSmall` object categories was intended to be a subset of `Caltech` visual categories. Therefore, we can predict the same concepts belonging to `Caltech` using the visual keywords generated from `ImageNetSmall`.

Category	# images (training)	AP		# visual keywords	
		LARK	Opelt06	LARK	Opelt06
all_souls	66	0.8983	**0.9141**	33	500
christ_church	271	0.7197	**0.8176**	24	500
hertford	33	0.6474	**0.7454**	27	500
magdalen	342	0.0484	**0.1111**	25	500
radcliffe_camera	141	0.8106	**0.8226**	328	500
Average	170.6	0.6249	**0.6822**	87.4	500
ashmolean	97	**0.7138**	0.0127	164	1
balliol	77	**0.2769**	0.0061	224	1
bodleian	107	**0.9952**	0.0896	949	1
cornmarket	29	**0.5117**	0.0047	291	1
keble	60	**0.5376**	0.0423	105	1
pitt_rivers	54	**1.0000**	0.0359	152	1
Average	70.7	**0.6725**	0.0319	314.17	1
Average	116.1	**0.6509**	0.3275	211.1	227.8

Table 6.7: Comparing LARK with Opelt06 (`OxfordBuilding`).

Category	# images (training)	AP		# visual keywords	
		LARK	Opelt06	LARK	Opelt06
Acer monspessulanum L	24	**0.9983**	**0.9983**	32	500
Arbutus unedo L	34	0.7078	**0.9158**	47	500
Paliurus spina-christi Mill	24	0.6466	**0.6976**	20	500
Cercis siliquastrum L	29	0.8091	**0.8510**	27	500
Nerium oleander	41	**0.9975**	0.9962	41	500
Quercus ilex L	36	**0.6584**	0.6211	43	500
Average	31.3	0.8030	**0.8467**	35	500
Corylus avellana L	21	**1.0000**	0.3541	448	1
Eriobotrya japonica	7	**1.0000**	0.0253	456	1
Quercus pubescens	16	**0.9589**	0.4490	50	1
Sorbus domestica	7	**1.0000**	0.8942	933	1
Crataegus monogyna Jacq	13	**0.6543**	0.1548	20	1
Robinia pseudoacacia L	22	**0.9963**	0.8580	176	1
Average	14.3	**0.9349**	0.4559	347.2	1
Average	22.8	**0.8689**	0.6513	191.1	250.5

Table 6.8: Comparing LARK with Opelt06 (`PlantLeaves_V1`).



(a) `OxfordBuilding` (b) `PlantLeaves_V1`

Figure 6.6: Precision-recall curves for the categories where no overfitting is observed with Opelt06.

Results are presented in table 6.9. We notice that the visual keywords learned from the `ImageNetSmall` training set outperformed the `Caltech` models in six categories and that they also did better, on average. This proves that the models generated are generic in the sense that they don't degrade and even improve retrieval results. Therefore, they may be applied to various test databases.

Category	Average precision	
	`Caltech` model	`ImageNetSmall` model
028.camel	0.0056	**0.0132**
127.laptop-101	0.0388	0.0382
158.penguin	0.0100	**0.0159**
172.revolver-101	0.0128	0.0096
189.snail	0.0067	**0.0178**
204.sunflower-101	0.5670	**0.7590**
218.tennis-racket	0.0819	0.0170
221.tomato	0.0156	**0.1018**
240.watch-101	0.0458	**0.0792**
250.zebra	0.0933	0.0787
Average	0.0878	**0.1130**

Table 6.9: Illustration of prediction on a different dataset. The third column presents AP results obtained by predicting on the `Caltech` test set with the visual keywords trained with `ImageNetSmall` training set.

Our second experiment is an attempt to build a hierarchical search engine that can semantically provide models for top level concepts and their sub-concepts. It was carried out using the `ImageNetSmall` dataset. The idea

is to draw together the `camel`, `penguin`, `snail` and `zebra` categories under the concept *animal*, then the `laptop`, `revolver`, `tennis_racket` and `watch` categories under the concept *man-made* and finally the `sunflower` and `tomato` categories under the concept *vegetation*. For training purposes, we used 30 images for each sub-concept. In summary, there were 120 images used in *animal* and *man-made* concepts and only 60 images used for *vegetation*. For prediction, we used a total of 13, 363 images. AP results are given in table 6.10. They are rather promising and demonstrate the feasibility of a hierarchical search engine with our method.

Category	# features	AP
animal	36	0.5516
man-made	22	0.5536
vegetation	33	0.4224

Table 6.10: Results of top semantical concepts.

6.4 Combining multiple visual features

Using various description types is a major key to improving the prediction accuracy. The experiments presented here are intended to show to what extent this claim is true. They were carried out on four datasets: `Caltech`, `OxfordBuilding`, `PlantLeaves_V1` and `PlantLeaves_V2` (cf. table 6.2). For each dataset, we computed the AP of each visual descriptor individually and we compared them with the AP obtained by the combination of all the descriptors.

The AP results of `Caltech` are shown in table 6.11. They are clustered into two groups. The first group comprises the object classes where both descriptors (SIFT and SURF) combined together achieve higher performances than each single descriptor, whereas the second group is composed of classes where the combination works less well than at least one of the descriptors. It is worth mentioning that for some object classes, SIFT performs much better than SURF whereas, for other classes, the opposite is true. Table 6.11 presents eight categories where SIFT is better and four other categories where SURF is. On average, SIFT is better than SURF and the collaboration of both is even better. One may notice that, *on average*, the total number of the

	Category	Average Precision			Number of visual keywords		
		SIFT	SURF	SIFT+SURF	SIFT	SURF	SIFT+SURF
Increase	145.motorbikes-101	0.6349	**0.7061**	**0.8068**	41	75	76
	177.saturn	**0.2890**	0.0278	**0.4741**	34	1	39
	218.tennis-racket	**0.0722**	**0.0675**	0.219	9	24	25
	232.t-shirt	**0.0960**	0.0899	**0.1490**	52	48	98
	251.airplanes-101	0.5368	**0.5751**	**0.7218**	35	45	56
	253.faces-easy-101	**0.8515**	**0.7857**	0.8641	78	61	58
	Average over 119 categories	0.0590	0.0464	**0.0816**	16.4	16.1	22.7
Decrease	146.mountain-bike	**0.1243**	0.0417	0.0511	32	29	15
	184.sheet-music	**0.1910**	0.1776	0.1746	29	55	27
	204.sunflower-101	**0.4251**	0.0781	0.2149	15	28	5
	224.touring-bike	0.0809	**0.1286**	0.0892	17	22	27
	234.tweezer	0.0406	**0.1010**	0.0808	4	82	13
	235.umbrella-101	**0.0775**	0.0166	0.053	19	11	25
	Average over 137 categories	0.0285	0.0247	0.0258	15.9	15.9	17.3
All	Average over the 256 categories	0.0426	0.0348	**0.0518**	16.1	16.0	19.8

Table 6.11: AP measured in three cases: SIFT only, SURF only and the combination of both in `Caltech` dataset.

visual keywords selected when using both descriptors is greater than when each descriptor operates individually. However, we estimate that these results are satisfactory since the increase in AP (i.e. 30%) is higher than the increase in the number of features (i.e. 26%). Besides, keeping less than twenty visual keywords is suitable from an interpretability point of view.

The conclusions drawn from the results of `Caltech` are also the same for the other datasets. That is, the overall performance (i.e. the average on all object categories) of descriptor combinations is better than using one descriptor at a time and, the number of visual keywords selected with descriptor combinations is higher than the number of visual keywords selected by any descriptor used alone. Furthermore, for some object categories, a given de-

scriptor combination may give a worse AP than one of the descriptors. Yet the reason for this behavior is unclear.

The results of `OxfordBuilding`, `PlantLeaves_V1` and `PlantLeaves_V2` are given respectively in tables 6.12, 6.13 and 6.14. For conciseness, and for each dataset, we only kept the average over all its object categories.

	Combination	ASIFT	SIFT	SURF
AP	**0.5672**	0.4819	0.3121	0.5078
# features	303.9	156.5	164.1	133.2

Table 6.12: Combining visual descriptors (`OxfordBuilding`).

	Combination	SIFT	ASIFT	SURF	DIPOLE
AP	**0.8689**	0.6999	0.5532	0.7642	0.6097
# features	191.1	101.2	116.8	123.3	88.5

Table 6.13: Combining visual descriptors (`PlantLeaves_V1`).

	Combination	ASIFT	SURF	DIPOLE	STD_GLOBAL
AP	**0.6464**	0.5810	0.5587	0.3081	0.5094
# features	466.5	258.9	230.3	137.3	346.2

Table 6.14: Combining visual descriptors (`PlantLeaves_V2`).

6.5 Adding a counter-class model

Discovering what is not the object may turn out to be a useful clue to knowing what the object is. From this perspective, apart from the object class model, we added a counter-class model. That is, we learned the negative class against the object class. The main drawback is that the number of visual keywords is nearly double. The final classifier is the resulting sum from both models: $H(x) = H_1(x) + H_2(x)$ where H_1 is the strong classifier of the object category and H_2 is the strong classifier of the opposite model.

$$H_1(x) = \sum_i a_i h_i(x) \quad \text{and} \quad H_2(x) = \sum_j b_j h'_j(x)$$

	Category	Object	Object + Counter-class
Increase	044.comet	0.049	0.3864
	067.eyeglasses	0.0417	0.2801
	129.leopards-101	0.2464	0.7867
	177.saturn	0.4741	0.8226
	182.self-propelled-lawn-mower	0.0549	0.3089
	234.tweezer	0.0808	0.3311
	Average over 181 categories	0.0525	0.0877
Decrease	037.chess-board	0.7293	0.3814
	086.golden-gate-bridge	0.1185	0.0873
	154.palm-tree	0.0736	0.0224
	230.trilobite-101	0.4152	0.3699
	253.faces-easy-101	0.8641	0.7697
	256.toad	0.0572	0.0242
	Average over 75 categories	0.0500	0.0370
All	Average over the 256 categories	0.0518	0.0728

Table 6.15: Adding a counter-class model (using LARK).

Let us take a weak classifier h' belonging to the counter-class and suppose that it is associated with the feature F' and the discriminative radius R'. This classifier will output

$$h'(x) = \begin{cases} -1 & \text{if } \mathrm{d}(F', x) \leq R' \\ +1 & \text{otherwise} \end{cases}$$

The experiment covered all the object categories of `Caltech`. For some object classes, the AP significantly increased or decreased. For others, it remained nearly constant. Table 6.15 illustrates some of the results where large changes occurred.

In fact, the changes recorded in AP may happen for various reasons. One of these is that the object doesn't always cover an entire test image. Sometimes, it constitutes only a small region. In this particular case, when adding a

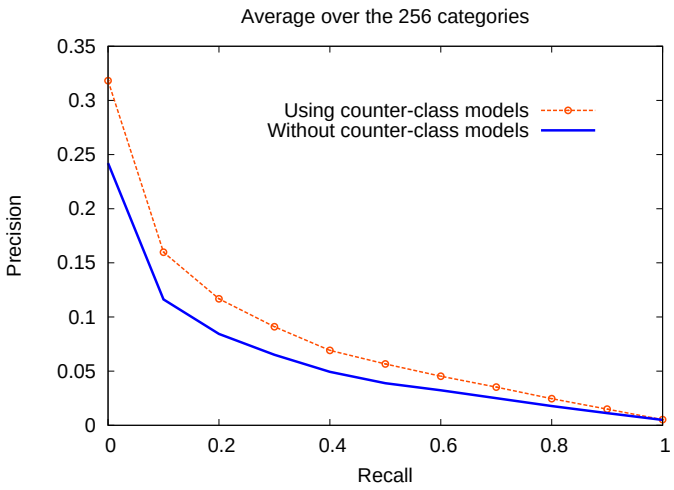


Figure 6.7: Adding a counter-class: Precision-Recall curve of `Caltech` dataset.

counter-class model for an image where there exists one or many negative objects—apart from the positive object we are looking for, weak classifiers of the counter-class model (i.e. $h'_j(.)$) will output -1 because they will find objects inside their discriminant radii. In other words, saying that there are negative objects in the image doesn't necessarily mean that there are no positive ones. This behavior, although correct, raises a serious problem by neglecting the object we are looking for. Given a test image I_t, four possible scenarios could occur:

- I_t contains at least one object:
  - if $\mathrm{d}(F', I_t) \leq R' \Rightarrow$ *bad classification*
  - if $\mathrm{d}(F', I_t) > R' \Rightarrow$ *good classification*
- I_t doesn't contain any objects
  - if $\mathrm{d}(F', I_t) \leq R' \Rightarrow$ *good classification*
  - if $\mathrm{d}(F', I_t) > R' \Rightarrow$ *bad classification*

6.6 Geometric consistency

So far, we have only evaluated our method using *simple* visual keywords which are described by only one local descriptor. We now evaluate LARK taking into account the local geometric constraints described in section 4.4.

Here, each visual keyword is represented by a structured set of local features. To make a comparison with standard boosting, we also extended the work by Opelt [Opelt 2006] to a geometrically consistent matching. We will refer to this method as GC_Opelt. The comparison results are given in table 6.16. They summarize the AP measured for LARK, Opelt06 and GC_Opelt in all the datasets.

Both `BelgaLogos` and `OxfordBuilding` datasets showed an increase in performance using the geometric approach either with LARK or GC_Opelt. This result confirms that a geometric matching strategy is useful when dealing with rigid objects, regardless of their relative size and scale in images. We can also notice that GC_Opelt AP is better than LARK AP by 2.7% in `BelgaLogos` and 1.4% in `OxfordBuilding`. Given the average number of features selected by GC_Opelt, which were respectively 478 and 500 (cf. table 6.18), we conclude that the geometric consistency approach helps AdaBoost to overcome the overfitting phenomenon. This is because weak hypotheses (which consist in structured coherent sets of interest points) are more discriminant than those hypotheses built with one feature per visual word, and which allow an error rate of zero to be reached quickly. On the other hand, the average number of visual keywords selected by LARK drops from 78.4 to 44 and from 211.1 to 32.5 respectively in `BelgaLogos` and `OxfordBuilding`. This observation concurs with the results presented in section 6.2.3 and shows again that while AdaBoost is greedy, BLasso feature selection is more compact and meaningful.

Looking at the prediction time (cf. table 6.17), we conclude that the increase in the GC_Opelt AP with respect to the LARK AP is not *objectively* advantageous since the prediction time needed by GC_Opelt models is nine times (respectively almost thirteen times) more than the time needed by LARK models to predict on `BelgaLogos` (respectively `OxfordBuilding`). On average, the prediction time by GC_Opelt models is seven times greater than the prediction time by LARK models.

Table 6.16 also shows that using the simple approach gives better performances in `Caltech`, `ImageNetSmall`, `ImagEval` and `PascalVOC`. These datasets are characterized by generic object categories with no fixed appearance. However, applying geometric consistency on `PlantLeaves_V1` and

PlantLeaves_V2 gives better results with GC_Opelt and lower results with LARK. On average, we see that the number of visual keywords selected by LARK using the geometric approach is always small.

Dataset	Simple approach		Geometric approach	
	LARK	Opelt06	LARK	GC_Opelt
BelgaLogos	0.2008	0.1772	**0.2320**	**0.2590**
Caltech	**0.0518**	**0.0516**	0.0450	0.0489
ImageNetSmall	**0.3979**	**0.4142**	0.2234	0.2008
ImagEval	**0.2177**	**0.2306**	0.1660	0.2131
OxfordBuilding	0.6509	0.3275	**0.6871**	**0.7015**
PascalVOC	**0.1285**	**0.1437**	0.0923	0.0965
PlantLeaves_V1	**0.8689**	0.6513	0.7069	**0.8080**
PlantLeaves_V2	**0.6464**	0.2744	0.6101	**0.6886**
Average	**0.3954**	0.2838	0.3454	**0.3777**

Table 6.16: Comparing the geometrically consistent approach to the simple approach: average precision.

Dataset	Simple approach		Geometric approach	
	LARK	Opelt06	LARK	GC_Opelt
BelgaLogos	317	1,244	21,048	187,990
Caltech	359	411	42,748	199,752
ImageNetSmall	45	154	2,366	5,086
ImagEval	71	986	8,992	51,132
OxfordBuilding	267	328	11,433	145,223
PascalVOC	46	378	5,244	33,882
PlantLeaves_V1	7	12	80	689
PlantLeaves_V2	498	198	3,049	32,610
Average	201	464	11,870	82,046

Table 6.17: Comparing the geometrically consistent approach to the simple approach: prediction time results (in seconds).

6.7 Using an efficient index structure

In order to reduce the processing time in training and prediction, it is useful to use an index structure that can efficiently retrieve kNN objects in a

Dataset	Simple approach		Geometric approach	
	LARK	Opelt06	LARK	GC_Opelt
BelgaLogos	78.4	326.4	44	478
Caltech	19.8	84	40.6	497.1
ImageNetSmall	60.3	500	29.2	55.9
ImagEval	24.8	500	29.1	500
OxfordBuilding	211.1	227.8	32.5	500
PascalVOC	61.5	500	59.5	475.5
PlantLeaves_V1	191.1	250.5	30.4	500
PlantLeaves_V2	466.1	110.8	39.3	500
Average	139.1	312.4	**38.1**	438.3

Table 6.18: Comparing the geometrically consistent approach to the simple strategy approach: number of visual keywords.

very short time. However, index structures are usually based on approximate range query search, and hence the retrieval performance may be reduced.

In our experiments, we relied on the state-of-the-art index structure presented in [Joly 2008]. It is a multi-probe LSH that defines a reliable *a posteriori* model taking into account some prior knowledge about the queries and the objects to be retrieved. It is based upon a nearest neighbor search. The principle of this method is to visit the most probable hash buckets of a given hash function according to their posterior probabilities. More precisely, the method selects the minimal set of hash buckets such that the global probability is higher than a quality control parameter α. This parameter controls the retrieval quality with respect to the gain in search time.

The experiments presented here only deal with our method. Besides the exhaustive mode where $\alpha = 1$, we carried out experiments with $\alpha = 0.8$. Note that the *same* value of α used in training is also used in prediction. The results obtained are shown in tables 6.19 and 6.20. As expected, we notice that the performance of the exhaustive mode is always better with the simple approach (on average, a gain of 7.8% in AP). However, in the geometric approach, there were three datasets with the approximate search (i.e $\alpha = 0.8$) that outperformed the exhaustive search. These were `BelgaLogos`, `OxfordBuilding` and `ImagEval`. From the previous section, `BelgaLogos` and `OxfordBuilding` showed an increase in the AP with the geometric approach

because they categorize rigid objects with same shape and appearance. Here, the approximate search gave even better results than the exhaustive search. For `ImagEval` and with $\alpha = 0.8$, notice that the AP with the geometric approach (cf.table 6.20) is slightly greater than the AP with the simple approach (cf.table 6.19). This was not the case with the exhaustive search where the simple approach clearly outperformed the geometric consistency approach.

In table 6.20, we can also notice that the average performance over all the datasets is almost equivalent between setting α to 1 and setting α to 0.8. Therefore, we conclude that approximate search is much more effective when using the geometric approach. This result supports the findings in [Law-To 2007].

Dataset	AP		Pred. Time (s)	
	$\alpha = 1$	$\alpha = 0.8$	$\alpha = 1$	$\alpha = 0.8$
`BelgaLogos`	**0.2008**	0.1386	317	31
`Caltech`	**0.0518**	0.0381	359	102
`ImageNetSmall`	**0.3979**	0.2834	45	10
`ImagEval`	**0.2177**	0.1878	71	14
`OxfordBuilding`	**0.6509**	0.5672	267	36
`PascalVOC`	**0.1285**	0.1104	46	12
`PlantLeaves_V1`	**0.8689**	0.6628	7	1
`PlantLeaves_V2`	**0.6464**	0.5457	498	43
Average	**0.3954**	0.3167	201.3	31.1

Table 6.19: Tradeoff between prediction quality and response time (Simple approach).

Figure 6.8 presents the tradeoff between retrieval quality and search time when varying the control parameter α. The curves are generated using the simple approach and are an average over four datasets: `BelgaLogos`, `ImagEval`, `OxfordBuilding` and `PlantLeaves_V1`. We see that when α varies from 0.2 to 0.6, the prediction time increases with a small slope (i.e quasi constant) while the AP increases significantly. In this case, there is no tradeoff since the gain in performance is worth the extra seconds needed in prediction. On the other hand, the tradeoff is clear when $\alpha > 0.6$. In fact, we see that the search time increases with a very steep slope while the performance continues with almost the same slope.

Dataset	AP		Pred. Time (s)	
	$\alpha = 1$	$\alpha = 0.8$	$\alpha = 1$	$\alpha = 0.8$
BelgaLogos	0.2320	**0.2459**	21,048	1,330
Caltech	**0.0450**	0.0316	42,748	6,940
ImageNetSmall	**0.2234**	0.2141	2,366	42
ImagEval	0.1660	**0.1902**	8,992	541
OxfordBuilding	0.6871	**0.7127**	11,433	873
PascalVOC	**0.0923**	0.0921	5,244	938
PlantLeaves_V1	**0.7069**	0.6606	80	104
PlantLeaves_V2	**0.6101**	0.5825	3,049	411
Average	**0.3454**	0.3412	11,870	1,397

Table 6.20: Tradeoff between prediction quality and response time (Geometric approach).

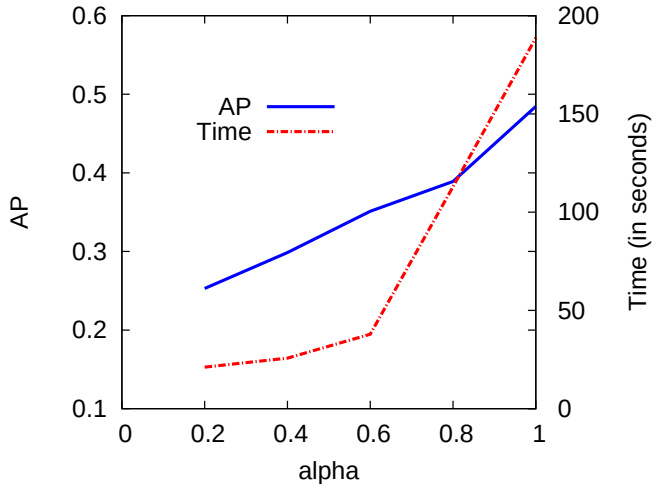


Figure 6.8: Tradeoff between retrieval quality and search time.

Chapter 7

Interpretability and Interactivity Experiments

The greatest challenge to any thinker is stating the problem in a way that will allow a solution.
Bertrand Russell

THE main common measures used for evaluating the predictive accuracy of content-based image retrieval systems and the learning algorithms involved are precision and recall [Huijsmans 2005]. However, understanding the models generated plays a key role for both computer vision experts and end-users. In fact, most shortcomings of computer vision techniques could be easily understood and corrected by end-users if we provided them with an intuitive visual representation of the trained object models. This chapter is intended to evaluate user interactivity. Since our interactive retrieval method is based upon visual keywords, we will first assess the merits of LARK feature selection from a comprehensibility perspective.

7.1 How interpretable are our visual keywords?

Beyond retrieval performance, an important advantage of selecting concise and meaningful visual keywords is to provide users with a better understanding of the retrieval capabilities of a given trained model. In this section, we present some of the visual keywords representing our models. Recall that these visual keywords are regions extracted around the selected interest points with a representation that takes into account the window size used in the description. Furthermore, they are presented in a descending order according to their

prediction weights. The patches we present here are intended to shed some light on the numerical results given in the previous chapter.

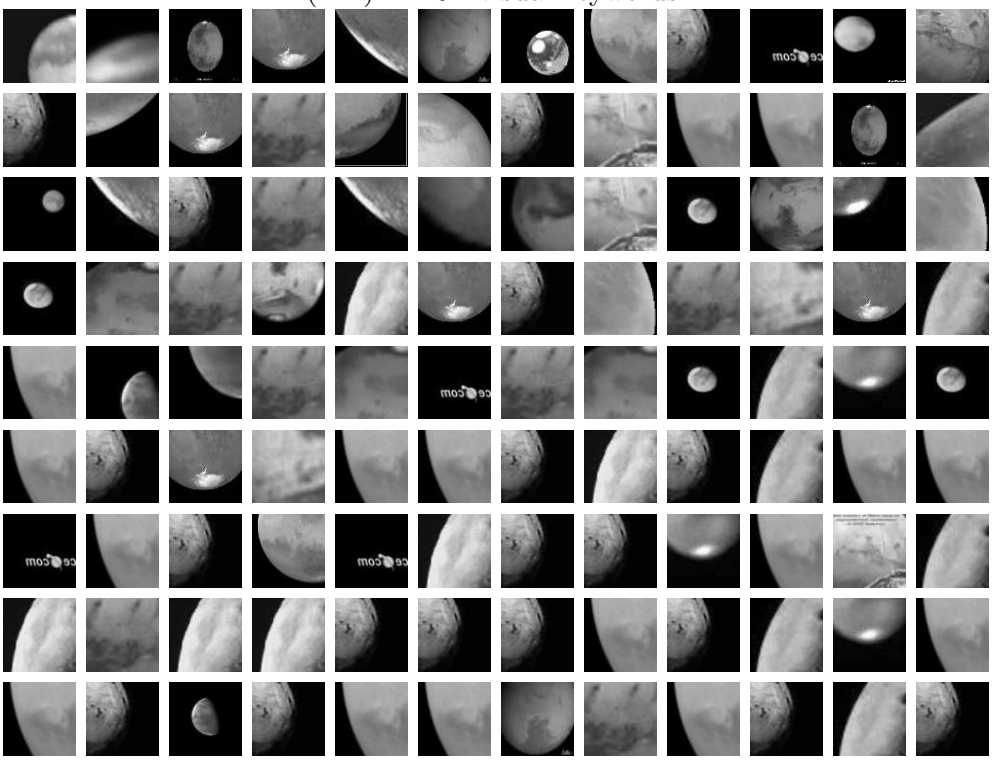
7.1.1 Visual interpretation of the quantitative results

Interpretable models help to analyze quantitative results. In this section, we provide some explanations of the experiments described in the previous chapter.

7.1.1.1 Interpretability of overfitting and overtraining

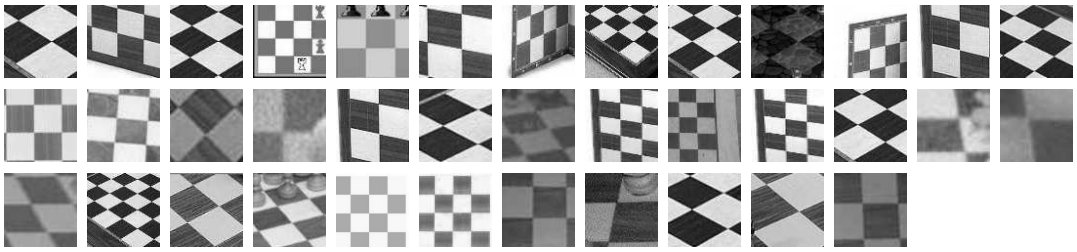
We first compare the feature selection of LARK (which uses BLasso) with [Opelt 2006] (which uses AdaBoost and which is referred to as Opelt06). The visual keywords presented here correspond to some object categories of the `Caltech` dataset, and whose quantitative results were given in table 6.4. Figure (7.1-A) shows both models of the 137.mars class. We notice that among the large number of the visual keywords selected by Opelt06, many are repeated. However, each visual keyword corresponds to a separate hypothesis having its own discriminant radius. This observation doesn't exclude the fact that these components are highly correlated, thus resulting in a poor prediction compared to the twelve visual keywords provided by LARK. In addition, this case is a clear illustration of AdaBoost overtraining and concurs the quantitative results presented in the previous chapter (cf. section 6.2.2). On the other hand, and with the 037.chess-board category (cf. figure 7.1-B), Opelt06 failed to generalize. In fact, AdaBoost stopped quickly and couldn't prevent overfitting. As a result, the prediction results are poor compared to LARK. We can state that, in various cases, LARK selects the most interesting and reliable visual keywords of the object class. For the 037.chess-board category, although some visual keywords selected by LARK may seem to be somehow similar, they differ slightly by an affine geometric transformation (to which neither SIFT nor SURF is invariant). The question raised here then relates to the descriptor strength rather than the learning algorithm.

(A-1) LARK visual keywords.



(A-2) Opelt06 visual keywords.

(A) 137.mars object category.

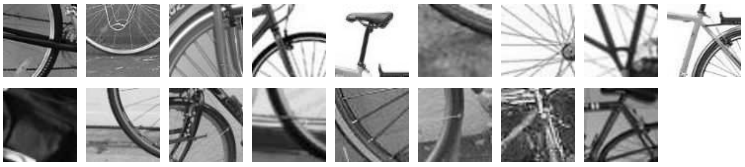


(B-1) LARK visual keywords.

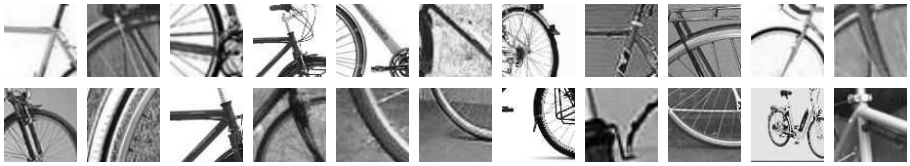
(B-2) Opelt06 visual keywords.

(B) 037.chess-board object category.

Figure 7.1: Illustration of the stop condition problem in AdaBoost with two object categories.



(a) SIFT-based visual keywords.



(b) SURF-based visual keywords.

(c) "SIFT+SURF"-based visual keywords.

Figure 7.2: Visual patches of 224.touring-bike category.

7.1.1.2 Interpretability of combining various types of description

Learning object categories and image features are definitely descriptor-related. Theoretically, adding more descriptions should give better results as shown in the previous chapter. However, in general (i.e average on all the visual object categories), this was not the case for all the object classes. In fact, some classes underwent a big reduction in the model size (i.e. 146.mountain-bike and 204.sunflower-101) which corresponds to a 20% loss in AP versus a 61% gain in reducing the number of visual keywords. For others, there is no clear answer: quantitatively, numbers don't reveal any improvement and qualitatively, interpreting an increase or a decrease in a descriptor performance by relying on visual patches is not obvious, mainly when these patches are parts from the object category. An illustrative example is given in figure 7.2.

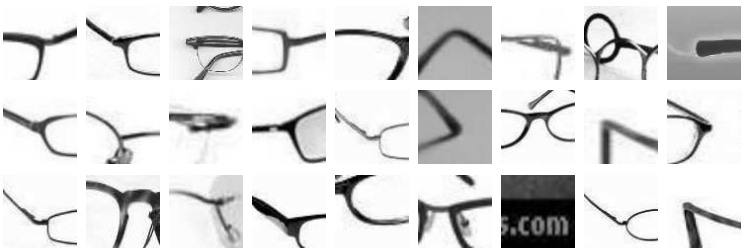
7.1.1.3 Interpretability of adding a counter-class

Defining the opposite category in order to learn a visual concept is by no means trivial. We may speculate about the opposite of a motorbike or the opposite of the American flag! The choice of negative examples must certainly take into account the composition of the test database. In addition, this choice must be as varied as possible in order to cover all possibilities. Yet it is statistical learning and the training data distribution may differ from the data to be predicted.

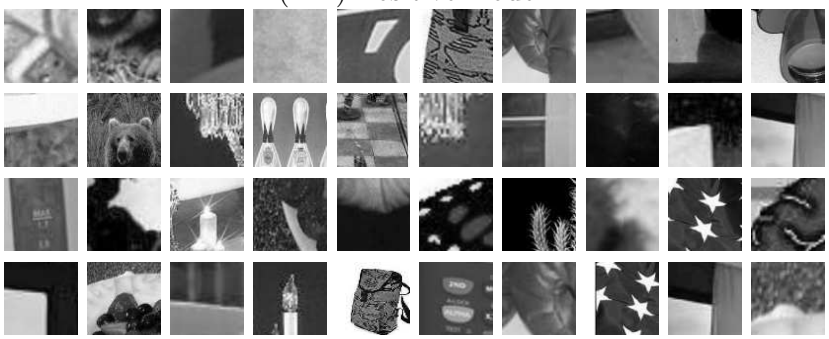
Our visual keywords allow the counter-class effectiveness to be interpreted visually. To illustrate the idea, we selected four categories from the `Caltech` dataset, two of which showed a remarkable improvement in prediction results. Figure 7.3 presents the two successful categories: 067.eyeglasses and 177.saturn. For these categories, we notice a high contrast between the background and the parts of the object. This creates a high edge response. It follows that the corresponding signatures are very distinctive versus the signatures of the counter-class. In addition, the shapes included in the opposite model are easily distinguishable from those belonging to the category, and we believe that is the key to the increase in performance. On the other hand, both positive and negative models of 086.golden-gate-bridge category have some similarity (cf. figure 7.4). Some patches indeed look homogeneous and almost uniform (i.e. the norms of gradients are very small). This fact made it more difficult to discern between what the object is and what it is not. Finally, the last example we present is the model of the 150.octopus category. Here again we notice that negatives bear a resemblance to positives since the descriptors used are robust to the negative transformation (cf. figure 7.4.B-3). This example is a failure case where the negative model is not statistically discriminant. That is, the negative images which were randomly chosen formed a small set (not large enough to cover variability) and were in some way "similar" to the object category. Thus, the learning algorithm concentrated on discerning between these examples rather than on generalizing.

7.1.1.4 Hierarchical retrieval

In this section, we show visually how LARK is able to summarize the visual keywords of higher semantic concepts. Figure 7.5 presents the visual

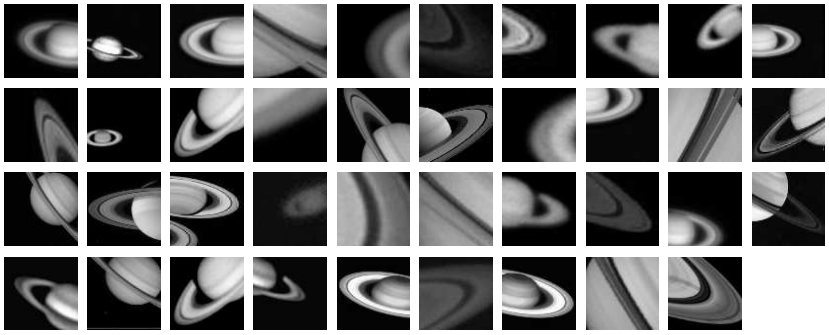


(A-1) Positive model.



(A-2) Counter-class model.

(A) 067.eyeglasses object category.



(B-1) Positive model.

(B-2) Counter-class model.

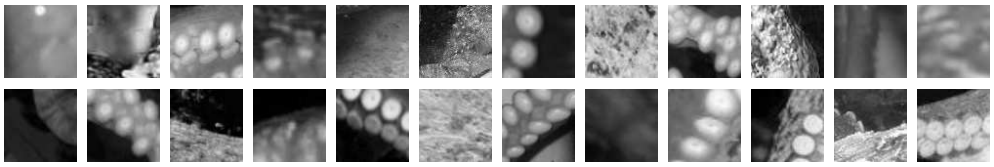
(B) 177.saturn object category.

Figure 7.3: Illustration of the case where adding a counter-class improves prediction. Visual keywords of 067.eyeglasses and 177.saturn categories presented with their respective counter-classes.

(A-1) Positive model.

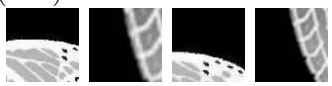
(A-2) Counter-class model.

(A) 086.golden-gate-bridge object category.



(B-1) Positive model.

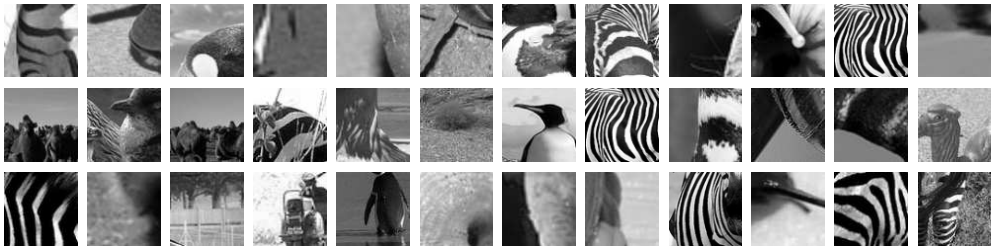
(B-2) Counter-class model.



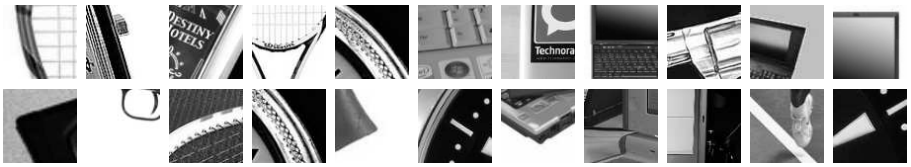
(B-3) Inverted colors of some visual keywords belonging to the counter-class model.

(B) 150.octopus object category.

Figure 7.4: Illustration of the case where adding a counter-class decreases results. Visual keywords of 086.golden-gate-bridge and 150.octopus categories presented with their respective counter-classes.



(a) The concept *animal:* camel, penguin, snail and zebra.



(b) The concept *man-made:* laptop, revolver, tennis_racket and watch.

(c) The concept *vegetation:* sunflower and tomato.

Figure 7.5: Visual keywords of higher semantic concepts.

keywords of the concepts *animal*, *man-made* and *vegetation* which were defined in section 6.3. From a global view, we see that the visual keywords are well related to the concepts defined. However, the proportion of the visual keywords attributed to each sub-concept differs. This is to be expected because the algorithm has to cover the intra-variability which may differ from one sub-concept to another. By analogy to usual categories, say *bike* for instance, the algorithm learns which parts of the object vary the most. Yet it should be noticed that all of the sub-categories were represented by at least two informative patches despite the model concision—particularly for the concepts where the number of sub-categories is relatively high. Notice, for instance, tentacles of snails in the *animal* concept and a trigger and a percussion cap in the *man-made* concept.

7.1.2 Interpretability of visual keywords ambiguity

An ambiguous visual keyword is a visual keyword that can fit two or more visual concepts. In other words, and by analogy to text, it is a word that has more than one meaning. Ambiguity of visual keywords can be a major bottleneck for both prediction accuracy and human interpretation.

Two examples illustrating the influence of ambiguity on prediction are shown in figures 7.6 and 7.7. They present the model *armoured_vehicle* belonging to the `ImagEval` dataset with the prediction matches in two (unseen) test images. The visual keywords were spread above and below the images to provide a better representation. In figure 7.6, we see that there are two visual keywords with a clear edge that matched the license plate of a car and the belt of a person. These same two patches matched, in figure 7.7, the roof of a building. Another ambiguous patch, which is highlighted in pink in figure 7.6, matched the head (hair) of a person. These "false" matches (i.e. they are numerically good and semantically correct but they are supposed to match armoured vehicle objects) can affect the retrieval accuracy by assigning high scores to images that are irrelevant to the concept being sought. Notice the windshield visual keyword, highlighted in light green in figure 7.7, that correctly matched the vehicle. Although correct, this visual keyword is still ambiguous without a context.

7.1.3 Description limitation

It happens that a visual keyword is perfectly understandable by humans but that it may sometimes fail to perfectly match the same visual concept. In fact, and in this case, the computational representation of the descriptor cannot encode the semantics behind the visual keyword.

This semantic gap is shown in figure 7.7 where two visual keywords "tires" falsely match the headlights of the vehicle, although in this case, the retrieval performance of the concept *armoured_vehicle* would not have been affected. A second example, which can however negatively affect retrieval results, is given in figure 7.8. Here, the clarity of the *sunglasses* patch didn't prevent the machine from confusing this object with the side of a stadium. Another case of description limitation is given in figure 7.9. The descriptor robustness

Figure 7.6: Armoured vehicle model (`ImagEval`).

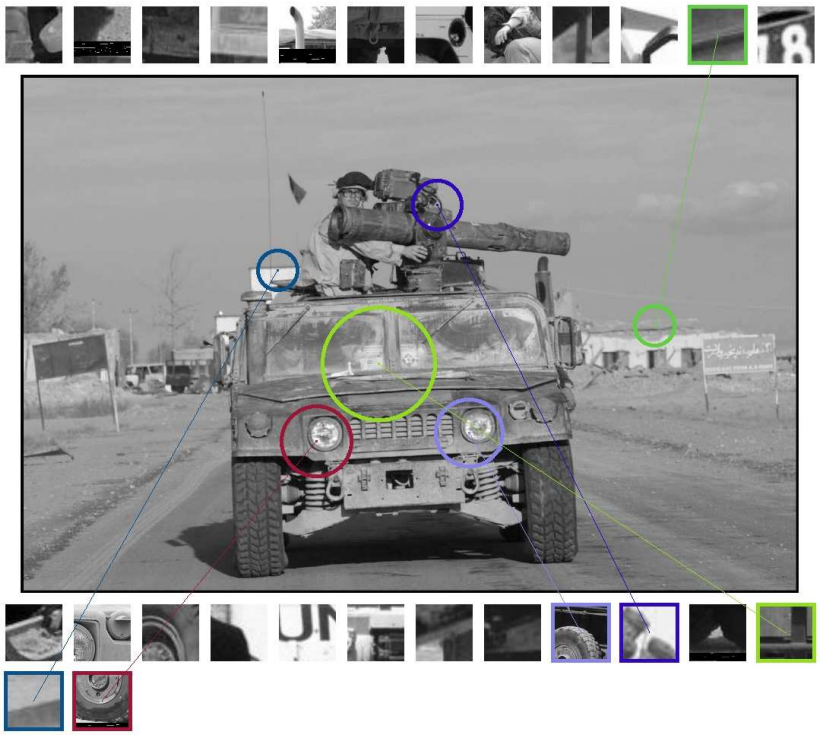


Figure 7.7: Armoured vehicle model (`ImagEval`).

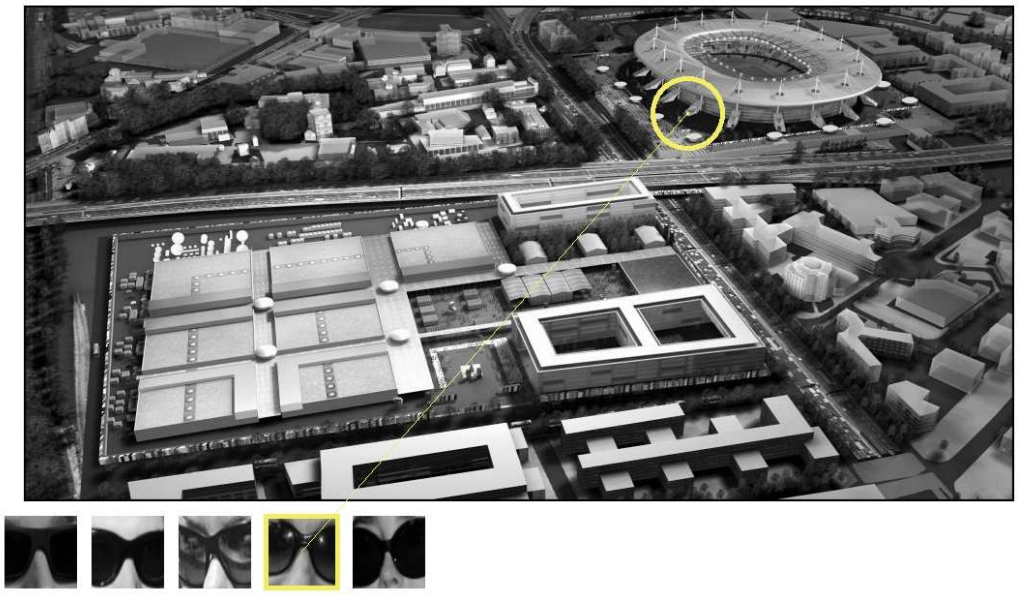


Figure 7.8: Sunglasses model (`ImagEval`).

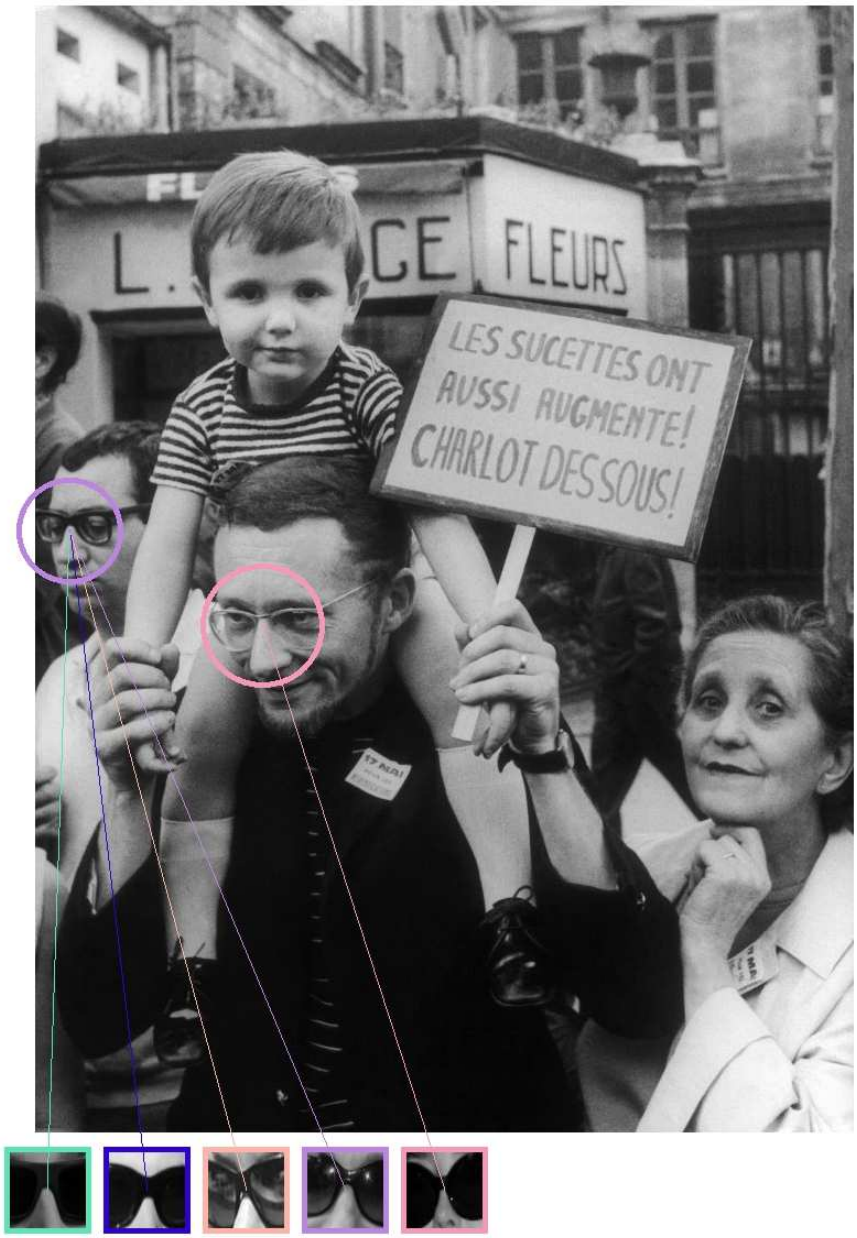


Figure 7.9: Sunglasses model (`ImagEval`).

allows all of the five *sunglasses* visual keywords to be matched with two pairs of spectacles. Despite the high semantic link, returning this image among the first ranked results is considered to be irrelevant to the query. Reducing the semantic gap can be achieved by considering geometric coherence between sets of interest points.

7.1.4 Applying geometric consistency

Applying the geometric approach described in section 4.4 can help to locate small and complex objects characterized by fixed shape and appearance. An example is given in figure 7.10. Note here that each visual keyword is composed of a structured set of interest points. Therefore, for a given patch, there is a set of interest points that is the best set that verifies the coherence of the affine transformation. In our representation, the position of the circle center corresponds to the central interest point belonging to the matched set.

Paradoxically, some objects are difficult to characterize due to their *simplicity* (i.e their lack of informative content). Take for instance the logo *Nike*. Using our `BelgaLogos` training dataset defined in the previous chapter, there were only eight relevant visual keywords to the concept (cf. figure 7.15), and in general, we were not able to correctly locate the logo in test images. This is because the maximum number of descriptors used per image is not sufficient to cover the particularity of this object (at most, there were three points that describe the object and its context). Therefore, we decided to carry out another experiment by keeping the same training data (i.e 58 images for the object class) but with a very high number of descriptors per image. We used the ASIFT descriptors with a maximum of $20,000$ points per image. This generated $1,058,455$ local features belonging to the object category. It is actually of interest to test the sparsity of the feature selection given this high number of input features. There were 127 visual keywords selected, and they are presented in figure 7.11. From the patches displayed, we estimate that the selection result is good. This figure also illustrates the semantic gap discussed earlier. In fact, there are two logos on a soccer ball (i.e two patches) that matched an eyebrow and the bottom of the nose of the player.

Notice the confusion that the *Nike* shape can engender. The model includes some noses, a thumb, T-shirt creases, etc. Figure 7.12 shows how

Figure 7.10: Puma model (`BelgaLogos`).

the eyebrow patch (4^{th} row, 4^{th} column) matched the logo on the player's wristband. We can also notice that the ASIFT descriptors used here are sensitive to color inversion (visual keywords with a white background match image features with a white background and vice versa). We can think of making them robust by taking into account the sign of the Laplacian of the blob detected, but this could be a double-edged sword (consider the example given in figure 7.4-B).

7.2 Interaction with the visual keywords

In the previous section, we showed that LARK visual keywords allow most of the known machine learning shortcomings (overfitting, opposite class, context, etc.) to be easily interpreted visually. In this section, we will try to evaluate the benefit of interactivity using our GUI interface described in chapter 5. As discussed, and as shown from some screenshots, users have the possibility to interact with the models by picking and re-weighting the visual keywords that best represent the visual concept they are looking for. They can indeed emphasize the object scale, context, relative pose and/or appearance. They can also discard ambiguous and contextual irrelevant words. Furthermore, they can create "sub-concepts" from the initially available patches.

For evaluation purposes, the problem is to create a ground truth for each user-specialized query. Suppose, for example, that the user chooses some visual keywords that correspond to "tires" from the initial model of the *armoured_vehicle* concept. Images relevant to this query may include some images that are not related to the parent concept (cf. figure 7.6 and 7.13), and may exclude other images like those including caterpillar-tracked tanks or which contain armoured vehicles with no visible tires (i.e the image is cropped).

Generating a ground truth is labor intensive and error-prone considering the size of the datasets. Therefore, we decided to adopt the P@n measure (cf. section 6.1.4) to compare the performance with and without model specialization. To make the evaluation task easier, we avoided constructing sub-concepts. The idea is to keep, for a given visual category, only the relevant patches by discarding any contextual, ambiguous and/or irrelevant visual

Figure 7.11: Nike model (`BelgaLogos`).

Figure 7.12: Nike model (`BelgaLogos`).

Figure 7.13: Armoured vehicle model (`ImagEval`).

Figure 7.14: Specialized *Adidas-text* model.

Figure 7.15: Specialized *Nike* model.

keywords. We present the results of three object categories: *Adidas-text* and *Nike* from the `BelgaLogos` dataset, and *Magdalen* from the `OxfordBuilding` dataset. The usual words used for querying the search engine are given respectively in figures 7.14, 7.15 and 7.16, and the results are presented in tables 7.1, 7.2 and 7.3. They all show that filtering out unrelated patches enhances the retrieval performance. The *Adidas-text* model was particularly good at all levels. Going from a cut-off of 10 to 50, the precision should normally decrease, which is not the case here. This is because—apart from a few irrelevant images—some of the top ranked images included the *Adidas* logo without the text (which is understandable from the visual keywords used for querying). For the two other categories, the precision at the first levels for the specialized models is always better. It outperformed the precision of the initial models up to P@30, then it became worse. From this observation, we understand that the full model is more generalizable. In fact, it summarizes intra-class variability better than the specialized model, which only allowed the images related to the keywords it comprises to be top ranked.

During our interactive experiments, we noticed that it is possible for a single patch not to perform well if it is used alone, even if it is interpretable enough for a human. There are two possible explanations for this. The first

Figure 7.16: Specialized *magdalen* model.

	ALL visual keywords	Specialized
P@10	0.3	0.6
P@20	0.2	0.75
P@30	0.1333	0.8333
P@40	0.1	0.825
P@50	0.08	0.86

Table 7.1: Comparing precision with and without model specialization for the *Adidas-text* category (`BelgaLogos`, retrieval from 9,476 images).

	ALL visual keywords	Specialized
P@10	0.1	0.6
P@20	0.2	0.35
P@30	0.23	0.33
P@40	0.325	0.25
P@50	0.28	0.24

Table 7.2: Comparing precision with and without model specialization for the *Nike* category (`BelgaLogos`, retrieval from 9,476 images).

	ALL visual keywords	Specialized
P@10	0.9	0.9
P@20	0.7	0.8
P@30	0.56	0.6
P@40	0.525	0.475
P@50	0.46	0.38

Table 7.3: Comparing precision with and without model specialization for the *Magdalen* category (`OxfordBuilding`, retrieval from 3,115 images).

Full model.

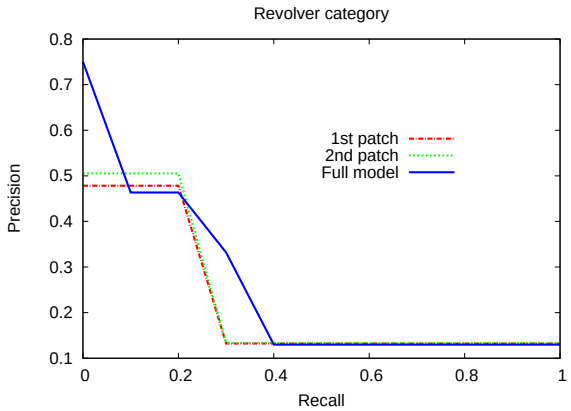


Figure 7.17: Performance of the *revolver* category (`ImageNetSmall`).

possible cause is that our visual representation of the patch (gray-scale image corresponding to the description window of an interest point/a set of interest points) doesn't faithfully reproduce the information the descriptor characterized (i.e. semantic gap). The second explanation is that, statistically speaking, the database may be somehow skewed so that a single feature cannot be very discriminative. This problem is illustrated in figure 7.17 by the *revolver* category. The full model comprises only two visual keywords which are shown at the top of the figure and their corresponding precision-recall curves are given below. Compare the precision for the first ranked results (say 5% recall), we see that the full model behaves quite well while each feature, alone, does not.

Chapter 8

Conclusion

Success isn't permanent, and failure isn't fatal.
Mike Ditka

8.1 Synthesis

THIS thesis addresses the problem of building humanly interpretable visual models in the context of object retrieval in image collections. Our motivation was that most shortcomings of computer vision techniques could be easily understood and corrected by end-users if we provided them an intuitive visual representation of the trained object models.

In that way, we first introduced the notion of visual keywords by analogy to visual words (used in many visual information retrieval systems) and to common keywords (used in text indexing and search) and we set our objectives as readability, conciseness and disambiguation. The idea is to represent a given object, object class or visual concept by a concise set of image patches that a user can easily handle and interpret. The problem is then to select, among all possible patches of the training images, the most discriminative subset, while meeting the constraint of conciseness in the number of patches.

We showed that, when using an appropriate high dimensional and sparse embedding of local features, this objective can be formulated as a discriminative learning problem with an L_1-regularization cost to enforce conciseness. To efficiently solve this problem, we then adapted an existing algorithm called *BLasso*—that approximates the exact solution of the *Lasso* path with a boosting-like procedure—to deal with a discriminative and a multiple-instance learning problem rather than a common feature selection. We named this algorithm LARK. We showed in the experiments that, on average, the feature selection by our method is always better than the feature selection performed

by a state-of-the-art method based on the AdaBoost algorithm. The reason for this is that the stop condition of AdaBoost is unclear and, in some ways, problematical. Choosing a large number of hypotheses normally leads to a better precision score but encourages overtraining. That means that many selected features happen to be correlated, and their joint collaboration which is intended to increase the average precision is almost always insignificant considering the linear increase in prediction time. AdaBoost sometimes avoids overtraining but tends to overfit, particularly with a reduced number of training images. In fact, it stops after selecting only few weak learners, which inevitably leads to a poor prediction. Using BLasso overcame this issue by stopping once the intra-class variability and inter-class differences have been efficiently learnt. Furthermore, we showed that both algorithms perform similarly for the top ranked images (from 20% to 40% recall) when AdaBoost doesn't overfit, and we registered a clear superiority of our method when it does. In this second case, we noticed a slight increase in the average of the total number of the visual words selected by BLasso. Here, BLasso tries to generalize considering the few images in the training set. We also concluded that it is always better to use various descriptor types to enhance the overall performance.

In order to reduce the ambiguity of the trained visual keywords, we then introduced an alternative approach, making use of local geometric constraints. We therefore replaced the single-feature representation of image patches by structured sets of local features. In this case, the membership function of a given patch in the other images is computed as a geometrically consistent matching in the whole dataset. Using this technique, we obtained a higher performance for rigid visual objects that don't vary much in appearance (e.g. logos), particularly when using an approximative search method. Moreover, we showed that, even with an extremely high number of descriptors, our feature selection performs very well in terms of quality and sparsity as long as the visual objects are encoded in some of these input descriptors. This highlights the interest of using a multiple-instance learning approach.

Finally, we focused our work on how to use the trained visual keywords for efficient retrieval and user interactivity. We first introduced several possible user interactions including eliminating ambiguous visual keywords and

emphasizing some object parts and/or appearance. We also pointed out how the semantic gap can have an adverse effect on retrieval performance. Furthermore, we integrated the proposed interactions in GUI allowing for user experiments and we showed numerically that specializing models improves retrieval precision.

Along with the quantitative results, we presented some visual models in an attempt to explain the quantitative results. We showed how straightforward it is easy to manipulate concise models and how our visual keywords allowed for interpretability.

8.2 Applications

Visual keywords are a straightforward tool. Our research can be used in various computer vision applications including event retrieval and plant identification to name but two.

Organizing media according to real-life events is gaining ever-increasing interest in the multimedia community. Event-centric indexing approaches are indeed very promising for discovering more complex relationships between data. Time and geographic information, jointly provided with media content, has of course a major role to play, but using visual content as complementary information might solve several limitations of the approaches that rely only on metadata. A typical application would be the following: given a query event record, represented by a set of photos, the task is to retrieve other records of the same event, typically generated by distinct users. In this context, learning discriminative visual keywords representing the event is meaningful. An example of a geometrically consistent visual keyword belonging to a set of pictures representing the event "Myanmar protest" is given in Figure 8.1.

If agricultural development is to be successful and biodiversity is to be conserved, then accurate knowledge of the identity, geographic distribution and uses of plants is essential. Unfortunately, such basic data and information is often only partially available for professional stakeholders, teachers, scientists and citizens, and often incomplete. One noticeable consequence is that simply identifying plant species is often a very difficult task; even for botanists themselves (the so-called taxonomic gap). Computer vision and visual infor-

Figure 8.1: Event search: an example of a geometrically consistent visual keyword.

mation retrieval are considered as very promising means of bridging this gap. However it is fundamental that botanists can interact on the trained visual models to control which plant's organs have been selected as an identification key. We believe that LARK is suitable for such supervised learning issues where end-users want to control and understand exactly what the machine did learn.

8.3 Perspectives

As we can learn from the experiments conducted, visual word ambiguity not only affects prediction accuracy but also human interpretation and subsequently user interactivity. In this direction, we have begun an experiment to measure the interpretability level of the visual keywords with respect to each type (or a combination of types) of descriptions. More precisely, the experiment consists in determining the rank of the visual keyword at which a given category is recognized by a human. The experiment is to be conducted with several users and various object categories. For this purpose, we built a user interface (cf. figure 8.2) that provides the visual keywords (of a random selected category, which is unknown to users) after selecting one or many description types. The visual keywords are displayed one by one each time users click the button "next". They are given in a descending order according to their predictive weights learned during training. Each time a patch is displayed, users can guess the visual concept these patches belong to. In fact, they can type whatever word comes in mind in a small box. The system automatically filters out the category names that don't match the input text. Along with the possible choices kept, a small picture is jointly displayed with

Figure 8.2: Visual word interpretation.

each proposition for better understanding. This experiment is still in progress and should be concluded soon.

In our next work, we will carry out more in-depth tests on description combinations. Although it is more effective on average, we noticed that—for a particular object category—the combination of local descriptors may sometimes have lower precision than if one of these descriptors is used alone. This behavior is not expected since the feature selection mechanism may discard any unreliable descriptors. In addition, we will try other additive constraints than the L_1-norm, particularly, the adaptive group lasso. The group lasso allows selection in a grouped manner. Therefore, we can focus on some image regions more than others. Yet this necessitates additional annotations and might affect the multiple-instance learning.

The performance of boosting procedures crucially depends on the choice of the weak learner. In this perspective, it is interesting to try out different discriminative boundaries. Rather than using hyperspheres, ellipsoids might, for instance, be used. This would only affect the classification error rate and would not change our image representation.

Since user interactivity is entirely based on visual words, it is necessary to provide a patch representation that faithfully reproduces the actual information which is coded by the descriptor. The intuitive idea is to use the aspects that the descriptor characterizes. However, coming up with an easily understandable representation is not always obvious and should probably be

followed by the original image patch so that users can spot what is numerically coded through patch comparison.

The visual words learnt act separately in the sense that each visual word contributes to the classification score without considering the other components of the model. Although we suggested richer visual words built from sets of local features, we might consider a more complex representation that takes into account the spatial configuration between these words. This would allow higher semantical levels to be achieved.

Introducing geometric consistency was one way to demonstrate the flexibility of our method. It is based on the fact that we can choose any similarity measure to rank images, and this choice can vary and be specifically designed for each type of local descriptor. Such flexibility allows us to cope with other types of multimedia documents. We can indeed apply the same mechanism to text, audio and video data, either separately or in a multimodal fashion.

Although current methods used in solving text documents problems are very advanced, our method can still be applied to automatically discover topics and provide the keywords that go with them. Audio and video features can also be processed in the same way that images are. Moreover, indexing videos usually takes into account image frames and audio content simultaneously. Here, we can process each part of the content separately then consider formulating more compound and semantical queries using logical operators.

Web content usually contains documents including text, images and frequently audio and video information. The same mechanism could be utilized to retrieve multimodal documents. The keywords defining the model would be a mixture of all these types of media.

The design of multimodal interfaces is considered to be a separate research area since the interface has to be as interactive as possible, and designers must put a great deal of thought into facilitating human-machine communication. However, we believe that our simple design could work, at least to some extent. In addition to the visual keywords (i.e. image patches) that may originate from different types of description, we could provide textual, audio or video keywords provided that we make a clear separation for each type of media so that users could easily distinguish between the different keywords they are dealing with. Since documents are ranked according to the number

of multimodal keywords found, two documents may have the same ranking score but be of different modes. Here, an additional option would be needed to help users to select one mode over another.

Bibliography

[Agarwal 2004] Shivani Agarwal, Aatif Awan and Dan Roth. *Learning to Detect Objects in Images via a Sparse, Part-Based Representation*. IEEE Trans. Pattern Anal. Mach. Intell., vol. 26, pages 1475–1490, November 2004. 23

[Agarwal 2006] Ankur Agarwal and William Triggs. *Hyperfeatures - Multilevel Local Coding for Visual Recognition*. In ECCV International Workshop on Statistical Learning in Computer Vision, 2006. 23, 38

[Agin 1973] Gerald J. Agin and Thomas O. Binford. *Computer description of curved objects*. In IJCAI'73: Proceedings of the 3rd international joint conference on Artificial intelligence, pages 629–640, San Francisco, CA, USA, 1973. Morgan Kaufmann Publishers Inc. 3

[Aizerman 1964] M. A. Aizerman, E. M. Braverman and L. I. Rozonoer. *The probability problem of pattern recognition learning and the method of potential functions*. Automation and Remote Control, pages 1175–1190, 1964. 26

[Amores 2007] Jaume Amores, Nicu Sebe and Petia Radeva. *Context-Based Object-Class Recognition and Retrieval by Generalized Correlograms*. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 29, no. 10, pages 1818–1833, 2007. 35, 38

[Andrews 2003] S. Andrews, I. Tsochantaridis and T. Hofmann. *Support vector machines for multiple-instance learning*. In Advances in Neural Information Processing Systems 15, pages 561–568, Cambridge, MA, 2003. MIT Press. 27

[Asada 1986] H. Asada and M. Brady. *The curvature primal sketch*. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 8, no. 1, pages 2–14, 1986. 14

[Bai 2010] Xue Bai and Siwei Luo. *Combining Bag of Words Model and Information Theoretic Method for Image Clustering*. In Zhigang Zeng and Jun Wang, editeurs, Advances in Neural Network Research and Applications, volume 67 of *Lecture Notes in Electrical Engineering*,

pages 423–430. Springer Berlin Heidelberg, 2010. 10.1007/978-3-642-12990-2_48. 24

[Bar-Hillel 2008] Aharon Bar-Hillel and Daphna Weinshall. *Efficient Learning of Relational Object Class Models*. Int. J. Comput. Vision, vol. 77, pages 175–198, May 2008. 26, 35, 39, 42, 45

[Baumberg 2000] Adam Baumberg. *Reliable Feature Matching across Widely Separated Views*. In CVPR, pages 1774–1781, 2000. 13, 20

[Bay 2006] Herbert Bay, Tinne Tuytelaars and Luc Van Gool. *SURF: Speeded Up Robust Features*. In 9th European Conference on Computer Vision, Graz Austria, May 2006. 16, 20

[Bay 2008] Herbert Bay, Andreas Ess, Tinne Tuytelaars and Luc Van Gool. *Speeded-Up Robust Features (SURF)*. Comput. Vis. Image Underst., vol. 110, pages 346–359, June 2008. 16, 20, 101

[Belongie 2002] S. Belongie, J. Malik and J. Puzicha. *Shape Matching and Object Recognition Using Shape Contexts*. IEEE Trans. Pattern Anal. Mach. Intell., vol. 24, pages 509–522, April 2002. 18

[Berchtold 1996] Stefan Berchtold, Daniel A. Keim and Hans-Peter Kriegel. *The X-tree: An Index Structure for High-Dimensional Data*. In Proceedings of the 22th International Conference on Very Large Data Bases, VLDB '96, pages 28–39, San Francisco, CA, USA, 1996. Morgan Kaufmann Publishers Inc. 46

[Berchtold 1998] Stefan Berchtold, Christian Böhm and Hans-Peter Kriegal. *The pyramid-technique: towards breaking the curse of dimensionality*. In Proceedings of the 1998 ACM SIGMOD international conference on Management of data, SIGMOD '98, pages 142–153, New York, NY, USA, 1998. ACM. 46

[Binford 1989] Thomas O. Binford. *Spatial understanding: the successor system*. In Proceedings of a workshop on Image understanding workshop, pages 12–20, San Francisco, CA, USA, 1989. Morgan Kaufmann Publishers Inc. 11

[Bishop 2007] Christopher M. Bishop. Pattern Recognition and Machine Learning (Information Science and Statistics). Springer, 1st ed. 2006. corr. 2nd printing édition, October 2007. 25

[Botterill 2008] Tom Botterill, Steven Mills and Richard Green. *Speeded-up Bag-of-Words algorithm for robot localisation through scene recognition*. In Image and Vision Computing New Zealand, pages 1–6, November 2008. 23

[Bouchard 2005] Guillaume Bouchard and Bill Triggs. *Hierarchical Part-Based Visual Object Categorization*. In Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1 - Volume 01, CVPR '05, pages 710–715, Washington, DC, USA, 2005. IEEE Computer Society. 38, 42

[Boughorbel 2005] S. Boughorbel, J-P. Tarel and N. Boujemaa. *The Intermediate Matching Kernel for Image Local Features*. In International Joint Conference on Neural Networks (IJCNN'05), jul 2005. 21

[Boureau 2010] Y-Lan Boureau, Francis Bach, Yann LeCun and Jean Ponce. *Learning mid-level features for recognition*. Computer Vision and Pattern Recognition, IEEE Computer Society Conference on, vol. 0, pages 2559–2566, 2010. 2, 23

[Bres 1999] Stéphane Bres and Jean-Michel Jolion. *Detection of Interest Points for Image Indexation*. In VISUAL '99: Proceedings of the Third International Conference on Visual Information and Information Systems, pages 427–434, London, UK, 1999. Springer-Verlag. 15

[Brown 2005] Matthew Brown, Richard Szeliski and Simon Winder. *Multi-Image Matching Using Multi-Scale Oriented Patches*. In Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1 - Volume 01, CVPR '05, pages 510–517, Washington, DC, USA, 2005. IEEE Computer Society. 23

[Canny 1986] J Canny. *A computational approach to edge detection*. IEEE Trans. Pattern Anal. Mach. Intell., vol. 8, pages 679–698, November 1986. 18

[Carneiro 2003] G. Carneiro and A. D. Jepson. *Multi-scale phase-based local features*. IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 1, pages 736–743, June 2003. 20

[Cass 1997] Todd A. Cass. *Polynomial-Time Geometric Matching for Object Recognition.* Int. J. Comput. Vision, vol. 21, no. 1-2, pages 37–61, 1997. 3

[Cheng 2008] Hong Cheng, Zicheng Liu, Nanning Zheng and Jie Yang. *A deformable local image descriptor.* Computer Vision and Pattern Recognition, IEEE Computer Society Conference on, vol. 0, pages 1–8, 2008. 18

[Chum 2004] O. Chum, J. Matas and S. Obdrzalek. *Enhancing RANSAC by Generalized Model Optimization.* In Proc. Asian Conf. Computer Vision, 2004. 33

[Comer 1979] Douglas Comer. *Ubiquitous B-Tree.* ACM Comput. Surv., vol. 11, pages 121–137, June 1979. 46

[Cristianini 2002] Nello Cristianini, Jaz Kandola, Andre Elisseeff and John Shawe-Taylor. *On kernel-target alignment.* In Advances in Neural Information Processing Systems 14, volume 14, pages 367–373, 2002. 44

[Dardas 2010] N. Dardas, Qing Chen, N.D. Georganas and E.M. Petriu. *Hand gesture recognition using Bag-of-features and multi-class Support Vector Machine.* In Haptic Audio-Visual Environments and Games (HAVE), 2010 IEEE International Symposium on, pages 1 –5, 2010. 52

[Datar 2004] Mayur Datar, Nicole Immorlica, Piotr Indyk and Vahab S. Mirrokni. *Locality-sensitive hashing scheme based on p-stable distributions.* In Proceedings of the twentieth annual symposium on Computational geometry, SCG '04, pages 253–262, New York, NY, USA, 2004. ACM. 46

[Deng 2007] Hongli Deng, Wei Zhang, Eric Mortensen and Thomas Dietterich. *L.: Principal curvature-based region detector for object recognition.* In In: Proc. CVPR, 2007. 38

[Deriche 1990] R. Deriche and O. Faugeras. *2D-Curves Matching Using High Curvatures Points : Applications to Stereovision.* Proceedings of the 10th International Conference om Pattern Recognition, vol. 1, pages 240–242, 1990. 14

[Dietterich 1997] Thomas G. Dietterich, Richard H. Lathrop and Tomás Lozano-Pérez. *Solving the multiple instance problem with axis-parallel rectangles*. Artif. Intell., vol. 89, pages 31–71, January 1997. 27

[Efron 2004] Bradley Efron, Trevor Hastie, Iain Johnstone and Robert Tibshirani. *Least angle regression*. Annals of Statistics, vol. 32, pages 407–499, 2004. 63

[Ellouze 2010] M. Ellouze, N. Boujemaa and A. M. Alimi. *IM(S)2: Interactive movie summarization system*. Journal of Visual Communication and Image Representation, vol. 21, no. 4, pages 283–294, 2010. 13

[Epshtein 2007] Boris Epshtein and Shimon Ullman. *Semantic Hierarchies for Recognizing Objects and Parts*. Computer Vision and Pattern Recognition, IEEE Computer Society Conference on, vol. 0, pages 1–8, 2007. 35, 42, 43, 56

[Etiévent 1999] Emmanuel Etiévent, Frank Lebourgeois and Jean-Michel Jolion. *Assisted Video Sequences Indexing: Motion Analysis based on Interest Points*. In ICIAP '99: Proceedings of the 10th International Conference on Image Analysis and Processing, pages 1059–1062, Washington, DC, USA, 1999. IEEE Computer Society. 13

[Fauqueur 2003] J. Fauqueur and N. Boujemaa. *New Image Retrieval Paradigm : Logical Composition of Region Categories*. In ICIP, 2003. 52

[Fauqueur 2004] J. Fauqueur and N. Boujemaa. *Region-based image retrieval: Fast coarse segmentation and Fine color description*, 2004. 12, 18

[Ferecatu 2005] Marin Ferecatu. *Image retrieval with active relevance feedback using both visual and keyword-based descriptors*. PhD thesis, UNIVERSITY OF VERSAILLES SAINT-QUENTIN-EN-YVELINES, July, 1 2005. 12, 13, 101

[Fergus 2003] R. Fergus, P. Perona and A. Zisserman. *Object class recognition by unsupervised scale-invariant learning*. In In CVPR, pages 264–271, 2003. 23, 34, 39, 42, 43

[Fergus 2005] R. Fergus, L. Fei-Fei, P. Perona and A. Zisserman. *Learning Object Categories from Google's Image Search*. In Proceedings of the Tenth IEEE International Conference on Computer Vision - Volume

2, ICCV '05, pages 1816–1823, Washington, DC, USA, 2005. IEEE Computer Society. 2, 23, 37

[Fischler 1973] M. A. Fischler and R. A. Elschlager. *The Representation and Matching of Pictorial Structures*. IEEE Trans. Comput., vol. 22, no. 1, pages 67–92, 1973. 3

[Fischler 1981] Martin A. Fischler and Robert C. Bolles. *Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography*. Commun. ACM, vol. 24, no. 6, pages 381–395, June 1981. 23

[Fisher 1936] Ronald A. Fisher. *The use of multiple measurements in taxonomic problems*. Annals Eugen., vol. 7, pages 179–188, 1936. 26

[Florack 1994] L. M. J. Florack, B. M. Ter Haar Romeny, J. J. Koenderink and M. A. Viergever. *General intensity transformations and differential invariants*. Journal of Mathematical Imaging and Vision, vol. 4, no. 2, pages 171–187, May 1994. 20

[Folsom 2004] T.C. Folsom. *Sparse Scene Sampling for Robot Vision*. Robotics and Applications, 2004. 13

[Forssén 2006] Per-Erik Forssén and Anders Moe. *Autonomous Learning of Object Appearances using Colour Contour Frames*. In Proceedings of the The 3rd Canadian Conference on Computer and Robot Vision, pages 3–, Washington, DC, USA, 2006. IEEE Computer Society. 31, 39

[Föstner 1987] M. A. Föstner and E. Gülch. *A Fast Operator for Detection and Precise Location of Distinct Points, Corners and Centers of Circular Features*. In ISPRS Intercommission Workshop, Interlaken, Switzerland, 1987. 38

[Freeman 1991] William T. Freeman and Edward H. Adelson. *The Design and Use of Steerable Filters*. IEEE Trans. Pattern Anal. Mach. Intell., vol. 13, no. 9, pages 891–906, 1991. 20

[Freund 1997] Yoav Freund and Robert E. Schapire. *A decision-theoretic generalization of on-line learning and an application to boosting*. Journal of Computer and System Sciences, vol. 55, no. 1, pages 119–139, 1997. 61

[Friedman 2000] Jerome Friedman, Trevor Hastie and Robert Tibshirani. *Additive logistic regression: a statistical view of boosting*. Annals of Statistics, vol. 28, 2000. 61

[Frome 2007] Andrea Frome, Fei Sha, Yoram Singer and Jitendra Malik. *Learning globally-consistent local distance functions for shape-based image retrieval and classification*. In In ICCV, 2007. 34, 41, 45

[Fu 2009] Zhouyu Fu and A. Robles-Kelly. *An instance selection approach to Multiple instance Learning*. Computer Vision and Pattern Recognition, IEEE Computer Society Conference on, vol. 0, pages 911–918, 2009. 27

[Gao 2006] Jianfeng Gao, Hisami Suzuki and Bin Yu. *Approximation lasso methods for language modeling*. In Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics, ACL-44, pages 225–232, Stroudsburg, PA, USA, 2006. Association for Computational Linguistics. 63

[Gehler] Peter Gehler and Sebastian Nowozin. *On Feature Combination for Multiclass Object Classification*. 44

[Gemert 2008] Jan C. Gemert, Jan-Mark Geusebroek, Cor J. Veenman and Arnold W. Smeulders. *Kernel Codebooks for Scene Categorization*. In ECCV '08: Proceedings of the 10th European Conference on Computer Vision, pages 696–709, Berlin, Heidelberg, 2008. Springer-Verlag. 30

[Gool 1996] Luc J. Van Gool, Theo Moons and Dorin Ungureanu. *Affine/ Photometric Invariants for Planar Intensity Patterns*. In ECCV '96: Proceedings of the 4th European Conference on Computer Vision-Volume I, pages 642–651, London, UK, 1996. Springer-Verlag. 20

[Gosselin 2007] Philippe H. Gosselin, Matthieu Cord and Sylvie Philipp-Foliguet. *Kernels on bags for multi-object database retrieval*. In Proceedings of the 6th ACM international conference on Image and video retrieval, CIVR '07, pages 226–231, New York, NY, USA, 2007. ACM. 39

[Gouet 1998] Valérie Gouet, Philippe Montesinos and Danielle Pelé. *Stereo Matching of Color Images using Differential Invariants*. In ICIP (2), pages 152–156, 1998. 15, 101

[Gouet 2000] V. Gouet, P. Montesinos, R. Deriche and D. Pelé. *Evaluation de détecteurs de points d'intérêt pour la couleur*. In Reconnaissance des formes et Intelligence Artificielle (RFIA'2000), volume II, pages 257–266, Paris, France, 2000. 15

[Grauman 2005a] Kristen Grauman and Trevor Darrell. *Efficient Image Matching with Distributions of Local Invariant Features*. In Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 2 - Volume 02, CVPR '05, pages 627–634, Washington, DC, USA, 2005. IEEE Computer Society. 21

[Grauman 2005b] Kristen Grauman and Trevor Darrell. *The pyramid match kernel: Discriminative classification with sets of image features*. In In ICCV, pages 1458–1465, 2005. 11, 33

[Griffin 2007] G. Griffin, A. Holub and P. Perona. *Caltech-256 Object Category Dataset*. Rapport technique 7694, California Institute of Technology, 2007. 100

[Grove 1998] Adam J. Grove and Dale Schuurmans. *Boosting in the limit: Maximizing the margin of learned ensembles*. In In Proceedings of the Fifteenth National Conference on Artificial Intelligence, pages 692–699, 1998. 61

[Hamzaoui 2010] A. Hamzaoui, A. Joly and N. Boujemaa. *Multi-source shared nearest neighbours for multi-modal image clustering*. Multimedia Tools and Applications, 2010. 13

[Harris 1988] C. Harris and M. Stephens. *A combined corner and edge detector*. In Proceedings of the 4th Alvey Vision Conference, pages 147–151, 1988. 15, 38

[Hastie 2009a] Trevor Hastie, Robert Tibshirani and Jerome Friedman. The elements of statistical learning: data mining, inference and prediction. Springer, 2 édition, 2009. 70

[Hastie 2009b] Trevor Hastie, Robert Tibshirani and Jerome Friedman. The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition. Springer Series in Statistics. Springer, 2nd ed. 2009. corr. 3rd printing5th printing. édition, September 2009. 25

[Haugeard 2009] J.-E. Haugeard, S. Philipp-Foliguet, F. Precioso and J. Lebrun. *Extraction of Windows in facade using Kernel on Graph of Contours*. In 16th Scandinavian Conference on Image Analysis, volume 5575 of *Lecture Notes in Computer Science*, pages 646–656. Springer, June 2009. 39

[Henrich 1989] A. Henrich, H. W. Six and P. Widmayer. *The LSD tree: spatial access to multidimensional and non-point objects*. In Proceedings of the 15th international conference on Very large data bases, VLDB '89, pages 45–53, San Francisco, CA, USA, 1989. Morgan Kaufmann Publishers Inc. 46

[Henrich 1998] Andreas Henrich. *The LSDh-Tree: An Access Structure for Feature Vectors*. In Proceedings of the Fourteenth International Conference on Data Engineering, ICDE '98, pages 362–369, Washington, DC, USA, 1998. IEEE Computer Society. 46

[Hervé 2009] N. Hervé and N. Boujemaa. *Visual word pairs for automatic image annotation*. In IEEE International Conference on Multimedia and Expo (ICME09), New York, jun 2009. 16

[Horaud 1990] R. Horaud, T. Skordas and F. Veillon. *Finding geometric and relational structures in an image*. Proceedings of the 1st European Conference on Computer Vision., pages 374–384, 1990. 14

[Hu 1962] Ming K. Hu. *Visual Pattern Recognition by Moment Invariants*. IRE Transactions on Information Theory, vol. IT-8, pages 179–187, February 1962. 11

[Huang 1998] Jing Huang, S Ravi Kumar, Mandar Mitra and Wei jing Zhu. *Spatial Color Indexing and Applications*, 1998. 12

[Huang 2004] F. J. Huang and Y. LeCun. *Loss Functions for Discriminative Training of Energy-Based Models.*, 2004. 44

[Huijsmans 2005] Dionysius P. Huijsmans and Nicu Sebe. *How to Complete Performance Graphs in Content-Based Image Retrieval: Add Generality and Normalize Scope*. IEEE Trans. Pattern Anal. Mach. Intell., vol. 27, pages 245–251, February 2005. 125

[Igual 2007] L. Igual, S. Seguí, J. Vitrià, F. Azpiroz, P. Radeva, Xvi Congreso, Argentino Bioingeniería, V Jornadas and Ingeniería Clínica. *Sparse*

Bayesian Feature Selection Applied to Intestinal Motility Analysis Abstract, 2007. 63

[Jain 1996] A. Jain and A. Vailaya. *Image retrieval using color and shape*, 1996. 12

[Jégou 2007] Hervé Jégou, Hedi Harzallah and Cordelia Schmid. *A contextual dissimilarity measure for accurate and efficient image search*. In Conference on Computer Vision & Pattern Recognition, jun 2007. 30, 38, 45

[Johnson 1997] Andrew Edie Johnson and Martial Hebert. *Recognizing Objects by Matching Oriented Points*. In Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition (CVPR '97), CVPR '97, pages 684–, Washington, DC, USA, 1997. IEEE Computer Society. 18

[Joly 2007] Alexis Joly. *New local descriptors based on dissociated dipoles*. In CIVR '07: Proceedings of the 6th ACM international conference on Image and video retrieval, pages 573–580, New York, NY, USA, 2007. ACM Press. 20, 101

[Joly 2008] Alexis Joly and Olivier Buisson. *A posteriori multi-probe locality sensitive hashing*. In Proceeding of the 16th ACM international conference on Multimedia, MM '08, pages 209–218, New York, NY, USA, 2008. ACM. 79, 121

[Joly 2009] A. Joly and O. Buisson. *Logo Retrieval with A Contrario Visual Query Expansion*. In Seventeen ACM international conference on Multimedia (MM '09), New York, NY, USA, 2009. ACM. 2, 23, 42

[JUR 2004] Scale-invariant shape features for recognition of object categories, volume 2, 2004. 14

[Kadir 2001] Timor Kadir and Michael Brady. *Saliency, Scale and Image Description*. Int. J. Comput. Vision, vol. 45, pages 83–105, November 2001. 16, 39

[Ke 2004a] Y. Ke and R. Sukthankar. *PCA-SIFT: A more distinctive representation for local image descriptors*, 2004. 19

[Ke 2004b] Yan Ke, Rahul Sukthankar, Larry Huston, Yan Ke and Rahul Sukthankar. *Efficient near-duplicate detection and sub-image retrieval.* In In ACM Multimedia, pages 869–876, 2004. 21, 23

[Kerr 2008] Dermot Kerr, Sonya Coleman and Bryan Scotney. *Comparing Cornerness Measures for Interest Point Detection.* In Proceedings of the 2008 International Machine Vision and Image Processing Conference, pages 105–110, Washington, DC, USA, 2008. IEEE Computer Society. 56

[Koenderink 1987] J J Koenderink and A J van Doom. *Representation of local geometry in the visual system.* Biol. Cybern., vol. 55, no. 6, pages 367–375, 1987. 20

[Kokkinos 2008] Iasonas Kokkinos and Alan L. Yuille. *Scale invariance without scale selection.* In CVPR. IEEE Computer Society, 2008. 40

[Kumar 2007] Ankita Kumar and Cristian Sminchisescu. *Support Kernel Machines for Object Recognition.* Computer Vision, IEEE International Conference on, vol. 0, pages 1–8, 2007. 44

[Laptev 2006] Ivan Laptev. *Improvements of object detection using boosted histograms.* In In Proc. BMVC, pages 949–958. BMVC, 2006. 31, 40, 43

[Law-To 2007] J. Law-To, L. Chen, A. Joly, I. Laptev, O. Buisson, V. Gouet-Brunet, N. Boujemaa and F. Stentiford. *Video Copy Detection: a Comparative study.* In CIVR '07: Proceedings of the 6th ACM international conference on Image and video retrieval, Amsterdam, The Netherlands, jul 2007. 122

[Lazebnik 2003] Svetlana Lazebnik, Cordelia Schmid and Jean Ponce. *Sparse Texture Representation Using Affine-Invariant Neighborhoods*, 2003. 18, 24

[Lazebnik 2004] Svetlana Lazebnik, Cordelia Schmid and Jean Ponce. *Semi-local Affine Parts for Object Recognition.* In British Machine Vision Conference, volume 2, pages 779–788, 2004. 13

[Lazebnik 2006] Svetlana Lazebnik, Cordelia Schmid and Jean Ponce. *Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural*

Scene Categories. In CVPR '06: Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pages 2169–2178, Washington, DC, USA, 2006. IEEE Computer Society. 2, 30, 33, 52

[Leung 2001] Thomas Leung and Jitendra Malik. *Representing and Recognizing the Visual Appearance of Materials using Three-dimensional Textons*. Int. J. Comput. Vision, vol. 43, pages 29–44, June 2001. 23, 24

[Li 1999] Chen Li, Edward Chang, Hector Garcia-Molina, James Ze Wang and Gio Wiederhold. *Clindex: Clustering for Similarity Queries in High-Dimensional Spaces*. Rapport technique, 1999. 46

[Li 2009] Wu-Jun Li and Dit-Yan Yeung. *Localized content-based image retrieval through evidence region identification*. Computer Vision and Pattern Recognition, IEEE Computer Society Conference on, vol. 0, pages 1666–1673, 2009. 27

[Li 2010] Zhenxiao Li and Liqing Zhang. *Affine Invariant Topic Model for Generic Object Recognition*. In ISNN (2), pages 152–161, 2010. 35, 38, 43

[Lindeberg 1998] Tony Lindeberg. *Feature Detection with Automatic Scale Selection*. Int. J. Comput. Vision, vol. 30, no. 2, pages 79–116, 1998. 15

[Ling 2005] Haibin Ling and David W. Jacobs. *Deformation Invariant Image Matching*. In Proceedings of the Tenth IEEE International Conference on Computer Vision - Volume 2, ICCV '05, pages 1466–1473, Washington, DC, USA, 2005. IEEE Computer Society. 18

[liu 2007] Local Ensemble Kernel Learning for Object Category Recognition, 2007. 34, 38, 40, 44, 56

[Loupias 2000] Etienne Loupias and Jean-Michel Jolion. *Indexation d'images : Aide au Télé-enseignement et Similarités Pré-attentives (Image indexing : Applications to Tele-teaching and Pre-attentive similarities)*. PhD thesis, Institut national des sciences appliquées de Lyon, November 2000. written in French. 14

[Lowe 1999] D. G. Lowe. *Object recognition from local scale-invariant features.* In Proc. of Int. Conf. on Computer Vision, pages 1150–1157, 1999. 13, 15

[Lowe 2003] David Lowe. *Distinctive image features from scale-invariant keypoints.* International Journal of Computer Vision., vol. 20, pages 91–110, 2003. 30

[Lowe 2004] David G. Lowe. *Distinctive Image Features from Scale-Invariant Keypoints.* International Journal of Computer Vision, vol. 60, no. 2, pages 91–110, November 2004. 2, 15, 18, 19, 21, 38, 42, 68, 101

[Loy 2003] Gareth Loy and Alexander Zelinsky. *Fast Radial Symmetry for Detecting Points of Interest.* IEEE Trans. Pattern Anal. Mach. Intell., vol. 25, pages 959–973, August 2003. 17

[Mairal 2010] Julien Mairal. *Sparse coding for machine learning, image processing and computer vision.* PhD thesis, Ecole normale supérieure de Cachan, November 2010. 59

[Manolopoulos 2003] Yannis Manolopoulos, Alexandros Nanopoulos, Apostolos N. Papadopoulos and Yannis Theodoridis. *R-Trees Have Grown Everywhere*, 2003. 46

[Maron 1998] Oded Maron and Aparna Lakshmi Ratan. *Multiple-Instance Learning for Natural Scene Classification.* In Proceedings of the Fifteenth International Conference on Machine Learning, ICML '98, pages 341–349, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc. 27

[Marszałek 2007] Marcin Marszałek and Cordelia Schmid. *Accurate Object Localization with Shape Masks.* In IEEE Conference on Computer Vision & Pattern Recognition, jun 2007. 31, 41, 44

[Matas 2002] J. Matas, O. Chum, U. Martin and T. Pajdla. *Robust wide baseline stereo from maximally stable extremal regions.* In Proceedings of British Machine Vision Conference, volume 1, pages 384–393, London, 2002. 16, 20

[Mease 2008] David Mease and Abraham Wyner. *Evidence Contrary to the Statistical View of Boosting: A Rejoinder to Responses.* Journal of Machine Learning Research 9, pages 195–201, 2008. 61

[Medioni 1986] G. Medioni and Y. Yasumoto. *Corner detection and curve representation using cubic B-splines.* IEEE International Conference on Robotics and Automation., vol. 3, pages 764–769, April 1986. 14

[Meinshausen 2009] Nicolai Meinshausen and Peter Buehlmann. *Stability Selection.* May 2009. 26

[Meir 2003] Ron Meir and Gunnar Rätsch. An introduction to boosting and leveraging, pages 118–183. Springer-Verlag New York, Inc., New York, NY, USA, 2003. 61

[Mikolajczyk 2003] K. Mikolajczyk, A. Zisserman and C. Schmid. *Shape Recognition with Edge-Based Features.* In British Machine Vision Conference, volume 2, pages 779–788, 2003. 14

[Mikolajczyk 2004] Krystian Mikolajczyk and Cordelia Schmid. *Scale & Affine Invariant Interest Point Detectors.* Int. J. Comput. Vision, vol. 60, no. 1, pages 63–86, 2004. 16, 20, 38

[Mikolajczyk 2005a] Krystian Mikolajczyk and Cordelia Schmid. *A Performance Evaluation of Local Descriptors*, 2005. 17, 19, 68

[Mikolajczyk 2005b] Krystian Mikolajczyk, Tinne Tuytelaars, Cordelia Schmid, Andrew Zisserman, J. Matas, F. Schaffalitzky, T. Kadir and L. Van Gool. *A comparison of affine region detectors.* International Journal of Computer Vision, vol. 65, no. 1/2, pages 43–72, 2005. 17, 21

[Mindru 2003] Florica Mindru, Tinne Tuytelaars, Luc Van Gool and Theo Moons. *Moment Invariants for Recognition under Changing Viewpoint and Illumination*, July 2003. 20

[Mokhtarian 1998] F. Mokhtarian and R. Suomela. *Robust Image Corner Détection Through Curvature Scale Space.* IEEE Transactions on Pattern Analysis and Machine Intelligence., vol. 20, no. 12, pages 1376–1381, 1998. 14

[Morel 2009] Jean-Michel Morel and Guoshen Yu. *ASIFT: A New Framework for Fully Affine Invariant Image Comparison.* SIAM J. Img. Sci., vol. 2, pages 438–469, April 2009. 15, 19, 21, 30, 101

[Mutter 2007] Stefan Mutter and Bernhard Pfahringer. *A Discriminative Approach to Structured Biological Data*, 2007. 26

[Newsam 2003] S. Newsam, S. Bhagavathy and B.S. Manjunath. *Object localization using texture motifs and Markov random fields*. International Conference on Image Processing, vol. 3, pages 1049–1052, 2003. 13

[Nistér 2008] David Nistér and Henrik Stewénius. Linear time maximally stable extremal regions, volume 5303, pages 183–196. Springer, 2008. 16

[Nowak 2006] Eric Nowak, Frédéric Jurie and Bill Triggs. *Sampling strategies for bag-of-features image classification*. In European Conference on Computer Vision. Springer, 2006. 32, 33, 40, 41, 52, 56

[Nowak 2007] Eric Nowak and Frédéric Jurie. *Learning visual similarity measures for comparing never seen objects*. In Proc. IEEE CVPR, 2007. 32, 41, 43, 45

[Obozinski 2010] Guillaume Obozinski, Ben Taskar and Michael I. Jordan. *Joint covariate selection and joint subspace selection for multiple classification problems*. Statistics and Computing, vol. 20, pages 231–252, April 2010. 63

[Opelt 2006] Andreas Opelt, Michael Fussenegger and Peter Auer. *Generic Object Recognition with Boosting*. IEEE Trans. Pattern Anal. Mach. Intell., vol. 28, no. 3, pages 416–431, 2006. Member-Pinz,, Axel. xv, 2, 34, 41, 61, 66, 71, 95, 103, 105, 107, 109, 119, 126

[Osadchy 2007] Margarita Osadchy, Yann Le Cun and Matthew L. Miller. *Synergistic Face Detection and Pose Estimation with Energy-Based Models*. J. Mach. Learn. Res., vol. 8, pages 1197–1215, May 2007. 31, 40, 44

[Osborne 2000] Michael R. Osborne, Brett Presnell and Berwin A. Turlach. *On the LASSO and Its Dual*. Journal of Computational and Graphical Statistics, vol. 9, no. 2, pages pp. 319–337, 2000. 63

[Pha 2007] Online Learning Asymmetric Boosted Classifiers for Object Detection, 2007. 43

[Philbin 2007] J. Philbin, O. Chum, M. Isard, J. Sivic and A. Zisserman. *Object Retrieval with Large Vocabularies and Fast Spatial Matching*. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2007. 2, 23, 33, 42

[Philbin 2008] James Philbin, Josef Sivic and Andrew Zisserman. *Geometric LDA: A generative model for particular object discovery.* In In Proceedings of the British Machine Vision Conference, 2008. 33, 35, 41, 43

[Pineda 2010] Gibran Fuentes Pineda, Hisashi Koga and Toshinori Watanabe. *Object Discovery by Clustering Correlated Visual Word Sets.* Pattern Recognition, International Conference on, vol. 0, pages 750–753, 2010. 2, 24

[Ponce 2006] J. Ponce, T. L. Berg, M. Everingham, D. A. Forsyth, M. Hebert, S. Lazebnik, M. Marszalek, C. Schmid, B. C. Russell, A. Torralba, C. K. I. Williams, J. Zhang and A. Zisserman. *Dataset issues in object recognition.* In Toward Category-Level Object Recognition, volume 4170 of LNCS, pages 29–48. Springer, 2006. 54

[Rätsch 2001] G. Rätsch, T. Onoda and K.-R. Müller. *Soft Margins for AdaBoost.* Mach. Learn., vol. 42, no. 3, pages 287–320, 2001. 61

[Raza 2010] S.H. Raza, R.M. Parry, Y. Sharma, Q. Chaudry, R.A. Moffitt, A.N. Young and M.D. Wang. *Automated classification of renal cell carcinoma subtypes using bag-of-features.* In Engineering in Medicine and Biology Society (EMBC), 2010 Annual International Conference of the IEEE, 31 2010. 52

[Rebai 2006] Ahmed Rebai, Alexis Joly and Nozha Boujemaa. *Constant tangential angle elected interest points.* In MIR '06: Proceedings of the 8th ACM international workshop on Multimedia information retrieval, pages 203–212, New York, NY, USA, 2006. ACM. 17

[Rebai 2007] Ahmed Rebai, Alexis Joly and Nozha Boujemaa. *Interpretability based interest points detection.* In CIVR '07: Proceedings of the 6th ACM international conference on Image and video retrieval, pages 33–40, New York, NY, USA, 2007. ACM. 17

[Roberts 1963] Lawrence G. Roberts. Machine perception of three-dimensional solids. Outstanding Dissertations in the Computer Sciences. Garland Publishing, New York, 1963. 11

[Rohr 1992] Karl Rohr. *Recognizing corners by fitting parametric models.* Int. J. Comput. Vision, vol. 9, no. 3, pages 213–230, 1992. 15

[Saffari 2010] Amir Saffari. *Multi-Class Semi-Supervised and Online Boosting*, 2010. 61

[Salgian 2006] A. Salgian. *Object Recognition Using Local Descriptors: A Comparison*. In George Bebis, Richard Boyle, Bahram Parvin, Darko Koracin, Paolo Remagnino, Ara Nefian, Gopi Meenakshisundaram, Valerio Pascucci, Jiri Zara, Jose Molineros, Holger Theisel and Thomas Malzbender, editeurs, Advances in Visual Computing, volume 4292 of *Lecture Notes in Computer Science*, pages 709–717. Springer Berlin / Heidelberg, 2006. 10.1007/11919629_71. 17, 38

[Schaffalitzky 2002] Frederik Schaffalitzky and Andrew Zisserman. *Multi-view Matching for Unordered Image Sets, or "How Do I Organize My Holiday Snaps?"*. In ECCV '02: Proceedings of the 7th European Conference on Computer Vision-Part I, pages 414–431, London, UK, 2002. Springer-Verlag. 20

[Schapire 1990] Robert E. Schapire. *The Strength of Weak Learnability*. Mach. Learn., vol. 5, no. 2, pages 197–227, 1990. 61

[Schmid 1996] Cordelia Schmid. *Appariement d'images par invariants locaux de niveaux de gris*. PhD thesis, Institut National Polytechnique de Grenoble, July 1996. written in French. 14

[Schmid 1997] C. Schmid and R. Mohr. *Local gray value invariants for image retrieval*. IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 19, no. 5, pages 530–535, 1997. 13

[Schmid 2000] Cordelia Schmid, Roger Mohr and Christian Bauckhage. *Evaluation of Interest Point Detectors*. International Journal of Computer Vision, vol. 37, no. 2, pages 151–172, 2000. 15

[Sebe 2000] N. Sebe, M. S. Lew, Q. Tian, T. S. Huang and E. Loupias. *Color Indexing Using Wavelet-Based Salient Points*. In Proceedings of the IEEE Workshop on Content-based Access of Image and Video Libraries (CBAIVL'00), CBAIVL '00, pages 15–, Washington, DC, USA, 2000. IEEE Computer Society. 16

[Sebe 2005] Nicu Sebe, Ira Cohen, Ashutosh Garg and Thomas S. Huang. Machine Learning in Computer Vision (Computational Imaging and Vision). Springer, 1st ed. 2005 édition, August 2005. 25

[Seinstra 2006] F. J. Seinstra and J. M. Geusebroek. *Color-Based Object Recognition on a Grid*. In ECCV Workshop on Computation Intensive Methods for Computer Vision, 2006. 34

[Shotton 2005] Jamie Shotton, Andrew Blake and Roberto Cipolla. *Contour-Based Learning for Object Detection*. In Proceedings of the Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1 - Volume 01, pages 503–510, Washington, DC, USA, 2005. IEEE Computer Society. 39, 42

[Singer 2000] Yoram Singer. *Leveraged Vector Machines*. In Advances in Neural Information Processing Systems 12, pages 610–616. MIT Press, 2000. 61

[Sivic 2003] Josef Sivic and Andrew Zisserman. *Video Google: A Text Retrieval Approach to Object Matching in Videos*. Computer Vision, IEEE International Conference on, vol. 2, pages 1470–1477 vol.2, April 2003. 2, 49

[Smith 1997] S.M. Smith and J.M. Brady. *SUSAN - A New Approach to Low Level Image Processing*. Int. Journal of Computer Vision, vol. 23, no. 1, pages 45–78, May 1997. 15

[Stentiford 2006] Fred Stentiford. *Attention-Based Vanishing point detection*. In In Proc. ICIP 2006, pages 8–11, 2006. 17

[Stöttinger 2007] J. Stöttinger, N. Sebe, T. Gevers and A. Hanbury. *Colour Interest Points for Image Retrieval*. In Computer Vision Winter Workshop, pages 83–91, 2007. 15

[Subramanya 2007] Amarnag Subramanya, Zhengyou Zhang, Arun C. Surendran, Patrick Nguyen, Mukund Narasimhan and Alex Acero. *A GENERATIVE-DISCRIMINATIVE FRAMEWORK USING ENSEMBLE METHODS FOR TEXT-DEPENDENT SPEAKER VERIFICATION*, 2007. 26

[Swain 1991] Michael J. Swain and Dana H. Ballard. *Color indexing*. International Journal of Computer Vision, vol. 7, pages 11–32, 1991. 12

[Tarr 1995] Michael J. Tarr and Heinrich H. Bülthoff. *Is Human Object Recognition Better Described By Geon-Structural-Descriptions Or By Multiple-Views?*, 1995. 3

[Tian 2001] Q. Tian, N. Sebe and M. S. Lew. *Image Retrieval Using Wavelet-Based Salient Points*, 2001. 16

[Tibshirani 1996] Robert Tibshirani. *Regression shrinkage and selection via the lasso*. J. Roy. Statist. Soc. Ser. B, vol. 58, no. 1, pages 267–288, 1996. 26, 62

[Tieu 2004] Kinh Tieu and Paul Viola. *Boosting Image Retrieval*. Int. J. Comput. Vision, vol. 56, no. 1-2, pages 17–36, 2004. 61

[Tirilly 2008] Pierre Tirilly, Vincent Claveau and Patrick Gros. *Language modeling for bag-of-visual words image categorization*. In Proceedings of the 2008 international conference on Content-based image and video retrieval, CIVR '08, pages 249–258, New York, NY, USA, 2008. ACM. 23, 52

[Tonnin 2004] F. Tonnin and P. Gros. *Interest point detection in wavelet and curvelet domains*. In F. Truchetet & O. Laligant, editeur, Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series, volume 5607 of *Presented at the Society of Photo-Optical Instrumentation Engineers (SPIE) Conference*, pages 92–102, November 2004. 16

[Torresani 2010] Lorenzo Torresani, Martin Szummer and Andrew Fitzgibbon. *Efficient Object Category Recognition using Classemes*. In European Conference on Computer Vision (ECCV), pages 776–789, September 2010. 34, 44

[Tuzel 2006] Oncel Tuzel, Fatih Porikli and Peter Meer. *Region Covariance: A Fast Descriptor for Detection And Classification*. In In Proc. 9th European Conf. on Computer Vision, pages 589–600, 2006. 42

[Tuzel 2008] Oncel Tuzel, Fatih Porikli and Peter Meer. *Pedestrian Detection via Classification on Riemannian Manifolds*. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 30, pages 1713–1727, 2008. 31, 40, 42, 43

[Ulusoy 2006] Ilkay Ulusoy and Christopher Bishop. *Comparison of Generative and Discriminative Techniques for Object Detection and Classification*. pages 173–195. 2006. 31, 38, 45

[van Gemert 2010] J. C. van Gemert, C. J. Veenman, A. W. M. Smeulders and J. M. Geusebroek. *Visual Word Ambiguity*. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 32, no. 7, pages 1271–1283, 2010. 44

[Varma 2002] Manik Varma and Andrew Zisserman. *Classifying Images of Materials: Achieving Viewpoint and Illumination Independence*. In Proceedings of the 7th European Conference on Computer Vision-Part III, ECCV '02, pages 255–271, London, UK, UK, 2002. Springer-Verlag. 24

[Vedaldi 2008] A. Vedaldi and B. Fulkerson. *VLFeat: An Open and Portable Library of Computer Vision Algorithms*. http://www.vlfeat.org/, 2008. 101

[Vedaldi 2009] Andrea Vedaldi, Varun Gulshan, Manik Varma and Andrew Zisserman. *Multiple Kernels for Object Detection*, 2009. 33, 44

[Vinh 2010] Nguyen Vinh and Michael Houle. *A Set Correlation Model for Partitional Clustering*. In Mohammed Zaki, Jeffrey Yu, B. Ravindran and Vikram Pudi, editeurs, Advances in Knowledge Discovery and Data Mining, volume 6118 of *Lecture Notes in Computer Science*, pages 4–15. Springer Berlin / Heidelberg, 2010. 10.1007/978-3-642-13657-3_4. 13

[Viola 2004] Paul Viola and Michael J. Jones. *Robust Real-Time Face Detection*. Int. J. Comput. Vision, vol. 57, pages 137–154, May 2004. 31, 40, 43

[Wang 2006] Gang Wang, Ye Zhang and Li Fei-Fei. *Using Dependent Regions for Object Categorization in a Generative Framework*. In CVPR '06: Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pages 1597–1604, Washington, DC, USA, 2006. IEEE Computer Society. 11

[Wang 2010] Jun Wang, Sanjiv Kumar and Shih-Fu Chang. *Semi-Supervised Hashing for Scalable Image Retrieval*. 2010. 46

[Weber 2000] M. Weber, M. Welling and P. Perona. *Towards Automatic Discovery of Object Categories*, 2000. 38

[Winder 2007] Simon A. J. Winder and Matthew Brown. *Learning local image descriptors*. In In CVPR, pages 1–8, 2007. 40, 68

[Winn 2005] J. Winn, A. Criminisi and T. Minka. *Object Categorization by Learned Universal Visual Dictionary*. In Proceedings of the Tenth IEEE International Conference on Computer Vision - Volume 2, ICCV '05, pages 1800–1807, Washington, DC, USA, 2005. IEEE Computer Society. 23, 24

[Wu 2009] Zhong Wu, Qifa Ke, Jian Sun and Heung yeung Shum. *A Multi-Sample, Multi-Tree Approach to Bag-of-Words Image Representation for Image Retrieval*, 2009. 2, 23

[Wyner 2003] Abraham Wyner. *On Boosting and the Exponential Loss*, 2003. 61

[Xiao 2010] Jianxiong Xiao, James Hays, Krista A. Ehinger, Aude Oliva and Antonio Torralba. *SUN Database: Large-scale Scene Recognition from Abbey to Zoo*. IEEE Conference on Computer Vision and Pattern Recognition, 2010. 30, 56

[Xu 2010] Huan Xu, Constantine Caramanis and Shie Mannor. *Robust regression and Lasso*. IEEE Trans. Inf. Theor., vol. 56, pages 3561–3574, July 2010. 26

[Yahiaoui 2006] Itheri Yahiaoui, Nicolas Hervé and Nozha Boujemaa. *Shape-Based Image Retrieval in Botanical Collections*. vol. 4261, 2006. 18

[Yang 2007] Jun Yang, Yu-Gang Jiang, Alexander G. Hauptmann and Chong-Wah Ngo. *Evaluating bag-of-visual-words representations in scene classification*. In Proceedings of the international workshop on Workshop on multimedia information retrieval, MIR '07, pages 197–206, New York, NY, USA, 2007. ACM. 23, 52

[Yang 2009] Jianchao Yang, Kai Yu, Yihong Gong and Thomas Huang. *Linear spatial pyramid matching using sparse coding for image classification*. In in IEEE Conference on Computer Vision and Pattern Recognition(CVPR, 2009. 33, 44

[Yu 2008] Jie Yu, Jaume Amores, Nicu Sebe, Petia Radeva and Qi Tian. *Distance Learning for Similarity Estimation*. IEEE Trans. Pattern Anal. Mach. Intell., vol. 30, pages 451–462, March 2008. 45

[Zha 2008] Joint multi-label multi-instance learning for image classification, 2008. 34

[Zhang 2004] Rui Zhang, Beng Chin Ooi and Kian-Lee Tan. *Making the Pyramid Technique Robust to Query Types and Workloads*. In Proceedings of the 20th International Conference on Data Engineering, ICDE '04, pages 313–, Washington, DC, USA, 2004. IEEE Computer Society. 46

[Zhang 2007] J. Zhang, S. Lazebnik and C. Schmid. *Local features and kernels for classification of texture and object categories: a comprehensive study*. International Journal of Computer Vision, vol. 73, page 2007, 2007. 21

[Zhao 2007] Peng Zhao and Bin Yu. *Stagewise Lasso*. J. Mach. Learn. Res., vol. 8, pages 2701–2726, 2007. 63, 64, 70

[Zhou 2006] Zhi H. Zhou and Min L. Zhang. *Multi-Instance Multi-Label Learning with Application to Scene Classification*. In Bernhard Schölkopf, John C. Platt and Thomas Hoffman, editeurs, NIPS, pages 1609–1616. MIT Press, 2006. 27

[Zhu 2005] Song-Chun Zhu, Cheng-En Guo, Yizhou Wang and Zijian Xu. *What are Textons?* Int. J. Comput. Vision, vol. 62, pages 121–143, April 2005. 13

[Zhu 2006] Qiang Zhu, Mei-Chen Yeh, Kwang-Ting Cheng and Shai Avidan. *Fast Human Detection Using a Cascade of Histograms of Oriented Gradients*. In Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 2, CVPR '06, pages 1491–1498, Washington, DC, USA, 2006. IEEE Computer Society. 43