SALIENCY-BASED REPRESENTATIONS AND MULTI-COMPONENT CLASSIFIERS FOR VISUAL SCENE RECOGNITION

Thèse n. 6424 2014
présentée le 26 Septembre 2014
à la Faculté des Sciences de Base
laboratoire SuperScience
programme doctoral en SuperScience
École Polytechnique Fédérale de Lausanne

pour l'obtention du grade de Docteur ès Sciences
par

Marco Fornoni

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

acceptée sur proposition du jury :

Prof. Colin Jones, président du jury
Prof. Hervé Bourlard, directeur de thèse
Prof. Barbara Caputo, co-directeur de thèse
Prof. Jean-Philippe Thiran, rapporteur
Prof. Vittorio Murino, rapporteur
Prof. Danijel Skočaj, rapporteur

Lausanne, EPFL, 2014

The journey of a thousand miles begins with one step.
— Tao Te Ching

To my family and my friends

Acknowledgements

Few things are as enjoyable as writing an acknowledgment section, as this is a moment in which one has the opportunity to express gratitude to all those who generously contributed to the successful completion of a big piece of work.

First of all, I am very grateful to my supervisor Prof. Barbara Caputo for giving me the opportunity to carry out my doctoral studies at Idiap and EPFL. Barbara, I am thankful for your steady support throughout my doctoral journey and for the scientific freedom and trust that you have always granted me. Your pragmatic and open-minded attitude and guidance were and are a source of inspiration for my personal and professional growth. A big thank goes also to Prof. Francesco Orabona, for his support and guidance with his admirable technical mastery during several stages of my scientific journey. Thank you Francesco.

I would like to take this occasion to thank my thesis director Prof. Hervé Bourlard, as well as Prof. Jean Philippe Thiran, Prof. Vittorio Murino, Prof. Danijel Skočaj and Prof. Colin Jones for accepting to be part of my examination jury and for kindly taking the time to read my thesis in their busy schedule. I am also grateful to the Idiap system group and the Idiap and EPFL secretariats for their indispensable help in infinitely many occasions.

Amongst the most-rewarding non-scientific privileges of a Ph.D. student at Idiap and EPFL is the opportunity to meet talented people from all over the world, driven here by a thirst for knowledge, a pronounced curiosity and a sharp analytical mind ; each of them carrying his background, homeland culture, colors and tastes. I would thus like to thank the many people I have met here and who have greatly enriched my days : thanks to Laurent for the many great moments in the wilds, the many dinners and the many musical sessions ; thanks to Deepu for all the nice moments together, for introducing and hosting me into his family and for showing me the magic of his homeland, Kerala ; thanks to Leo for just being the funny and generous mate he naturally is ; thanks to Jagan for all the inspiring discussions about life, God and philosophy ; thanks to Thomas and Vincent for introducing me to climbing and slacklining ; thanks to Anindya for the many stimulating discussions about almost everything ; thanks to Ilja for the many stimulating discussions about almost only machine learning, for his company in Chicago and, together with Novi, for proof-reading part of this thesis ; thanks to Arjan for his support and his friendship ; thanks to Nesli, Ufuk, Rémi, Gwénolé, Marc, Vicky, Kai, Manuel, Roy, Harsha, Paul, Alexandros, Alex, Samira, Mohammad, Majid, David, Raphael, Ramya, Lakshmi, Sriram, Kenneth, Gülcan, Rui, PE, Cijo, Pranay, Dayra, Petr, Cosmin,

Charles, Nikos, Nik, Nicolae, Hugo, Paco, Tatiana, Serena, Gigi, Roger, Dinesh, Phil and Hari for organizing and participating to many funny social events, filling Martigny with their cheerful spirit. Thanks also to Michel for struggling to teach us some French and for sharing with us some bits of the special Valaisan life style.

A very special thanks goes to Ivana. I really do not exaggerate if I say that without her I would not be writing these words now. Her presence, trust and support in the most difficult moments provided me the drive and lift to continue and succeed. As I used to say, I should have written an entire acknowledgement section just for her.

Finally, it is needless to say I am infinitely grateful to my parents and to my full family for everything they did for me.

M. F.

Abstract

Visual scene recognition deals with the problem of automatically recognizing the high-level semantic concept describing a given image as a whole, such as the environment in which the scene is occurring (e.g. a mountain), or the event that is taking place (e.g. a rock climbing event). Scene categories, especially those related to man-made places and events, present high degrees of intra-class variability and inter-class similarity, which in turn require robust and discriminative recognition systems. An additional requirement for potential applications, such as vision-based spatial reasoning for mobile robots, is efficiency of the classification procedure. The objective of this thesis is to address these challenges, by proposing suitable image representations and classification algorithms.

The first part of the thesis focuses on the representation task. We propose a bottom-up image descriptor capturing perceptually coherent structures independently of their position. In particular, our method separately pools features extracted from two perceptually different image regions : the most salient region and the remaining non-salient one. By complementing this *Saliency-driven Perceptual Pooling* (*SPP*) with an ad-hoc spatial pooling operation, we obtain compact and robust image representations, particularly suited for indoor and sports scenes.

The second part of the thesis is concerned with the classification step. We propose an efficient multi-component classification algorithm, named *Multiclass Latent Locally Linear SVM* (*ML3*), able to automatically learn a set of sub-categorical linear models for each class, in a principled latent SVM framework. By linearly combining the sub-categorical models with sample and class specific weights, ML3 is able to efficiently learn smooth non-linear decision boundaries, competitive with those obtained by Gaussian kernel SVMs. ML3 also shows very competitive trade-offs between training time and performance, while ensuring high efficiency of the prediction phase.

In the last part of the thesis, we use the ML3 algorithm to improve the efficiency and performance of a recently proposed image classification algorithm, named *NBNN*, designed to cope with classes with a large diversity. Specifically, we show how with a modification of the NBNN scoring function it is possible to use ML3 to learn a discriminative and compact set of prototypical local features for each class, and thus avoid the extensive Nearest Neighbor search used by NBNN. The resulting algorithm, named *NBNL*, greatly reduces the memory requirements and testing complexity of NBNN, while significantly improving its performance.

The approaches proposed in this thesis effectively exploit the spatial, salient and task-driven structures present in the images, producing compact representations and relatively efficient

classification procedures.The SPP representations provide competitive scene recognition performances when coupled with non-linear kernels, while the ML3 algorithm can be used to partially fill the gap between linear and non-linear kernels. Although the performance of NBNN-based methods on scene recognition tasks is still below the one obtained by traditional SVM-based approaches, the proposed NBNL algorithm reduces the performance gap, while significantly speeding up the testing phase. Experiments on three publicly available scene recognition datasets (MIT-Indoor-67, 15-Scenes and UIUC-Sports) show the value of the proposed approaches.

Key words : visual scene recognition, saliency maps, feature pooling, multi-component classification, multi-class classification, locally linear SVM, latent SVM, naive Bayes nearest neighbor

Résumé

La reconnaissance visuelle des scènes consiste à déterminer le concept sémantique de haut niveau qui décrit une image dans son ensemble, comme l'environnement dans lequel la scène se dresse (e.g. une montagne), ou l'événement qui s'y déroule (e.g. une activité d'escalade). Les différentes catégories de scènes, en particulier celles liées aux endroits et événements créés par l'homme, présentent une variabilité intra-classe et une similarité inter-classe très importante, ce qui nécessite des systèmes de reconnaissance robustes et discriminatifs. Un besoin supplémentaire pour de possibles applications, comme le raisonnement spatial en utilisant l'information visuelle pour des robots mobiles, est l'efficacité de la procédure de classification. L'objectif de cette thèse est de résoudre ces problèmes, en proposant des représentations d'images et des algorithmes de classification adaptés.

La première partie de cette thèse porte principalement sur l'étape de représentation. Nous proposons un descripteur d'image de bas en haut, qui capture les structures cohérentes sur le plan perceptif indépendamment de leur position dans l'image. En particulier, notre méthode met en commun les caractéristiques extraites de deux régions différentes sur le plan perceptif séparément : la région la plus saillante et l'autre région non saillante. En complétant cette mise en commun perceptive guidée par la saillance (SPP) avec une opération de mise en commun spatiale ad hoc, nous obtenons des représentations d'images compactes et robustes, particulièrement adaptées aux scènes d'intérieur et de sport.

La deuxième partie de cette thèse concerne l'étape de classification. Nous proposons un algorithme efficace de classification à plusieurs composants, dénommé SVM multi-classe latent localement linéaire (ML3), capable d'apprendre automatiquement un ensemble de modèles linéaires sous-catégoriques pour chaque classe, dans un cadre reposant sur un SVM latent. En combinant linéairement les modèles sous-catégoriques avec des pondérations liées aux échantillons et aux classes, ML3 est en mesure d'apprendre efficacement des frontières de décision lisses et non-linéaires, qui rivalisent avec celle obtenues par des SVMs à noyau gaussien. ML3 montre également un équilibre très intéressant entre durée d'apprentissage et performance, tout en assurant une bonne efficacité lors de l'étape de prédiction.

Dans la dernière partie de cette thèse, nous utilisons l'algorithme ML3 pour améliorer l'efficacité et la performance d'un algorithme de classification d'images récemment proposé, NBNN, qui a été conçu de façon à s'adapter à des classes avec une très grande variabilité. Plus particulièrement, nous montrons comment il s'avère possible, en modifiant la fonction de score du NBNN, d'utiliser ML3 pour apprendre un ensemble discriminatif et compact de caractéristiques locales prototypique pour chaque classe, et, ainsi, d'éviter de recourir à

la vaste recherche des plus proches voisins utilisée par NBNN. L'algorithme qui en résulte, dénommé NBNL, réduit grandement les besoins en mémoire et la complexité de l'évaluation par rapport au NBNN, tout en améliorant significativement les performances.
Les approches proposés dans cette thèse exploitent efficacement les structures spatiales, saillantes ainsi que celles destinées à des tâches spécifiques, présentes dans les images, générant ainsi des représentations compactes et des procédures de classification relativement efficaces. Les représentations SPP fournissent des performances compétitives en reconnaissance de scènes lorsqu'elles sont associées avec des noyaux non-linéaires, tandis que l'algorithme ML3 peut être utilisée pour combler partiellement l'écart entre noyaux linéaires et non-linéaires. Bien que les performances en reconnaissance de scènes des méthodes reposant sur NBNN soient encore en dessous de celles obtenues avec les approches classiques à base de SVM, l'algorithme NBNL proposé réduit cet écart de performance, tout en accélérant significativement l'étape de classification. Des expériences conduites sur trois bases de données de reconnaissance de scènes accessibles au public (MIT-Indoor-67, 15-Scenes et UIUC-Sports) révèlent l'utilité des méthodes proposées.

Mots clefs : reconnaissance visuelle des scènes, cartes de saillance, mise en commun de caractéristiques, classification à plusieurs composants, classification multi-classe, SVM localement linéaire, SVM latent, classification naïve bayésienne par plus proches voisins

Contents

List of Figures

List of Tables

Glossary

BoW	Bag of visual Words
CCCP	Constrained Concave Convex Procedure
ICA	Independent Component Analysis
LHS	Left-Hand Side
LLSVM	Locally Linear SVM
MAP	Maximum A-Posteriori
ML3	Multiclass Latent Locally Linear SVM
NN	Nearest Neighbor
NBNL	Naive Bayes Non-linear Learning
NBNN	Naive Bayes Nearest Neighbor
PCA	Principal Component Analysis
RHS	Right-Hand Side
SGD	Stochastic Gradient Descent
SIFT	Scale Invariant Feature Transform
SPM	Spatial Pyramid Matching
SPP	Saliency-driven Perceptual Pooling
SVM	Support Vector Machine

Notation

$\boldsymbol{A}_{ij}^{t}$	a matrix (indexed for some purpose)
$\boldsymbol{a}_{ij}$	a vector (indexed for some purpose)
$\left(a_{ij}\right)_k$	the k-th entry of the vector $\boldsymbol{a}_{ij}$
a	a scalar
$\mathscr{R}$	a set
$f(\cdot)$	a real-valued, vector-valued, or matrix-valued function
$\boldsymbol{A}^\top$	the transpose of the matrix $\boldsymbol{A}$
$\mathrm{Tr}\,(\boldsymbol{A})$	the trace of the matrix $\boldsymbol{A}$
$\boldsymbol{A}\cdot\boldsymbol{B}$	the Frobenius inner product between the matrices $\boldsymbol{A}$ and $\boldsymbol{B}$
$\|\boldsymbol{A}\|_F$	the Frobenius norm of the matrix $\boldsymbol{A}$
$\|\boldsymbol{a}\|_p$	the p-norm of the vector $\boldsymbol{a}$
$\boldsymbol{a}^+$	the elementwise maximum between the vector $\boldsymbol{a}$ and 0
$\boldsymbol{a}\geq\boldsymbol{b}$	a vector inequality which holds i.i.f. $a_i\geq b_i$, $\forall i$
$\boldsymbol{a}\geq b$	a vector inequality which holds i.i.f. $a_i\geq b$, $\forall i$
$\lvert a\rvert$	the absolute value of a
$\lvert a\rvert_+$	the maximum between a and 0
$\mathbf{1}\left(\mathrm{p}\right)$	the indicator function of the predicate p (an equation, or an inequality)

Except when explicit from the context, we also make use of the following naming conventions:

$\mathscr{X}\subseteq\mathbb{R}^d$	the input space
$\mathscr{Y}$	the output space
$(\boldsymbol{x},y)$	an (input,output) sample
d	the dimensionality of the input space
n	the number of training samples
c	the number of classes in a given classification problem
m	the number of components in a given model
r	the number of local descriptors in a given image
k	the number of visual words in a BoW model

1 Introduction

1.1 Motivation

With the advent and widespread commercialization of inexpensive digital cameras, large amounts of digital images are being generated every day. Computers have become the main mean of storage and fruition of images, and the necessity to efficiently group, categorize and search this vast quantity of digital images has become a critical matter. Hence, efforts in developing various low and high-level *computer vision* techniques, which may help in this direction come with no surprise. Techniques such as color analysis and face detection or recognition have already been deployed in digital photo organizers and digital cameras. Still, there is a growing need for methods able to reliably and efficiently provide additional high-level information about the captured images. One basic type of high-level information that can be used to facilitate the management of large databases of images is the one regarding the global environment in which each image is captured. For example, one may be interested in retrieving all the images in which a given person is appearing in a mountain scenario, or in an office environment. In order to provide such high-level annotations of the images, it is thus necessary to design computer vision algorithms able to efficiently recognize these concepts.

Parallel to the ubiquitous diffusion of digital cameras, recent years have also seen the breakthrough of mobile robotics into the consumer market. Domestic robots have become increasingly common and are now extensively used to perform simple tasks, such as vacuum cleaning, or cutting the grass in the garden. Major automobile manufacturers and technology companies have already announced short-term plans to commercialize vehicles making use of cameras, radar and other sensors to assist the driver, or even autonomously conduct the passengers. In order to simplify the communications between humans and these artificial agents, and to enable a high-level reasoning using abstract spatial concepts, the human representation of space should also be understood and reproduced by artificial agents. For example, a domestic robot may be asked to “clean the bathroom”, while a car may be asked to “stop at the gas station”, or at “the parking area”. Consequently, a robot’s definition of “bathroom”, or “parking area” should point to the same set of places that a human would recognize as such.

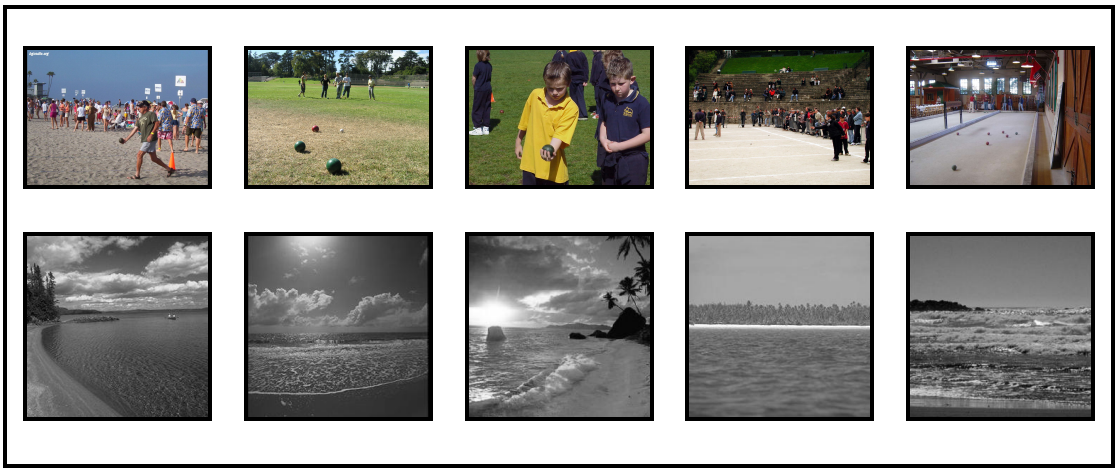

Figure 1.1 – Examples of scene categories exhibiting intra-class variability. Top: structural variability in images from the scene category “bocce”. Bottom: view-point variability in images from the scene category “coast”.

1.2 Statement of the problem and challenges

In computer vision, the task of automatically annotating a single image with the categorical label that best describes the scene as a whole is known as *scene recognition*. As opposed to objects, scenes are mainly characterized as places in which humans can move [Oliva and Torralba, 2001]. This definition can be extended to include events, such as sports activities [Li and Fei-Fei, 2007] (e.g. a “sailing” scene, or a “croquet” scene).

The most important challenges in scene recognition come from the complexity of the concepts to be recognized and the variability of the conditions in which the images are captured. Scenes from the same category may often look different, while scenes from different categories may look similar. We refer to the variability in the appearance of images within a single scene category as *intra-class variability*, while the similarity of images belonging to different categories is referred to as *inter-class similarity*. Besides intra-class variability and inter-class similarity, an additional challenge for scene recognition, coming from the application domain, is due to *efficiency requirements*.

Intra-class variability is mainly due to two factors:

1. **Structural complexity and variability.** Scenes are complex high-level concepts, in turn composed of several complex parts, whose number, types and configurations cannot be fixed *a priori*. Take for example the category “office”. An office would likely contain desks and chairs, but their number and spatial arrangement may vary from instance to instance. Moreover, additional parts, such as computers, printers, telephones, lamps, shelves, books, white-boards, plants and windows may or may not be present.

2. **View-point variability.** Scenes can look very different from different points of view. This is especially true for indoor scenes, where the distance between the subject of the picture

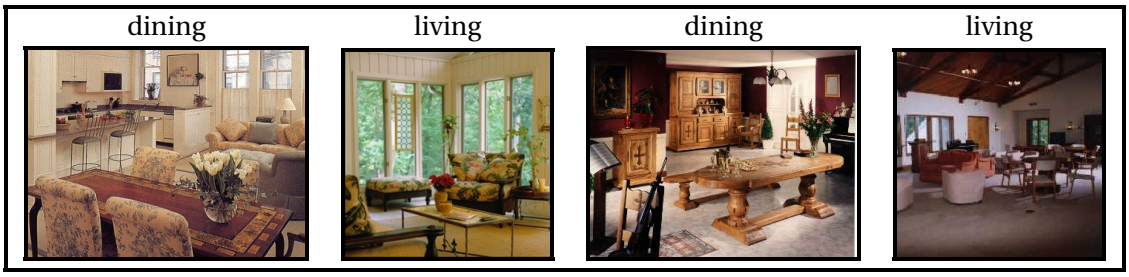

Figure 1.2 – Examples of scenes exhibiting high degrees of inter-class similarity. Images annotated with "living" belong to the scene category "living room", while scenes annotated with "dining" belong to the scene category "dining room".

and the observer is often very low. For example, a bedroom may be captured from a viewpoint in which the full bed is visible, or from the opposite viewpoint, in which only other objects such as a television, wardrobes, mirrors and desks are fully visible.

Due to the high levels of visual variability, images belonging to a certain scene category may cluster into so called visual *sub-categories*. A visual sub-category consists of images from the same scene category having a common perceptual appearance.

It is to be noted that the structural variability of scene categories is higher than for object categories [Ehinger, 2013]. Take, for example, object categories such as car, dog, washing machine, or mobile phone. Although there might be sub-categories (e.g. "smart-phone" vs "old-generation mobile phone") and the final appearance may be very different, it is not difficult to think of a prototypical set of parts with a prototypical spatial configuration for each object category, or sub-category (e.g. two wheels in front, two wheels on the back, a body above the wheels and several windows above the body, for the category "car"). The same reasoning cannot be easily reproduced for many scene categories, as the "office" category described above, other indoor categories, or even for sport scenes. A match of "bocce", for example, can be played indoor, as well as outdoor, on a proper framed field, on the grass, or even on the beach. Moreover, the exact number of players, bowls and their relative positions may vary continuously from image to image (see Figure 1.1).

In Figure 1.1 we report some examples of images from the scene categories "bocce" and "coast", illustrating the effects of structural and view-point variability, respectively. As it can be seen, images from the category "bocce" do present a considerable degree of structural variability. It is indeed difficult to predict which parts may be expected in the images and in which number. Coastal scenes present a much lower degree of structural variability (parts such as water, sky and land are always present, approximatively in a fixed number). On the other hand they still present a noticeable degree of intra-class variability, due to the variable view-point of the observer.

In addition to high levels of intra-class variability, scene categories are also characterized by high degrees of inter-class similarity. As an example, consider the categories: "sea coast",

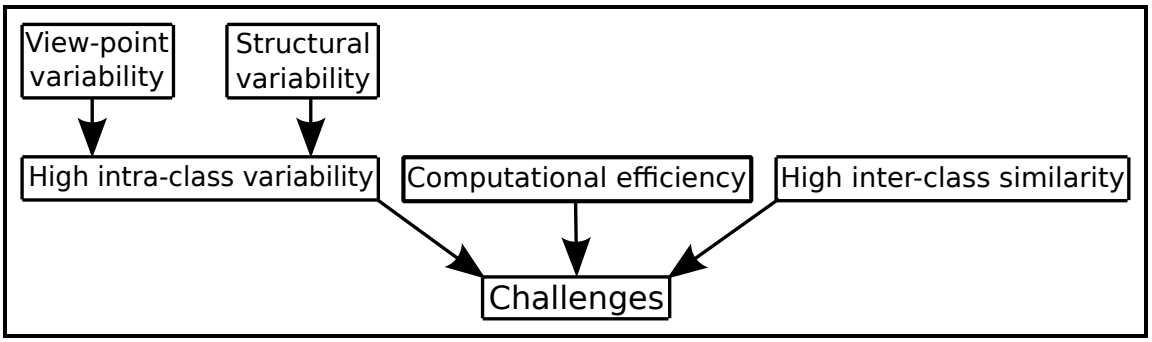

Figure 1.3 – Challenges of the scene recognition problem.

"living room", "dining room" and "lake shore". While it may be relatively easy to distinguish a sea coast from a living room, or a lake shore from a dining room, it may be more difficult to discriminate between a living room and a dining room. These scene categories, indeed, present very similar visual appearances, sharing also a similar distribution of parts (e.g. chairs, sideboards, televisions and sofas). Similarly, a sea coast may not be easily discriminated from a lake-shore. In Figure 1.2 we illustrate this problem for the "dining room" and "living room" categories. As it can be seen there is an evident overlap in the parts (e.g. objects) appearing in images from the two classes. This results in a discrimination problem that may be challenging even for humans.

Finally, for a scene recognition algorithm to have any practical utility it has to be computationally efficient. Indeed, in order to process and annotate large amounts of pictures - as in the digital photo management scenario described above - efficiency of the recognition phase becomes crucial. Computational efficiency becomes even more crucial if we consider the mobile robot scenario. Indeed, mobile robots need to be able to process images at a rate fast enough to ensure smoothness of movement and responsiveness. Furthermore, as opposed to standard computers, mobile robots may also be constrained by reduced computational resources.

The combination of high intra-class variability, high inter-class similarity and computational efficiency requirements makes scene recognition a very challenging problem. A compact visualization of these three combined challenges is provided in Figure 1.3.

1.3 Objectives and approach

The problem of recognizing the environment in which a given scene is taking place can be viewed as an image classification task: given a set of possible scene labels (e.g. sea coast, living room, dining room and lake shore), the image has to be assigned to the one that best represents it. A modern approach to tackle such problems is to make use of *machine learning*, a branch of computer science and artificial intelligence studying systems able to learn from data. Given a dataset of images annotated with the desired (*groundtruth*) label, the recognition

system passes through a *training* stage and an *evaluation*, or *testing* stage. During the training stage, the system makes use of a portion of the dataset, named *training set*, to learn a mapping from the images to the labels. We refer to the learned mapping as the *model*. In the testing stage, the performance of the learned model is evaluated on the remaining portion of the dataset, named *testing set*, or *test set*.

A typical image classification pipeline can be decomposed into two main blocks:

1. **Image representation.** The purpose of this component is to pre-process the images and output image signatures preserving information that may be important for the classification task, while filtering out the rest.
2. **Classification algorithm.** The purpose of this component is to provide a model able to correctly classify the signatures computed from the images in the training set, while performing similarly well on other unseen images, as the ones contained in the test set. An image is said to be correctly classified if it is assigned the same label as the groundtruth.

The main goal of this thesis is to develop image representations and classification algorithms able to efficiently recognize scene categories. As previously discussed, scene categories are characterized by high levels of complexity, intra-class variability (due to both view-point and structural variability) and inter-class similarity. A scene recognition algorithm should thus be able to produce models complex and invariant enough to cope with such levels of variability. The categorical models should also be discriminative enough to allow a fine discrimination between very similar scene categories. Finally, in order to have any practical utility, the models should also be efficient to train and, even more importantly, efficient to evaluate. Consequently, the research questions that we are aiming to answer in this thesis are the following:

1. *Is it possible to design compact and discriminative image representations able to effectively describe images presenting very high levels of structural and view-point variability?*
2. *Is it possible to design efficient and discriminative classification algorithms able to cope with the high level of complexity and variability of scene categories?*

In the attempt to positively answer these questions, throughout this thesis we adopt the following design choices:

1. **Low-to-mid level image representations.** As discussed above, scene categories are structured and complex entities. It would thus be highly desirable to employ representations making use of high-level concepts, such as the statistics of object occurrences in the scenes. As shown by Vogel and Schiele [2004], using such information alone would be sufficient to solve small scene recognition problems. Unfortunately, as pointed out by Torresani et al. [2010], state of the art object detectors are still unreliable, essentially behaving as texture and shape recognizers. Moreover, although major advances have been achieved in recent years [Dubout and Fleuret, 2012], object detectors are still relatively expensive to evaluate. This is especially true if the detection process has to be repeated for a large number of objects, in the order of hundreds, or thousands, as required by current state of the art high-level representations [Torresani et al., 2010; Li et al., 2010].

For these reasons, we opt to directly make use of more efficient and well-understood low and mid-level representations, strictly related to the visual appearance of the images. We thus leave the task of modeling higher-level structures to the scene classification algorithm.

2. **Multi-component categorical models.** Given the high structural complexity and variability of scene categories, and considering also the relative simplicity of low and mid-level representations, it becomes necessary to make use of complex categorical models, able to recognize samples belonging to a high number of visual sub-categories. For example, it would be necessary to learn different models for the different views of the coast scene category. This can be naturally accomplished by using models in which each category is described by a set of *components*, each one specialized to a set of samples sharing similar perceptual properties. Multi-component models [Dollár et al., 2008; Felzenszwalb et al., 2010; Gu et al., 2012] naturally allow to represent complex categories by means of a set of simple and specialized sub-categorical components.

3. **Supervised discriminative learning.** As discussed before, besides the high levels of structural complexity and variability, another major problem of scene categorization is that of high inter-class similarity. In order to cope with this problem we chose to make use of discriminative learning algorithms, directly trained to minimize the number of miss-classification errors. Moreover, since sub-categorical annotations of the images are not available (e.g. annotation of the view-points, or of the structural type of the scene), we decide to adopt a weakly supervised approach, in which the component(s) associated to each image have to be automatically inferred.

In the following Section we provide a brief description of the contributions made in this thesis, instantiating them within the structure of the thesis itself.

1.4 Contributions and structure

As motivated and discussed in the previous Sections, this thesis aims at designing and evaluating compact image representations and efficient classification algorithms suitable for scene recognition problems. A brief description of each Chapter and the related contribution is as follows:

- **Chapter 2: Related works.** In this Chapter we introduce and discuss a prototypical scene recognition pipeline, providing a review of the works related to each of its constituent blocks.

- **Chapter 3: Spatial and saliency-driven image representations.** In this part of the thesis we aim at designing image representations able to cope with the high intra-class variability and computational efficiency requirements of scene recognition problems. In contrast with traditional spatial representations [Lazebnik et al., 2006], we aim at designing representations able to capture perceptually coherent structures, independently from their positions in the image. To this end, we propose to separately pool image features extracted from two

perceptually different regions of the image: the most salient (and usually more complex) region and the remaining non-salient one. By complementing this saliency-driven pooling, named *Saliency-driven Perceptual Pooling* (*SPP*), with a simple spatial pooling operation we obtain compact and robust image representations. The proposed representations are shown to be particularly suited for indoor and sports scenes, outperforming more complex spatial representations on several scene recognition tasks. From a computational point of view, the main limitation of the scene recognition pipeline proposed in this Chapter is the usage of exponential χ^2 kernel classifiers [Fowlkes et al., 2004], which are expensive to train and to evaluate.

- **Chapter 4: The ML3 classification algorithm.** In this Chapter we aim at addressing the high intra-class variabilities, the high inter-class similarity and the computational efficiency requirements of scene recognition problems, by designing a new classification algorithm. In order to make the classification algorithm efficient to train and to evaluate, we opt to avoid the use of kernels. Instead, we propose a multi-component algorithm, named *Multi-class Latent Locally Linear SVM* (*ML3*), able to automatically learn a set of sub-categorical linear models for each class, in a principled *latent SVM* framework [Yu and Joachims, 2009]. By linearly combining the components of the model with sample and class specific weights, ML3 proves to be able to efficiently learn smooth non-linear decision boundaries, competitive with those obtained by Gaussian kernel classifiers [Shawe-Taylor and Cristianini, 2004]. Compared to other state of the art multi-component algorithms, the proposed algorithm is also shown to provide very competitive trade-offs between training time and performance. We apply ML3 to the SPP image representations proposed in Chapter 3. The scene recognition performance obtained in this way is still lower than the one obtained by the exponential χ^2 kernel classifiers used in Chapter 3. Nonetheless, it is close to the performance obtained by a Gaussian kernel classifier, and it is achieved at a fraction of the computational resources required by the latter (in terms of training time, testing time and memory footprint).

- **Chapter 5: The NBNL classification algorithm.** In the approach discussed in Chapter 4, the components of the ML3 model are learned and evaluated on the full images. In this Chapter we abandon this approach in favor of a patch-based approach, in which different parts of an image can be assigned to different components. We follow the *Naive Bayes Nearest Neighbor* (*NBNN*) framework [Boiman et al., 2008], a recently proposed image classification algorithm designed to cope with classes with a very large diversity. Within this framework, we show how with a modification of the NBNN scoring function it is possible to use ML3 to learn a discriminative and very compact set of prototypical local features for each class, thus avoiding the extensive Nearest Neighbor search used by NBNN. The resulting algorithm, named *NBNL*, preserves the robustness of NBNN, while greatly reducing its memory requirements and testing complexity, and significantly improving its performance. On small scene recognition problems, the NBNL algorithm combined with a SPP pooling approach is shown to provide recognition performances on par with the most competitive kernel classifiers considered in Chapter 3.

- **Chapter 6: Conclusions.** This Chapter summarizes the achievements of this thesis, draws the conclusions and outlines some potential direction for further research.

1.5 References

The contributions discussed in Chapter 3, Chapter 4 and Chapter 5 of this thesis are based on the preliminary works presented in the following peer-reviewed publications:

Marco Fornoni and Barbara Caputo. Indoor scene recognition using task and saliency-driven feature pooling. In Proc. of British Machine Vision Conference, BMVC, pages 1–12, 2012

Marco Fornoni, Barbara Caputo, and Francesco Orabona. Multiclass latent locally linear support vector machines. In Cheng Soon Ong and Tu-Bao Ho, editors, JMLR W&CP, Volume 29: ACML, pages 229–244, 2013

Marco Fornoni and Barbara Caputo. Scene recognition with naive bayes non-linear learning. In Proc. of the 22nd International Conference on Pattern Recognition (ICPR). IEEE, August 2014

2 Related Works

Visual scene recognition is a topic that has been extensively studied from different points of view. From a biological perspective Wolfe [1998] proposed a model of the human visual memory based on the concept of gist. With a series of thought experiments and links with relevant literature, he proposed that what humans capture about a scene is composed by two main components: 1) information about basic image features, the existence of surfaces, shapes and their spatial configuration; 2) a list of recognized objects (and their spatial configuration), selected through an attention mechanism. From a computational point of view, a model for scene recognition is usually built upon a set of *visual primitives*, such as image patches, or regions, which constitute the basic building blocks for constructing more complex representations of the scene. Each of these visual primitives can consider information at several spatial resolutions, ranging from a single pixel to the full scene, and the set of visual primitives used by a scene recognition algorithm determines the spatial resolution of the visual information accessible to the algorithm. Each primitive can be described by three types of *descriptors*:

1. *Low-level*, the descriptor of a visual primitive is constructed by directly using the low-level features extracted from the considered area; this approach assumes that the low-level features are describing aspects of an image that could be directly linked to its semantics.
2. *Mid-level*, after the low-level feature extraction, an intermediate encoding is computed to represent each feature with respect to a learned set of prototypical low-level features.
3. *High-level*, the low-level features are used to evaluate computational models of human-understandable concepts (e.g. models of objects, or scenes subparts) and produce a descriptor endowed with semantics.

Note that the proposed descriptors taxonomy only aims to encompass all the methods relevant to this thesis. It is neither intended to be general, nor to match other paradigms adopted by the computer vision community [Marr, 1982; Wilson and Keil, 2001].

Different methods have also been considered to aggregate the set of descriptors computed from the visual primitives, and compose the final *image representation* (or *image signature*). For example, the representation of the image can simply be defined as the collection of disjoint

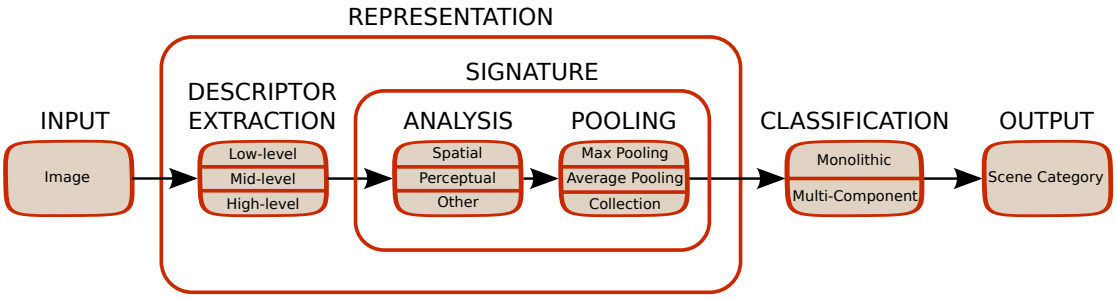

Figure 2.1 – The general scene recognition pipeline considered in this thesis.

descriptors computed from the visual primitives, or it can be constructed in a statistical way, e.g. by pooling the descriptors over the full image to produce a synthetic signature. Moreover, in order to exploit the additional structure present in the considered scenes, the final representation is often constructed by performing an analysis of the spatial position of the descriptors, or of some perceptual information such as their saliency. Finally, leveraging the designed image representations, a computational model for a given scene concept (e.g. a "mountain scenery") is constructed using one of several possible types of classifiers.

A visualization of the scene recognition pipeline considered throughout this thesis is illustrated in Figure 2.1. The proposed pipeline is only intended to be illustrative: not all the building blocks reported have to necessarily be present in a scene recognition system, while some other might be added, or might be merged together. Still, the pipeline is general enough to represent a large set of approaches that are relevant to this thesis and that will be discussed in this Chapter.

Throughout the thesis, the adjectives characterizing some parts of the representation block of the pipeline may be used to describe the overall image representation as well. For example, an image representation making use of high-level descriptors may be referred to as a high-level representation, while an image representation making use of a spatial, or a saliency analysis may be referred as a spatial representation, or a saliency-driven representation.

Before proceeding with the discussion about the different blocks of the scene recognition pipeline illustrated in Figure 2.1, it is important to dwell on how the scene recognition problem is empirically evaluated in the computer vision community. For this purpose in Section 2.1 we describe the standard datasets used for benchmarking scene recognition methods, and their corresponding evaluation protocols. The blocks of the pipeline in Figure 2.1 are extensively described in the subsequent Sections. Specifically, the descriptor extraction block, the signature block and the classification block are covered in Sections 2.2, 2.3 and 2.4, respectively.

Table 2.1 – List of scene recognition publications and datasets used. From the list we exclude the publications in which a new dataset was proposed.

Work	MIT-Indoor-67	15-Scenes	Sports	Other
Li et al. [2010]	✓	✓	✓	LabelMe-9
Çakir et al. [2011]	✓	✓		
Pandey and Lazebnik [2011]	✓			
Wu and Rehg [2011]	✓	✓	✓	
Fornoni and Caputo [2012]	✓	✓	✓	
Kwitt et al. [2012]	✓	✓	✓	LabelMe-9, SUN
Jiang et al. [2012]		✓	✓	21-Land-Use
Parizi et al. [2012]	✓			
Sadeghi and Tappen [2012]	✓	✓	✓	
Zheng et al. [2012]	✓	✓	✓	
Juneja et al. [2013]	✓			
Vitaladevuni et al. [2013]	✓	✓	✓	
Fornoni and Caputo [2014]	✓	✓	✓	
Xie et al. [2014]	✓			SUN

2.1 Datasets

Throughout the years, three main datasets have been established as standard benchmarks for scene recognition algorithms: the MIT-Indoor-67 [Quattoni and Torralba, 2009], the 15-Scenes [Lazebnik et al., 2006] and the UIUC-Sports [Li and Fei-Fei, 2007] datasets. In Table 2.1 we report a list of recently published scene recognition approaches, with the corresponding list of scene recognition datasets used. As it is possible to see the three above mentioned datasets are used in most of the recent scene recognition publications. Accordingly, these datasets are also the main benchmarks considered in this thesis. Note that, since we focus on classification of single images, we do not consider datasets for evaluating approaches addressing the problem of classifying scenes in video sequences (e.g. [Pronobis and Caputo, 2005; Luo et al., 2006; Pronobis and Caputo, 2009; Wu et al., 2009; Pronobis et al., 2010]).

In the following, for each of the three dataset considered throughout this thesis (MIT-Indoor-67, 15-Scenes and UIUC-Sports) we provide a description of the collection procedure, a description and a visualization of the images belonging to the dataset, and a description of the benchmarking procedure. In addition we provide a synthetic description of the other scene recognition datasets appearing in Table 2.1 (SUN [Xiao et al., 2010], LabelMe-9 [Li et al., 2010] and 21-Land-Use [Yang and Newsam, 2010]). For additional details we refer the interested reader to the appropriate publication.

MIT-Indoor-67

Since its introduction, the MIT-Indoor-67 dataset [Quattoni and Torralba, 2009] has become one of the most important and most challenging benchmarks for scene recognition algorithms.

Figure 2.2 – Example images from the 67 classes of the MIT-Indoor-67 dataset, organized by scene group. (Adapted from Quattoni and Torralba [2009])

According to Table 2.1, it is now the most used dataset for this class of problems. It consists of images of indoor scenes captured in unconstrained and cluttered conditions and collected using online image search engines, online photo sharing sites and the LabelMe dataset [Russell et al., 2008]. It contains 15,620 images belonging to 67 different categories, at a minimum resolution of 200 pixels in the smallest axis and with a minimum of 100 images per category. The 67 scene categories are grouped into 5 big scene groups: Store, Home, Leisure, Public place and Working Place.

It is worth noting that in indoor environments the location of meaningful regions and objects varies drastically within each category, while the close-up distance between the camera and the subject makes the variations due to view-point changes even more severe. This results in a very high degree of intra-class visual variability. Moreover, many of the classes (e.g. classes belonging to the same scene group) present a high degree of visual inter-class similarity. Images sampled from the categories in the five scene groups are visualized in Figure 2.2.

The standard benchmarking procedure for this dataset consists of randomly selecting 100 images per category and split them into 80 images for training and 20 for testing. The scene

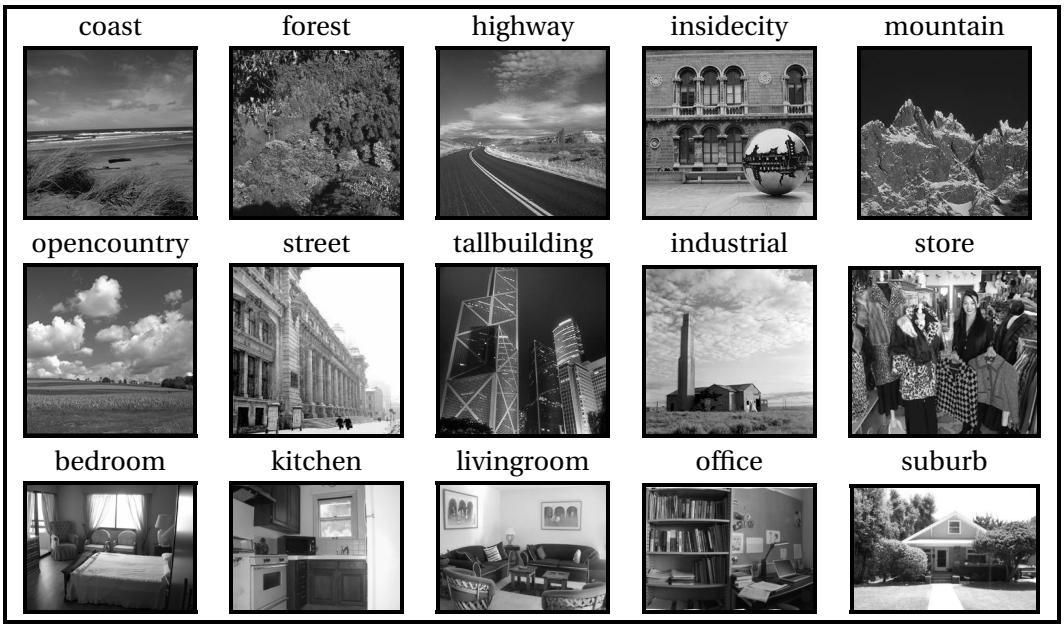

Figure 2.3 – Example of images from the classes of the 15-Scenes dataset.

recognition performance is measured by the multiclass accuracy, defined as the average of the diagonal of the confusion matrix. Using this benchmarking procedure, the scene recognition accuracy reported at the moment this dataset was published is 26%.

15-Scenes

The 15-Scenes dataset [Lazebnik et al., 2006] is a well established scene recognition benchmark, containing images of both outdoor and indoor scene environments. The collection was gradually built over the years: the initial 8 outdoor classes were collected by Oliva and Torralba [2001]; four additional indoor categories and one additional outdoor category were added by Fei-Fei and Perona [2005]; finally, two additional categories (one indoor and one outdoor) were introduced by Lazebnik et al. [2006].

In its final version, the 15-Scenes dataset contains 4485 low-resolution and gray-valued images, with 210 to 410 images per class. The 15 scene categories are: bedroom, coast, forest, highway, industrial, insidecity, kitchen, livingroom, mountain, office, opencountry, store, street, suburb and tallbuilding.

In Figure 2.3 we report one image example for each category. As it is possible to see, the inter-class similarities are lower for this dataset, with the largest visual similarities occurring amongst indoor classes.

For this dataset the standard benchmarking protocol consists in randomly selecting 100 training images per class and using the remaining ones for evaluation. The scene recognition

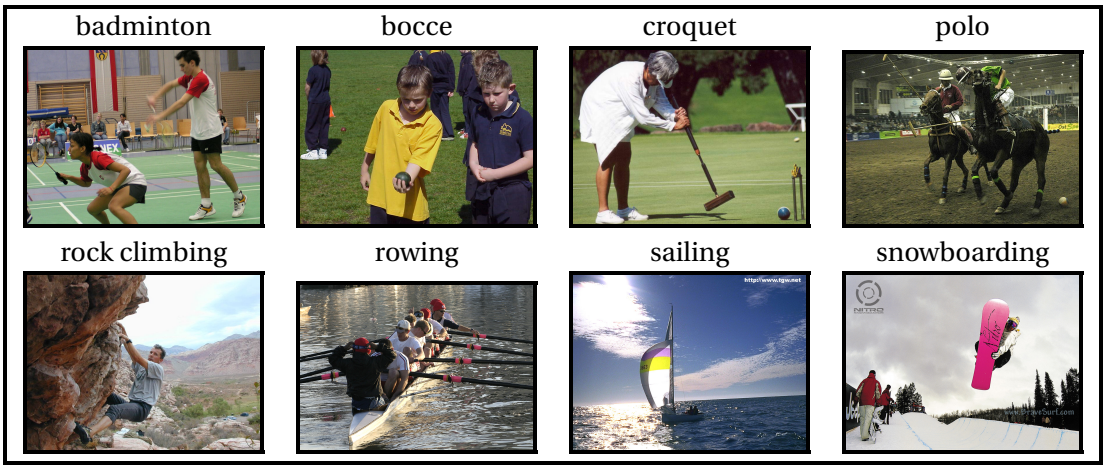

Figure 2.4 – Example of images from the classes of the UIUC-Sports dataset.

performance is measured by the multiclass accuracy, defined as the average of the diagonal of the confusion matrix. Using this benchmarking procedure, the scene recognition accuracy reported at the moment this dataset was published is 81.4%.

UIUC-Sports

The UIUC-Sports dataset [Li and Fei-Fei, 2007] is a collection of images of sports scenes. According to the authors, although sport categories represent events and not just places (as the categories in the 15-Scenes, or the MIT-Indoor-67 datasets), sport recognition can be approximated and viewed as a scene recognition problem. As mentioned in Section 1.3, the main challenges of this problem lie in the high levels of structural variability, due to clutter and variability of the environment in which the events are taking place, and to the wide variety of subjects and poses in each category. Some sports categories, like croquet and bocce, also present high levels of visual similarity.

This dataset has been used in the large majority of works addressing scene recognition problems (see Table 2.1) and it contains images from 8 sport categories: badminton, bocce, croquet, polo, rock climbing, rowing, sailing and snowboarding. The number of images per category varies between 137 and 250. A visualization of images sampled from each category is reported in Figure 2.4.

The benchmarking protocol for this dataset consists in selecting 70 images per class for the training set and 60 for the test set. The scene recognition performance is measured by the multiclass accuracy, defined as the average of the diagonal of the confusion matrix. Using this benchmarking procedure, the scene recognition accuracy reported at the moment this dataset was published is 74.4%.

Other datasets.

For completeness, we report here a short description of other datasets that have been occasionally used to evaluate the performance of scene recognition approaches:

– **SUN** [Xiao et al., 2010]. The SUN (Scene UNderstanding) dataset is a large scale dataset containing images from 397 scene categories, selected using the WordNet ontology [Fellbaum, 1998]. Each scene category contains at least 100 color images, retrieved using search engines. A subset of the dataset is annotated with objects.
– **LabelMe-9** [Li et al., 2010]. The LabelMe-9 dataset is a subset of the LabelMe dataset [Russell et al., 2008] containing images from 9 scenes categories: beach, mountain, bathroom, church, garage, office, sail, street, forest. Each class contains 100 images, split into 50 for training and 50 for testing.
– **21-Land-Use** [Yang and Newsam, 2010]. The 21-Land-Use dataset contains images of aerial orthoimagery downloaded from the United States Geological Survey (USGS) National Map. There are 21 classes, each represented by 100 images.

In this Section we have provided a review of datasets, benchmarking procedures and evaluation metrics used by the scene recognition community. In the next Section we begin the discussion of the blocks composing the scene recognition pipeline introduced at the beginning of this Chapter.

2.2 Descriptor Extraction

Over time, a large number of visual primitives and descriptors have been considered for scene recognition tasks. Without aiming to be exhaustive, in this Section we will discuss the most important ones, ordering them according to the taxonomy introduced before.

2.2.1 Low-level descriptors

The descriptors belonging to this category simply consist of low-level features extracted from the visual primitives. One of the first descriptors specifically designed for scene recognition is the *Spatial Envelope*, also known as *GIST* [Oliva and Torralba, 2001]. The visual primitive considered by this descriptor is tipically either a large image patch or the full image. In this work the authors analyze the global appearance properties (not related to objects) used by humans in order to categorize outdoor scenes. By performing an experiment with seventeen human observers they argue that the five most important global properties used by humans to categorize outdoor scenes are: *Degree of Naturalness, Degree of Openness, Degree of Roughness, Degree of Expansion, Degree of Ruggedness*. They then propose a computational model for each of these properties and combine them to obtain a final image signature. Using this descriptor the authors obtained good results in classifying low-resolution outdoor scenes into categories like: mountains, seaside, forest, etc. All the same, the approach was later found to be unsuitable for indoor scene categories [Quattoni and Torralba, 2009].

Another example of a low-level descriptor used to address scene recognition problems is the one proposed by Linde and Lindeberg [2004]. In this work the authors suggest to use multi-dimensional histograms of low-level features (such as normalized gradient magnitude and RGB chromatic cues at multiple scales) as a robust way to describe images. The visual primitive considered by this descriptor is the full image. Exploiting the fact that multi-dimensional histograms are mostly zero except for a small portion of the cells, they design a sparse and sorted representation enabling to accumulate histograms with a number of cells of the order of $45^{14} \approx 10^{23}$ (14-D histogram with 45 quantization levels). The proposed descriptor obtained state of the art performances on the ETH-80 object categorization task [Leibe and Schiele, 2003], while also being successfully employed in indoor scene recognition tasks [Pronobis et al., 2010; Fornoni et al., 2010].

Local descriptors

Descriptors in this set, also referred to as *local features*, use small image patches (or regions) as visual primitives. Often, interest point detectors [Schmid et al., 2000] are used to select relevant locations in the image and extract features around them. The most influential work in this direction is arguably the *Scale Invariant Feature Transform (SIFT)* proposed by Lowe [2004]. The visual primitives considered in this approach are image patches extracted at local extrema of the scale-space. A 128-dimensional descriptor of each patch is obtained by computing a histogram of gradient orientations (with 8 reference orientations) in each of 4×4 sub-patches. Early works have demonstrated the potential of this low-level local descriptor, with its main advantage being the robustness w.r.t. occlusion and clutter. Lowe [2004] applied it to object instance recognition in occluded scenarios, while Caputo and Jie [2009] combined it with exact and approximate matching techniques to solve object and place recognition problems. Another popular descriptor of this family is the Histogram of Oriented Gradients (*HOG*). Introduced by Dalal and Triggs [2005] for human detection, it has also been used for generic image classification [Bosch et al., 2007] and scene recognition [Fornoni et al., 2010] tasks. Similarly to SIFT, HOGs capture the distribution of edge orientations within a given image region (computed on the output of a Canny edge detector). The orientations range is quantized into k bins and each edge is assigned to the corresponding binned orientation, with a weight proportional to the value of the gradient. This descriptor is not rotationally invariant but has shown good performance in indoor categorization tasks [Fornoni et al., 2010], in which usually the gradient directions are not strongly rotated.

While the image representation obtained by directly using low-level descriptors have shown some initial success in image classification and scene recognition tasks, these representations may not be able to robustly describe complex scenes, like indoor ones. In order to produce robust image representations, mid-level descriptors have thus been introduced.

2.2.2 Mid-level descriptors

Amongst the most successful image representations for scene recognition we find the mid-level ones. The two components that characterize these representations are:

1. Usage of low-level local features and some form of learning to identify a set of prototypical local features.
2. Usage of the learned prototypical local features to encode the low-level local features extracted from images.

Due to the feature encoding procedure used by these methods, mid-level descriptors are often referred to as mid-level *feature encodings*. Moreover, in analogy with bag-of-words models for text classification [Joachims, 1998], the set of prototypical features is often referred to as a *dictionary*, while each prototypical feature is referred to as a *visual word*, or a *codeword*. The learning procedure, in turn, is called *dictionary learning*. Finally, the image representations obtained using this type of descriptors are often referred to as *Bag of visual Words* (*BoW*) representations. In the following we review the most important forms of dictionary learning and encoding used to compute the mid-level descriptors and obtain BoW representations.

Hard encoding

One of the first examples of mid-level descriptors was proposed by Csurka et al. [2004]. The main idea of this approach is to use the k-means algorithm [MacQueen, 1967] to cluster local features (such as SIFT descriptors) extracted around interest points of several training images, to form a dictionary of k visual words. The learned dictionary is subsequently used to encode each SIFT feature in a given image. In this work, a local feature is encoded by an extremely sparse k-dimensional binary vector, with the only non-zero element in the position corresponding to the closest visual word in the dictionary. This encoding technique is also known as *hard quantization*, *hard assignment*, or simply *vector quantization*.

A first significant improvement of the BoW method was proposed by Fei-Fei and Perona [2005], where the interest point detection was replaced with a dense sampling scheme, extracting SIFT features over an evenly spaced grid of points (with a stride of 10px). There are two main reasons why in BoW models this sampling strategy works better than using interest points: 1) dense sampling provides a representation of uniform regions (such as sky, or walls), which are important for many recognition tasks (e.g. scene recognition) and are typically discarded by interest point detectors; 2) by densely sampling the features, the final image signature can be constructed using a much higher number of samples, providing more robust estimate of the visual words distribution [Chatfield et al., 2011].

The main advantage of the mid-level descriptors produced by hard-coding procedures lies in their robustness to some amount of deformation (thanks to the quantization of the features). This invariance, which for the first time enabled researchers to obtain very good results on small image classification tasks, is unfortunately also their main limitations when dealing

with more complex tasks. Indeed, due to quantization effects, a large part of the information contained in the low-level features is completely discarded. In order to address this problem, several alternative encoding schemes have been proposed [Yang et al., 2008, 2009; Jegou et al., 2010; Wang et al., 2010a; Zhou et al., 2010; Perronnin et al., 2010; Chatfield et al., 2011; Wang et al., 2013], as it will be discussed below.

Sparse and local encodings

Encoding a low-level feature using a soft combination of codewords can result in a smaller reconstruction error w.r.t. hard quantization. Moreover, as argued by Olshausen and Fieldt [1997], sparse combinations obtained using an over-complete dictionary may also be more robust to noise than dense combinations. Accordingly, Yang et al. [2009] propose to replace hard quantization with a sparse encoding framework, named Sparse Coding (SC). Instead of assigning a local feature to the closest cluster center, each low-level feature is encoded with a combination of codewords selected to minimize a cost function defined as the sum of a reconstruction error and a sparsity-inducing regularizer. This encoding framework has recently been complemented by approaches focusing on locality, as a form of sparsity [Yu et al., 2009; Wang et al., 2010a]. These approaches replace the sparsity-inducing regularizer in the cost function, with a localization error function, or with localization constraints. With this choice the codewords lying in a neighborhood of the feature to be encoded are favored over distant ones. Yu et al. [2009] provide theoretical and empirical evidence in favor of this approach, arguing also that while locality produces sparsity, the reverse is not true. Following this idea, Wang et al. [2010a] provide a fast approximated version of the encoding which simply consists of minimizing the reconstruction error for a given feature, using only its closest dictionary entries instead of the full dictionary. This fast approach is named approximated *Locality-constrained Linear Coding* (*LLC*).

Fischer and Super-Vector encodings

More recently, several techniques that encode the relative displacement of a given feature w.r.t. its assigned codewords have been proposed [Zhou et al., 2010; Perronnin et al., 2010; Jegou et al., 2010; Chatfield et al., 2011]. For example, Perronnin and Dance [2007] use a Gaussian Mixture Model (*GMM*) [Bishop, 2006] to learn a vocabulary composed of several weighted centers and diagonal covariance matrices. The descriptor of a given set of low-level features is then obtained by computing the gradient of the log-likelihood of the features w.r.t. the GMM parameters (the centers and the diagonal covariance matrices) and by subsequently normalizing the resulting gradient vector using the Fisher information matrix [Jaakkola and Haussler, 1998]. Differently from vector quantization, sparse coding and locality-constrained linear coding the resulting descriptor, which is called *Fisher Vector* (*FV*), has size $2kd$, where d is the dimensionality of the local features to be encoded (e.g. $d = 128$, for SIFT features). Using an improved version of this technique (with additional ℓ_2 and power normalization, to enhance the foreground and reduce sparsity) Perronnin et al. [2010] achieve very promising

results on several object recognition datasets [Everingham et al., 2007; Griffin et al., 2006], at the expense of a very high-dimensional representation. Using this encoding, Juneja et al. [2013] reported a state of the art performance (60.77% accuracy) on the MIT-Indoor-67 dataset. A more compact (kd-dimensional) and simplified version of this encoding (replacing GMM with k-means and thus not considering the covariance of the descriptors) named Vector of Locally Aggregated Descriptors (*VLAD*, [Jegou et al., 2010]) was also used for image retrieval tasks. Finally, another approach resulting in similar descriptors is the *Super-Vector* (*SV*) coding, introduced by Zhou et al. [2010]. In this work a given local feature is encoded by a sparse kd-dimensional vector in which there is only one non-zero d-dimensional sub-vector, in the position corresponding to the closest dictionary entry. The proposed encoding is motivated by an approximation error argument for β-Lipschitz derivative smooth functions [Zhou et al., 2010] and, similarly to VLAD and Fisher Vector, it encodes the relative displacement of each feature w.r.t. the considered center.

Benchmarking mid-level encodings

A detailed experimental evaluation of the most popular feature encoding techniques (such as Hard Coding, LLC, Fisher Vector and Super Vector) was performed by Chatfield et al. [2011], with the conclusions that better performances can be obtained by using:

1. Larger vocabularies and higher sampling densities for the local features.
2. More descriptive forms of encoding (e.g. techniques that also encode the relative displacement of a feature w.r.t. the considered centers, such as FV and SV).

Unfortunately, the most descriptive encodings lead to a drastic increase (by several orders of magnitude) in the dimensionality and density of the final image signatures, to the point that even for relatively small datasets the training data might not fit into memory [Chatfield et al., 2011]. Additionally, training a classifier on data having several hundred thousands non-sparse variables can be a challenging and time-consuming task. To address these problems for this type of encodings (Super Vector encoding, Fisher Vector encoding, etc), Chatfield et al. [2011] propose to compute a linear kernel matrix [Shawe-Taylor and Cristianini, 2004] using the image signatures and subsequently solve the dual classification problem.

Supervised Dictionary Learning

Another way to improve the performance of mid-level descriptors is to train the dictionary containing the feature prototypes in a supervised fashion. For example, Wang et al. [2013] propose a *Max-margin Multiple-instance Dictionary Learning* (*MMDL*) algorithm to cluster the local features in a set of $k+1$ different clusters, while also correctly classifying each feature as belonging to its correct class. Specifically, they propose to learn $c \times (k+1)$ hyperplanes $\boldsymbol{w}_{y,j}$ ($j \in \{1,2,..,k+1\}$, $y \in \{1,2,..,c\}$, where c is the number of classes in the considered problem) and use them to encode each local feature $\boldsymbol{x}_i$ with $v_{y,j}(\boldsymbol{x}_i) = \boldsymbol{w}_{y,j}^{\top}\boldsymbol{x}_i$. Similarly to Wang et al. [2013], Yang et al. [2008] propose to jointly learn a set of codewords $\begin{bmatrix} \boldsymbol{w}_{y,1} & \boldsymbol{w}_{y,2} & \dots & \boldsymbol{w}_{y,k} \end{bmatrix}$

for each class y, together with a set of combination coefficients $\boldsymbol{\alpha} \in \mathbb{R}^k$ common for all classes. In this case, however, the codewords $\boldsymbol{w}_{y,j}$ are also used for the final classification of the images. Using this framework with $k = 300$ visual bits, the authors show highly improved object recognition [Everingham et al., 2006] performances w.r.t. the standard BoW approach.

Learning mid-level patches

Most of the descriptors discussed so far only use a predefined set of local patches (possibly at multiple scales), represented using low-level features (e.g. SIFT). This choice might be suboptimal for a given task, as it only focuses on a fixed set of image structures, ignoring the others. To address this problem several authors have proposed methods to automatically discover visual structures that might be helpful for the considered image classification task [Yao et al., 2011; Pandey and Lazebnik, 2011; Singh et al., 2012; Doersch et al., 2013; Sun and Ponce, 2013; Li et al., 2013; Juneja et al., 2013]. For example, Singh et al. [2012] perform discriminative clustering on the HOG representation of randomly sampled image patches from each class, to obtain a class-specific set of mid-level *discriminative patches* which are then used as filters. The HOG representation of an image is then convolved with all the obtained mid-level discriminative patches and the responses pooled to produce the final image signature. The resulting method achieves promising results (38.1% accuracy) on the MIT-Indoor-67 dataset. Very recently, a similar approach was proposed by Juneja et al. [2013], where the discriminative clustering is replaced by a part-mining algorithm, initialized with super-pixels segmentations. The proposed approach outperforms Singh et al. [2012], obtaining a 46.10% accuracy on the MIT-Indoor-67 dataset. Similar approaches were recently proposed by [Doersch et al., 2013; Sun and Ponce, 2013].

2.2.3 High-level descriptors

While mid-level descriptors encode the low-level features w.r.t. a learned set of visual words devoid of any explicit semantic, high-level representations explicitly involve an intermediate classification w.r.t. human-understandable concepts. We can divide high-level representations into two main sets:

1. *Generic concepts*. The descriptors are obtained by evaluating a set of generic concept classifiers at multiple image locations and scales.

2. *Task-specific concepts*. The concept classifiers used are strictly related to the considered image categorization task (i.e. the considered concepts are the same as the final image classes). The concept classifiers are evaluated at multiple locations and scales of each image.

Generic concepts

The basic assumption of this class of descriptors is that the distribution of concepts in an image is highly correlated with its semantics and thus helpful when performing image classification. The work of Vogel and Schiele [2004] is one of the first successful contributions in this sense. The main idea of this work is to assess the typicality of an image w.r.t. a given category by comparing the statistic of occurrence of nine high-level concepts, with the prototypes learned from a dataset of images. Specifically, the authors collect a dataset of images of 6 outdoor natural scene categories, divided each image into 10×10 patches, and annotated each of them w.r.t. nine high-level concepts: *sky*, *water*, *grass*, *trunks*, *foliage*, *field*, *flowers* and *sand*. The high-level descriptor of a given patch is thus a binary vector encoding the occurrence of the generic concepts in the considered patch. Using this technique the authors achieve a categorization performance of 89.3% on the considered dataset. Nonetheless, if the manual annotation of the concepts in each image is replaced by an automatic classification using k-NN (with color and texture features) the classification accuracy drops to 67.2%. As reported by the authors, an in depth analysis of the results shows that the classification performance for a given class is indeed strongly correlated with the performance of the concept classifier that is most discriminative for the particular class.

Instead of representing an image with a low-dimensional histogram of concepts found in the image, Torresani et al. [2010] propose to use the responses of a large set of concept classifiers, trained with a large set of features. Specifically, the authors make use of a large-scale concept ontology [Naphade et al., 2006] to identify a set of (2659) representative categories (named *classemes*) for general object recognition tasks. A search engine is then used to collect a set of training images for each category and 13 different feature extractors are used together with a multiple-kernel learning method [Gehler and Nowozin, 2009] to learn an object category detector for each classeme. Given an image, the high-level descriptors are in this case obtained by evaluating at multiple locations the responses of the trained classeme classifiers. The proposed method achieves competitive object recognition performances on the Caltech-256 dataset [Griffin et al., 2006].

Another important work in this direction was proposed by Li et al. [2010]. In this work, the authors use several manually annotated image databases to select 177 frequently occurring objects. A DPM object detector [Felzenszwalb et al., 2008] is trained for each object and then evaluated on each image location, at multiple scales. Similarly to Torresani et al. [2010], the high-level descriptors for a given image are obtained by evaluating at multiple locations and scales the responses of the trained object detectors. The proposed approach is shown to achieve a recognition accuracy of 37.6% on the MIT-Indoor-67 dataset.

More recently, Li et al. [2013] propose to use a search engine to retrieve images for a set of 716 categories selected from WordNet [Fellbaum, 1998]. Each image is then modeled as a bag of patches, and *multiple instance learning* (*miSVM*) [Andrews et al., 2002] is used to discriminatively learn a so called *single-concept* patch classifier. Furthermore, since one single

patch model per category is not sufficient to represent its complexity, the authors propose to further cluster the patches that are positively classified (for each category) into 20 subcategories, forming a vocabulary of 716×20 words. The descriptor of a given patch of a query image is obtained by concatenating the scores of the trained miSVM classifiers.

Task-specific concepts

This stream of works finds one of its roots in the work of Szummer and Picard [1998], in which the authors use a k-NN classifier to independently classify regions of an image w.r.t. the target classes and to subsequently combine the classification results with a majority voting scheme. In their proposal the authors partition each image into 4×4 patches and classify each of them comparing it with all the patches extracted from the training set, independently of their location. The descriptor of a given patch is thus a binary vector, with the only 1 in the position corresponding to the class assigned to the patch. Using this technique the authors were able to separate indoor images from outdoor ones, with an accuracy of 90.3%. The same approach was followed by Serrano et al. [2004], but in this case the classifiers for each subpart of the image were SVM [Cristianini and Shawe-Taylor, 2010] with Gaussian kernels. An advantage of this technique is that the subregion scores are combined numerically rather than by majority voting, thus minimizing the impact of ambiguous labeling.

More recently, the Naive Bayes Nearest Neighbor (NBNN) algorithm [Boiman et al., 2008] was proposed for image classification tasks. Similarly to Szummer and Picard [1998], in NBNN the classes are directly represented by unordered sets of local descriptors extracted from patches of training images, and the algorithm classifies a query image by directly comparing its local descriptors with those contained in each class-specific set of local descriptors. Instead of using a k-NN classifier to assign each patch of a given image to the closest class, a patch-to-class distance is obtained using an approximated Nearest Neighbor search w.r.t. each class-specific training set of local descriptors. Image-to-class distances are then computed by summing the patch-to-class distances of all the local descriptors in the image. This representation can either be used to directly assign an image to the class at the shortest distance (as in Boiman et al. [2008]), or it can be used to form an image representation, with the classification step performed by a separate classifier [Tuytelaars et al., 2011].

2.3 Image signature

An image is not simply an orderless collection of features, on the contrary, a significant amount of information is carried by the spatial distribution of the features, their relative perceptual properties, or their importance for the task at hand. Not surprisingly, several authors have tried to encapsulate this information in the final image signature [Lazebnik et al., 2006; Cao et al., 2010; Jia et al., 2012; McCann and Lowe, 2012a; Sadeghi and Tappen, 2012; Sharma et al., 2012; Russakovsky et al., 2012; Xie et al., 2014]. This Section is devoted to review the most important works in this direction.

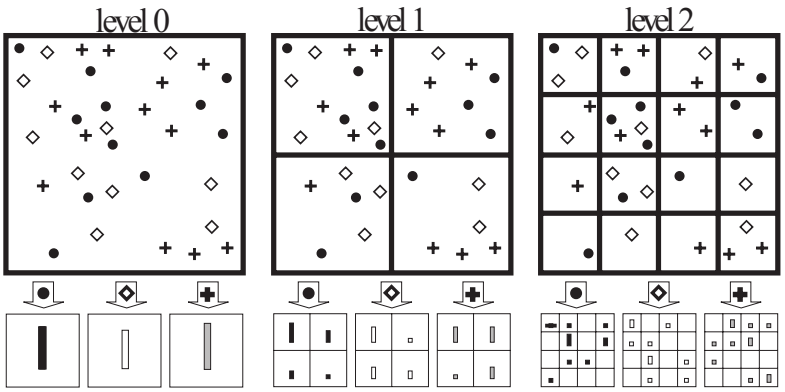

Figure 2.5 – Visualization of the Spatial Pyramid Matching approach. (Adapted from Lazebnik et al. [2006])

2.3.1 Spatial analysis

The spatial distribution of the features within each image is not uniform. Many scene recognition tasks, for example, do present a strong spatial consistency due to the effect of gravity: complex objects and structures tend to be distributed on the bottom of the image, while the top part is often occupied by more uniform areas such as sky or ceiling. In order to exploit these consistencies, several works have proposed ways to encode the position of the descriptors in the final image signature.

Image partitioning

One way to implicitly encode the position of the local descriptors in the final image representation is to partition each image into a set of regions with a fixed relative position (e.g. left half and right half of the image). A set of region-specific representations can then obtained by separately pooling the descriptors in each region, and concatenating the results in the final image signature. One of the most famous works in this direction, proposed by Lazebnik et al. [2006], introduces an extension of the BoW approach allowing to produce a coarse-to-fine quantization of the spatial position of each local feature. Instead of computing a single BoW representation on the full image, the authors propose to pyramidally partition each image into exponentially smaller patches and to compute a BoW representation for each of them. In particular, an image is uniformly partitioned into 2^{2l} patches having the same aspect ratio of the original image, for each $l \in \{0, 1, \ldots, L\}$. A BoW representation is subsequently computed for each patch. For example with $L = 2$, a BoW representation is computed on the full image ($l = 0$), four additional BoW representations are computed dividing the image in 2^2 ($l = 1$) patches, and sixteen final BoW representations are computed by evenly dividing the image into 2^4 ($l = 2$) patches. The final image signature is then obtained by concatenating the BoW representation of all the considered patches. A visualization of this technique is provided in Figure 2.5. With this extension (often referred to as *Spatial Pyramid Matching*, or *SPM*) of the original BoW representation the authors demonstrated significative performance improvements on

the 15-Scenes dataset (from 74.8% to 81.4%).

Inspired by the success of SPM, several authors have proposed alternative image partitioning techniques. Marszałek et al. [2007] propose to combine the ($L = 1$) SPM approach with a horizontal spatial grid with three (upper, middle and lower) regions. The proposed partitioning scheme proved to be important to obtain the best results for the *Pascal 2007 Visual Object Classes (VOC) Challenge* [Everingham et al., 2010]. Cao et al. [2010] propose to partition an image along several directions (e.g. horizontal, vertical and with other angulations) and in circular bins, compute a BoW representation for each cell and use a boosting algorithm to select the most representative patches. Related approaches are proposed by Sharma and Jurie [2011]; Jiang et al. [2012], obtaining promising results on the 15-Scenes dataset. In these works, similarly to Cao et al. [2010], the authors use a learning algorithm to select a task-oriented image partitioning, from an initial over-complete set of image partitions. A slightly different approach was proposed by Jia et al. [2012]. Instead of using a learning algorithm to select the best image partitioning scheme from an over-complete set of patches, the authors propose to greedily select the most useful single features from the full set of histogram bins extracted from all the randomly generated partitions. Interestingly enough, this work shows that at a constant image signature dimensionality, a random feature selection from an over-complete set outperforms the SPM representation.

Spatial encodings

Though the image partitioning schemes discussed so far succeed in capturing some spatial information, they perform hard quantization of the spatial information and require to mine the set of all possible image partitions. This either results in a consistent increase in the dimensionality of the representation, or in a complex training procedure. Alternative approaches avoiding some of these issues have been introduced by Koniusz and Mikolajczyk [2011]; Krapac et al. [2011]; Sánchez et al. [2012]; McCann and Lowe [2012a]. McCann and Lowe [2012a], for example, propose to concatenate the relative location of each feature to the local descriptor, with a weight λ to balance the importance of the location description w.r.t. the appearance description. The augmented local descriptors are then employed in a canonical dictionary learning procedure with a very large number of visual words (up to 65536), obtaining a dictionary in which each visual word also contains a prototypical position of the word in the image. Using this technique with multiple values of λ, the authors show promising performances on object recognition tasks [Fei-Fei et al., 2007; Griffin et al., 2006]. Unfortunately, though the absence of multiple image partitions speeds up the training and contributes at keeping the dimensionality low, the need for large dictionaries and multiple encodings (with different values for λ) largely cancels the last improvement.

2.3.2 Saliency analysis

While the spatial distribution of the features in each class might follow a consistent pattern and thus provide important information about the category of a query image, the consistency in the spatial location of the descriptors might not be the only one, or the best one that can be exploited. In indoor scenes, for example, the close-up distance between the camera and the subject makes small variations in the viewpoint to cause drastic changes in the captured scene. Moreover, even images taken from very similar viewpoints still present an intrinsic variability in the spatial layout of objects and structures present in the image, due to the intrinsic variability in the design of the environments. Instead of focusing on the exact location of the local features, several authors have thus tried to treat the image features according to other properties, such as their contrast [Law et al., 2012], their saliency for the task at hand [Sadeghi and Tappen, 2012], or their 3D orientation [Xie et al., 2014]. Amongst the approaches making use of saliency, we can distinguish three main trends in the literature:

1. Approaches that make use of saliency to select and match a subset of the image features that are more discriminative for the task at hand, regardless of their exact position in the scene.
2. Approaches that weight the importance of features in a given position, according to their saliency in the scene.
3. Approaches that make use of saliency to segment the image into foreground and background, and separately process the features in the two regions.

Examples of the first category are the works of Gao and Vasconcelos [2005]; Moosmann et al. [2006] and Parikh et al. [2008], in which patches are randomly sampled from the images according to a discriminatively learned saliency map. In the same category, but subverting the usual assumption that high-saliency regions are the most informative ones, Rapantzikos et al. [2009] employ a bottom-up spatio-temporal saliency model to segment videoclips and progressively discard high saliency regions. Though not explicitly making use of the notion of saliency, other works have focused on identifying the most important image areas for the classification task at hand [Pandey and Lazebnik, 2011; Sadeghi and Tappen, 2012]. For example, using large patches of fixed size, Sadeghi and Tappen [2012] propose to train a multiclass discriminative latent SVM [Yu and Joachims, 2009] to learn a patch prototype for each class. Specifically, for any given query image, the maximal patch detection score of the trained classifier is used to estimate the confidence for that image to belong to a given class. The outputs of several patch classifiers (with patches of different sizes) are then concatenated together to obtain a signature for each image, and a non-linear classifier is trained on this representation. Implicitly, the proposed technique focuses only on the most important (salient) patches of each image, discarding the information outside them. The authors applied this approach to the MIT-Indoor-67, the 15-Scenes and the Sports datasets, achieving respectively 44.41%, 85.81% and 86.25% accuracy.

In the second category we find the works of Harada et al. [2011] and Sharma et al. [2012], where images are segmented using a regular grid, and the local descriptors extracted within a given

patch are weighted according to the discriminative saliency of the patch. Another approach that has been proposed weights separately each single local descriptor according to a measure of its discriminative saliency [Marszalek and Schmid, 2006; Feng et al., 2011], as opposed to uniformly weighting the descriptors according to the patch they belong to.

Some approaches from the third category include [Russakovsky et al., 2012] and [Law et al., 2012]. For example, Russakovsky et al. [2012] use a multi-instance learning approach to automatically segment an image into a foreground patch and the remaining background. Each of the two regions is separately represented using a BoW encoding and the two representations are then combined together. The proposed approach is not tested on scene recognition problems, but it is shown to obtain state of the art performance on PASCAL 2007 object recognition dataset [Everingham et al., 2007].

2.3.3 Pooling

Once the local descriptors have been extracted and analyzed and the pooling regions have been defined, the final image descriptor can be constructed by concatenating the results of the pooling operation applied to each of the considered pooling regions. The pooling operation, in turn, can be performed in several ways:

- *Collection*. Using this technique, each considered pooling region is simply represented by the unordered collection of local descriptors extracted from it, without any further processing.

- *Average pooling*. Using this technique, the descriptors belonging to a given region are averaged together [Csurka et al., 2004].

- *Max pooling*. Taking inspiration from biology [Serre et al., 2005] and in order to better preserve the responses to rarely occurring visual words, only the maximal value measured in each dimension of the descriptors belonging to a region is kept.

The first type of pooling (collection) is used mostly by early approaches using low-level local representations [Schmid et al., 2000; Caputo and Jie, 2009] and by the NBNN algorithm [Boiman et al., 2008]. Average pooling is typically used with BoW models using hard quantization, in the high-level representations considered by Szummer and Picard [1998] and Vogel and Schiele [2004], and in Fischer Vector and Super Vector encodings. Max pooling is typically used by BoW approaches employing sparse, or local encoding techniques, as well as high-level approaches making use of pre-trained object and concept detectors, such as Object-bank, and Classemes. From the point of view of classification, average pooling is well-known to be a good performer when used with kernel classifiers [Shawe-Taylor and Cristianini, 2004]. On the other hand, an analysis by Boureau et al. [2010] supports the idea that max-pooling is better suited to linear classifiers.

2.4 Classification

The last component of a typical scene recognition system is the classifier. Over time, a large number classifiers have been proposed and used in scene recognition systems, and many possible criteria may be used to form a taxonomy [Jain et al., 2000], [Kuncheva, 2004, Chapters 1,3]. Among them, one which is relevant to this thesis makes a distinction between two categories:

1. *Monolithic* classifiers, where each class is represented by a single unitary model.
2. *Multi-component* classifiers, where each class is represented by a model that can be decomposed into a set of components, while the component (or combination of components) used is allowed to vary according to the point to be classified.

Although the term "Multi-component" is mainly used in the object detection literature [Dollár et al., 2008; Felzenszwalb et al., 2010; Gu et al., 2012], we apply it to any classifier (not necessarily designed for object detection) matching the description at point 2. above. We favor this denomination over other ones like "Multi-prototype" [Aiolli and Sperduti, 2005], or "Multi-hyperplane" [Wang et al., 2011b], as it does not carry any connotation about the form of the components (e.g. "prototypes", or "hyper-plane") and it is thus more general. The remaining part of this Section is dedicated to a short review of the most relevant algorithms in the two above mentioned classes.

2.4.1 Monolithic Classifiers

Main characteristic of monolithic classifiers is that it is not possible (or it is not evident how) to decompose the model used by the classifier into separate components specialized for a given set of samples. We can distinguish two main groups of monolithic approaches, according to the complexity of the decision boundary that they are able to deliver:

1. *Linear classifiers.* Classifiers belonging to this group are only able to separate the samples in the input space using a hyperplane (a linear decision boundary).
2. *Non-linear classifiers.* Classifiers belonging to this group are able to produce complex decision surface, non-linearly separating the samples in the input space.

In the remaining of this Section we provide a brief review of the most important approaches in each of the two above mentioned groups.

Linear Classifiers

A typical example of monolithic linear classifiers is that of linear *Support Vector Machines* (*SVMs*) [Cristianini and Shawe-Taylor, 2010]. This category of algorithms makes use of the theory of max-margin classifiers [Cristianini and Shawe-Taylor, 2010] to select one single hyperplane that linearly separates the positive samples from the negative ones, with good generalization properties. Though the success of SVMs is largely due to the theory of kernels [Shawe-Taylor and Cristianini, 2004], in the last years there has been a pronounced

interest in the linear version of the algorithm, mostly due to the emergence of well performing high-dimensional mid-level representations [Yang et al., 2009; Wang et al., 2010a; Perronnin et al., 2010], coupled with the increasing availability of large scale datasets [Torralba et al., 2008; Deng et al., 2009]. For very high-dimensional problems, indeed, linear classifiers tend to perform better than non-linear ones, as the latter are harder to train (they often present more hyper-parameters to tune, or require non-convex optimization procedures) and more sensitive to the "curse of dimensionality" [Bengio et al., 2005]. Another important advantage of linear SVMs is due to their training complexity which only grows linearly w.r.t. the number of training samples and the dimensionality of the data [Joachims, 2006]. On the other hand, a linear SVM can only learn a single linear decision boundary (hyper-plane) and thus presents a limited ability to represent complex class patterns, especially on low-dimensional problems [Yoon et al., 1993; Blackard and Dean, 1999]. This limitation can be tackled at a controllably low computational cost, by making use of the multi-component methods discussed in the Section 2.4.2.

Non-linear classifiers

Linear-threshold algorithms, like the linear SVM algorithm introduced above can only learn linear decision boundaries. However it is often the case that data is not separable by a simple linear hyperplane. In such cases, a separating hyperplane may still be found by non-linearly pre-mapping the original vectors into a new high (and possibly infinite) dimensional space, called the Feature Space, where the samples become linearly separable. A linear classifier can subsequently be trained in this space, resulting in a non-linear decision boundary in the original space. Arguably, the most popular example of this technique is represented by kernel SVMs [Shawe-Taylor and Cristianini, 2004], which are grounded well-understood algorithms able to deliver state of the art performance on almost any task. These algorithms implicitly project the samples into a Feature Space, and perform a linear classification in this space, without neither explicitly defining the space, nor explicitly computing the projection. Unfortunately, the testing time of such methods grows linearly with the number of support vectors, while their training complexity grows cubically with the training set size. In order to address these problems, several approximation methods have been proposed [Williams and Seeger, 2000; Rahimi and Recht, 2007; Cotter et al., 2011; Vedaldi and Zisserman, 2012]. A classical example is the Nyström method [Williams and Seeger, 2000], in which a data dependent m-dimensional explicit feature map for a given kernel is formed by approximating the eigendecomposition of the $n \times n$ training kernel matrix, using only a $m \times m$ sub-matrix ($m \leq n$). More recently, Rahimi and Recht [2007] proposed to approximate any translation-invariant kernel (like the Gaussian) using an explicit feature map obtained by a randomized projection matrix sampled from the Fourier transform of the kernel function. Using this method, the dimensionality required to achieve a desired approximation precision ϵ is unfortunately proportional to $1/\epsilon^2$. The problem can be alleviated whenever the input features are very sparse and the kernel to approximate is the Gaussian. Indeed, by using an explicit feature map based on a truncated Taylor expansion of the kernel function, Cotter et al. [2011] showed that

the approximation precision depends only on the number of non-zero elements in the input features, making it more suitable for sparse data.

2.4.2 Multi-component Classifiers

Multi-component classifiers are non-linear classifiers representing each class with a set of sub-models (components), specialized to different sets of samples. These methods have the ability to select the component(s) to be used according to the query point to be classified, and are sometimes referred to as *adaptive classifiers* [Jacobs et al., 1991; Jain et al., 2000; Wang et al., 2011b]. The set of multi-component classifiers can accommodate a large number of approaches, among which we find:

1. *Nearest Neighbor classifiers* [Cover and Hart, 1967].
2. *Naive Bayes Nearest Neighbor classifiers* [Boiman et al., 2008].
3. *Local classifiers* [Bottou and Vapnik, 1992].
4. *Ensemble methods* [Kuncheva, 2004].
5. *Manifold learning methods* [Yu et al., 2009; Ladicky and Torr, 2011].
6. *Multi-hyperplane classifiers* [Kohonen et al., 1996; Aiolli and Sperduti, 2005].

In the following we provide a review of the most important examples in each of the above classes.

Nearest Neighbor classifiers

Arguably, the most simple example of a multi-component classifier, and one of the first classifiers to be employed in scene recognition tasks [Oliva and Torralba, 2001; Szummer and Picard, 1998] is the *Nearest Neighbor* (*NN*) classifier [Cover and Hart, 1967], in which the model of a given class consists of all the training samples (components), while a query sample is simply assigned to the class of the nearest sample (or of the majority of the k closest samples) in the training set. The NN algorithm produces non-linear boundaries, does not require any training and, since the training samples used to produce a decision for a given query sample are only those in the neighborhood of the query sample, the method also falls in the category of adaptive classifiers. Unfortunately, due to the lack of training, the set of components (in this context named *prototypes* [García et al., 2012]) used by the NN model is large and have the simplest possible form: the full unprocessed set of training samples. The algorithm requires storing of all the training samples and performing an expensive search each time a query sample has to be classified. These drawbacks have been studied by many researchers and many solutions have been proposed to speed up the NN computation by selecting, or generating a representative set of prototypes to be used in testing [García et al., 2012; Triguero et al., 2012]. In scene recognition tasks, this method was used by Szummer and Picard [1998] to classify images as being shot indoor, or outdoor, and by Oliva and Torralba [2001] to classify outdoor scenes into eight categories (see also Section 2.2.1). The approach is

also used to compute the image-to-class distances in the NBNN methods [Boiman et al., 2008] discussed below, and in Section 2.2.3.

Naive Bayes Nearest Neighbor classifiers

Classifiers belonging to this group are Nearest Neighbor approaches designed for problems in which each sample can be modeled as a bag of local features, while each local feature can be considered to be conditionally independent from the others, given the class of the sample. A prominent example of a problem that can be modeled in this way is that of image classification. Indeed, as also discussed before, each image can be represented as a collection of local features, where the correlations among the local features can be ignored, once the label of the image is known.

In Naive Bayes Nearest Neighbor (NBNN) approaches [Boiman et al., 2008] classes are directly represented by unordered sets of local features extracted from the training samples. A NBNN algorithm classifies a query sample by directly comparing its local features with those contained in each class-specific set. Specifically, as also described in Section 2.2.3, a local-feature-to-class distance is obtained using an approximated Nearest Neighbor search w.r.t. each class-specific training set. A sample-to-class distance is then obtained by summing the local-feature-to-class distances for all the local features of the query sample. As it is easy to see, we can thus regard the classifier as a multi-component classifier, with one component for each prototypical local feature in the training set.

The resulting method has been mostly applied to image classification problems, achieving performance comparable with that of simple BoW representations. By replacing the sample-to-sample distances used by the standard Nearest Neighbor algorithm, with a sample-to-class distance, the NBNN algorithms promise a higher degree of robustness and generality, especially when applied to categorization problems. Nonetheless, many authors have pointed out how its success relies heavily on the large number of local features in the training set, limiting its scalability to real-world applications [Behmo et al., 2010; Wang et al., 2010b, 2011a; Tuytelaars et al., 2011; Timofte et al., 2012; Escalante et al., 2014]. Moreover, the somehow flat structure imposed on the space of local features limits the expressiveness of the model, which tends to underperform methods based on learning [Behmo et al., 2010; Wang et al., 2011a; Timofte et al., 2012]. Based on these observations, many authors have thus tried to better exploit the structure of the local features to improve the recognition performance, reduce the testing time, or reduce the memory footprint of the algorithm. In Local NBNN [McCann and Lowe, 2012b], the class-conditional probability estimates for a given local feature are performed by restricting the search only to classes whose local features are present in the neighborhood of the considered feature. By ignoring the probability estimates for classes that do not lie in a neighborhood of the feature, the authors show an increase in the recognition performances and a greater scalability w.r.t. the number of classes. In [Vitaladevuni et al., 2013], the authors propose to apply unsupervised learning (PCA) to the local features. This simple idea allows to compress the data and speed up the distance computation, while preserving

or increasing the predictive performances. [Timofte et al., 2012] modify the NBNN scoring function, replacing the 1-NN patch-to-class distance computation with a k-NN approach (with $k > 1$), coupled with LLC encoding, Sparse Coding, or Collaborative Coding [Zhang et al., 2012]. In LI2C [Wang et al., 2011a], the Euclidean distance is replaced by a Mahalanobis distance and a supervised distance learning procedure is performed to learn a set of class-specific metrics. This results in improved recognition performance, with good results obtained by using only five to ten percent of the training data. Instead of learning a metric for each class, Tuytelaars et al. [2011] introduce a method to construct a kernel from the vanilla NBNN likelihood estimates and proposed to use it to train a SVM classifier. The main advantage of this approach is that it allows to integrate NBNN with existing kernel-based methods, for example by combining it with kernels based on BoW models. In a more recent work [Rematas et al., 2012], each class is partitioned into several clusters and, for a given query image, an NBNN image-to-cluster distance is computed for each cluster. These NBNN image-to-cluster distances are then used to construct a richer NBNN kernel, resulting in improved performances w.r.t. [Tuytelaars et al., 2011]. Adopting a rather different approach, [Çakir et al., 2011] propose to learn a set of prototypical local features for each class, by training a class specific codebook. During prediction the NBNN image-to-class distances are then computed by using the learned codebooks, instead of the complete training set. Finally, there have also recently been attempts to apply Prototype Generation (*PG*) techniques [Triguero et al., 2012] to the NBNN algorithm. Specifically, Escalante et al. [2014] perform an evaluation of a large number of PG algorithms, for the task of generating a representative set of SIFT prototypes for each class. The study proves some of the considered algorithms to be able to significantly reduce the number of prototypes, while preserving (or occasionally even increasing) the performance of NBNN, on a simple object recognition task.

Local classifiers

A special category of multi-component algorithms consists of *local classifiers*, in which each component is explicitly specialized to a given region of the input space [Bottou and Vapnik, 1992]. The appealing statistical properties of local classifiers have been analyzed by Vapnik [1991]. The idea is that the capacity of a classifier should locally match the density of the training samples in a specific area of the input space: low-density areas of the input space would require a low-capacity classifier, while more populated zones would benefit from a high capacity one. Such localization could be achieved by either using a separate classifier with a specific capacity in each area, or by building a set of classifiers with the same capacity, but constrained to have access to different amounts of training samples originated in different parts of the space. Following the second approach, many successful algorithms have been proposed.

Building on the Nearest Neighbor classifier and on the theory of Vapnik [1991], Bottou and Vapnik [1992] proposed to train a linear classifier using k-Nearest Neighbors (k-NN) of a testing sample and then use it to label the sample. Yang and Kecman [2008] introduced a

properly weighted Euclidean distance for the k-NN computation, while Zhang et al. [2006] and Kecman and Brooks [2010] used a linear (and a non-linear) SVM as the local classifier. These simple local models perform surprisingly well in practical applications. However, due to the k-NN search and the local training that has to be performed for each testing sample, such models are memory inneficient and slow to test, which makes them unsuitable for large scale problems. To alleviate these problems Cheng et al. [2010] proposed to discriminatively cluster the training set into k clusters and separately train and test an SVM for each of them, thus eliminating the problem of training a separate classifier for each testing sample, while reducing also the time for the NN search. A similar approach was proposed by [Segata and Blanzieri, 2010], using a different technique to obtain the set of centers and to assign each query point to a center.

Manifold learning methods

More recently, manifold learning methods have been proposed to approximate non-linear functions, using a local combination of linear ones. For example, in Yu et al. [2009], the combination coefficients are given by the reconstruction coordinates obtained using Local Coordinate Coding (LCC). In Locally Linear SVM (LLSVM), Ladicky and Torr [2011] make use of inverse Euclidean distances as a form of manifold learning, while also learning all the local models in a single optimization problem. LLSVM was shown to outperform LCC both in terms of number of anchor points needed (hundreds instead of thousands) and accuracy. This approach was further improved in Zhang et al. [2011], by combining it with a more sophisticated manifold learning scheme, named Orthogonal Coordinate Coding (OCC). In Qi et al. [2011] a hashing function is used to group samples with the same hash and to train a separate model for each hashing value. To smooth the resulting irregular piecewise-linear boundary, the authors also introduce a "global reference classifier", which is additionally used to classify test samples with unknown hashes. Although efficient, all the methods in this group use sample-to-component assignments that are learned with a separate unsupervised learning phase, unaware of the supervised classification task. Hence, similarly to local classifiers, the components are allocated only taking into account the distribution of the samples, without considering the labels.

Ensemble methods

Adaptive mixtures of local experts [Jacobs et al., 1991] have also been used to learn non-linear functions as local linear combinations of linear ones, with a trainable gating function assigning a weight to each model, for each sample [Gönen and Alpaydin, 2008; Fu et al., 2010]. In these methods the sample-to-model assignments are obtained in a discriminative way. However, the optimization problem is often very complex and the training procedure quite slow.

Multi-hyperplane classifiers

Another set of multi-component approaches closely related to the work presented in this thesis includes the Multi-Prototype SVM (MProtSVM) by Aiolli and Sperduti [2005] and the Adaptive Multi-Hyperplane Machine (AMM) by Wang et al. [2011b]. In these works the authors propose to train a set of competing linear SVMs (hyperplanes) for each class, assigning a given query sample to the class of the hyperplane producing the maximal score for that sample. The AMM model is trained with a stochastic procedure and the number of hyperplanes used to represent a class is automatically adapted to the task. Specifically, during training a new hyperplane is added when none of the existing ones is able to explain a given training sample (by producing a positive score), while hyperplanes with small enough norms are periodically pruned. A more general architecture that has been used to obtain multi-component models is represented by the Latent SVM framework [Felzenszwalb et al., 2008; Yu and Joachims, 2009]. In Latent SVMs, a feature map depending on a latent variable is used to map each sample to a given feature space, while the value of the latent variable (and thus the projection of the sample) is selected to be the one maximizing the confidence of the linear model being trained, or evaluated. The main limitation of all these classifiers is that only one single hyperplane is selected to be used for each query sample. Consequently, they can only learn non-smooth piecewise-linear decision boundaries.

2.4.3 Sub-categories and multiple components in object and scene recognition

The appearance of artificial entities such as objects and indoor scenes can change profoundly with the design of the entity and the viewpoint, while different objects or scenes might share common properties. A way to model these variabilities and relationships is to impose, or learn a hierarchy of categories and sub-categories, reflecting their semantic, task-driven, or perceptual similarities. The main advantage of perceptual and task-driven hierarchies over semantic ones is that the structure of categories and sub-categories can be learned from the data, rather than being imposed from a human semantic perspective. Based on this idea, several authors have tried to learn a decomposition of each class into a set of sub-categories [Felzenszwalb et al., 2010; Gu et al., 2012; Divvala et al., 2012; Lan et al., 2013; Hoai and Zisserman, 2013]. In object detection tasks, for example, the state of the art *Deformable Parts Models* (*DPM*) by Felzenszwalb et al. [2010] explicitly make use of multiple components to represent each object. Specifically, a Latent SVM is used to learn several components for each object, each one initialized using images with bounding boxes of a given aspect ratio (e.g. frontal images versus lateral ones). Divvala et al. [2012] showed that considerable performance gains could be obtained by increasing the number of components and switching the initialization step from the aspect-ratio heuristic to an appearance-based clustering (k-means). Conspicuous improvements were also reported by Lan et al. [2013] using a discriminative initialization of the components, based on exemplar-SVMs [Malisiewicz et al., 2011] and Affinity Propagation [Frey and Dueck, 2007].

The ideas of sub-category and multi-component modeling have also encountered some success in scene recognition tasks [Boureau et al., 2011; Pandey and Lazebnik, 2011; Parizi et al., 2012; Vitaladevuni et al., 2013]. For example, Pandey and Lazebnik [2011] adapt the DPM model of Felzenszwalb et al. [2010] for indoor scene recognition problems. Parizi et al. [2012] propose instead to partition the image into patches and use a Latent SVM, or a generative model to iteratively learn the patch-to-component assignments and the models. Finally, another multi-component method that has been successfully used on scene recognition databases is the NBNN algorithm [Behmo et al., 2010; Wang et al., 2011a; Çakir et al., 2011; Vitaladevuni et al., 2013]. As discussed in Section 2.4.2, this method is based on the NN classifier and as such it inherits its advantages (e.g. lack of training procedure) and all its limitations (e.g. high computational and space complexity during testing).

3 Spatial and Saliency-driven Representations of Visual Scenes

One of the main trends in visual recognition literature is employing representations which couple statistical characterizations of the image, with a description of their spatial distribution. This is usually done by computing statistical encodings of different image patches, using a rigid spatial pyramid (*SPM*) image-partitioning scheme, and by concatenating them to obtain the final image signature. While these fine-grained spatial encodings are able to capture and exploit the spatial regularities in the images, they are unsuitable to handle the spatial variabilities characteristic of difficult image categories, such as indoor scenes, or sports scenes. Furthermore, they often result in high-dimensional representations, which may be unsuitable for scene recognition applications having memory, or computational efficiency requirements.

In this Chapter we present a bottom-up approach designed to discover visual structures regardless of their exact position in the scene. The proposed methodology makes use of saliency maps to segment each image in two regions: the most salient region and the remaining non-salient one. This segmentation provides a representation of the images that depends on the relative perceptual complexity of the discovered regions, and that is complementary to the canonical spatial representations. We evaluate the proposed technique on the three public scene recognition datasets considered throughout this thesis [Quattoni and Torralba, 2009; Lazebnik et al., 2006; Li and Fei-Fei, 2007]. Our results prove this approach to be effective in the indoor scenario, while being also meaningful for other scene recognition tasks, such as the recognition of sports scenes. Furthermore, the proposed image representations are also up to one order of magnitude more compact than the canonical SPM representations.

3.1 Introduction

Since the seminal works of Oliva and Torralba [2001] and Lazebnik et al. [2006], the mainstream approach to scene recognition has been based on global, appearance-based image representations, computed over rigid spatial partitions of the image. This approach, in various forms, has given good results for many image classification problems, but proved to be inadequate when dealing with complex image categories, such as indoor scenes [Quattoni

Figure 3.1 – Saliency-driven segmentation of images from office and kindergarden categories (MIT-Indoor-67 dataset [Quattoni and Torralba, 2009]). For each image, a saliency map [Itti et al., 1998] was computed and then then segmented in two regions: the most and least salient 50%. Dark areas correspond to low saliency regions.

and Torralba, 2009], characterized by high degrees of spatial variability. Furthermore, due to the pyramidal quantization of the spatial information, the representations obtained using this approach are also very high-dimensional. In this Chapter we investigate these issues and propose a *Saliency-driven Perceptual Pooling* approach (*SPP*) designed to capture structural properties of the scenes, independently from their position in the images. Specifically, we make use of a saliency map to segment each image in two regions: the most and the least salient image regions. The SPP image representation is then obtained by separately pooling the local features extracted from each region (salient and non-salient) and concatenating the resulting image descriptors. Visualization examples of this pooling technique are shown in Figures 3.1 and 3.2. As it can be seen, the saliency-driven pooling can isolate areas sharing common perceptual properties, while being located in different regions of the image, and without explicitly modeling their semantics.

Saliency operators are typically designed as visual attention systems useful to predict human fixations [Harel et al., 2006; Itti et al., 1998]. Although human fixation are not directly related to the importance of a given area for scene recognition tasks, when humans capture an image they do so by selecting the view-point and the framing using their attentional mechanisms. Consequently, the most salient areas of an image captured by a human may actually be related to the category of the scene, with the relationship being established by the human observer at the time of the observation (i.e. when the image was taken). It can also be empirically shown (see Section 3.4.2) that in scene recognition problems the saliency of a patch is related to its visual complexity, with salient areas having higher average complexity. By pooling together patches with a similar level of saliency we may thus be able to capture perceptually coherent structures (with a similar level of visual complexity) that are related to the category of the scene, without explicitly modeling their semantics, or their spatial locations. In Figure 3.1 we can see, for example, that in the kindergarden category, chairs and desks located in different regions

Figure 3.2 – Saliency-driven segmentation of images from badminton and snowboarding categories (UIUC-Sports dataset [Li and Fei-Fei, 2007]). For each image, a saliency map [Itti et al., 1998] was computed and then then segmented in two regions: the most and least salient 50%. Dark areas correspond to low saliency regions.

are all captured by the salient region, while floors, ceilings and walls are collected by the non-salient one. A similar effect is illustrated in Figure 3.2, for both indoor and outdoor sports scenes (badminton and snowboarding). As it can be seen, while the pose, location and scale of the subjects differ from image to image, they are consistently captured in the most salient area of the image. On the other hand, important information related to the environment in which the events are taking place is preserved in the non-salient areas.

The contributions presented in this Chapter are the following: (1) we propose a Saliency-driven Perceptual Pooling (*SPP*) approach able to group areas of the image with different perceptual complexities, regardless of their exact position in the image; (2) we propose a saliency operator making use of the local descriptors that are to be pooled, as the sole input for the computation of the map; (3) we show that the combination of SPP with a simple spatial pooling scheme results in a compact descriptor, outperforming higher-dimensional SPM encodings, on two of the three publicly available scene recognition datasets considered in this thesis.

The rest of the Chapter is organized as follow: in Section 3.2 we briefly review the related approaches discussed in Chapter 2, highlighting the differences with our method; in Section 3.3 we detail our technique and the saliency operators being used; in Section 3.4 we empirically analyze the various components of the proposed method and report the experimental results on several scene recognition databases; a final discussion is provided in Section 3.5.

3.2 Related works

We present the works related to our approach by instantiating the scene recognition pipeline introduced in Chapter 2 for the method discussed in this Chapter. A visualization of the specific scene recognition pipeline used in this work is reported in Figure 3.3. As it can be seen, similarly to the approaches discussed in Section 2.2.2 we adopt the framework of mid-level feature

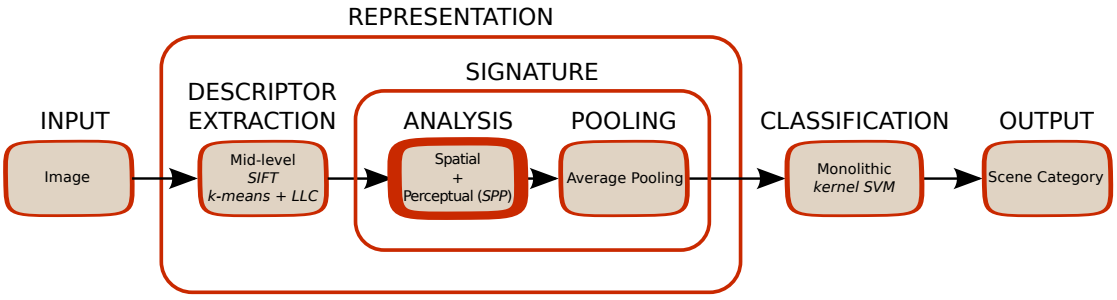

Figure 3.3 – The scene recognition pipeline proposed in this Chapter. The component of the pipeline related to the main contribution of this Chapter is highlighted by a thick border.

representation, using dense SIFT patches [Fei-Fei and Perona, 2005] with k-means [Csurka et al., 2004] and LLC [Wang et al., 2010a] for dictionary learning and feature encoding. The image signatures are then obtained by means of the average pooling procedure discussed in Section 2.3.3, while the image classification is delegated to a monolithic non-linear kernel SVM classifier [Cristianini and Shawe-Taylor, 2010], discussed in Section 2.4.1.

With respect to this pipeline the main contribution of this work concerns the definition of the regions on which the pooling operation is performed. Similar to the approaches discussed in Sections 2.3.1 and 2.3.2, we propose to define the pooling regions by performing a spatial and saliency analysis of the local features. However, differently from the methods discussed in the above mentioned Sections, we propose to combine both the spatial and the saliency-driven representation in the final image descriptor. Amongst the numerous approaches proposed to exploit the spatial consistencies in the images (see Section 2.3.1), the one that is most closely related to the technique used in this work is the approach used by Marszałek et al. [2007], winning the Pascal VOC 2007 competition [Everingham et al., 2010]. Differently from this method, the spatial pooling approach proposed in this Chapter does not make use of any SPM partitioning scheme, and consequently results in more compact image descriptors.

As discussed in Section 2.3.2, there are three main classes of approaches related to the Saliency-driven Perceptual Pooling approach presented in this Chapter:

1. Approaches that make use of saliency to select and match a subset of the image features that are more discriminative for the considered task, regardless of their exact position in the scene.
2. Approaches that weight the importance of features in a given position, according to their saliency in the scene.
3. Approaches that make use of saliency to segment the image into foreground and background, and separately process the features in the two regions.

The method that we propose substantially differs from the ones in the first two categories, in that we neither use saliency to select which features to retain, nor preserve the spatial information associated to the salient/non-salient regions. We instead make use of a bottom-up saliency operator to define pooling regions that preserve perceptually coherent structures in

the final image signature. We may thus include this work in the third category, of "approaches that make use of saliency to segment the image into foreground and background, and separately process the features in the two regions". Within this category, the proposed approach is to the best of our knowledge the first to make use of bottom-up saliency operators to define the foreground and background regions.

A saliency function directly operating in the space of the local features extracted from the image was also proposed in Walker et al. [1998]. Differently from this method, our approach makes use of the AIM framework [Bruce and Tsotsos, 2005], turning the multi-dimensional joint distribution estimation problem into a set of independent mono-dimensional estimation problems.

This Chapter is based on the work first presented in Fornoni and Caputo [2012].

3.3 The proposed approach

Let us assume that an image (of height h and width w) is represented by a matrix $\boldsymbol{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_r]^T \in \mathbb{R}^{r \times d}$, of d-dimensional local descriptors. Let us also assume to have a visual vocabulary $\boldsymbol{V} \in \mathbb{R}^{d \times k}$ (where k is the number of visual words), used to encode $\boldsymbol{X}$ into an intermediate representation $\boldsymbol{C} = [\boldsymbol{c}_1, \ldots, \boldsymbol{c}_r]^T \in \mathbb{R}^{r \times k}$. A histogram of visual words over a region $\mathscr{R} \subseteq \{1, 2, \ldots, r\}$ can then be computed as the average code $\bar{\boldsymbol{c}}_{\mathscr{R}} = \frac{1}{|\mathscr{R}|} \sum_{i \in \mathscr{R}} \boldsymbol{c}_i$ (assuming $\boldsymbol{c}_i \geq 0$ and $\|\boldsymbol{c}_i\|_1 = 1$). This corresponds to applying the average pooling approach discussed in Section 2.3.3, to the region $\mathscr{R}$.

Here we focus on ways to define sets of pooling regions $(\mathscr{R}_1, \mathscr{R}_2, \ldots, \mathscr{R}_l)$ that are non-overlapping and span the full image:

$$|\mathscr{R}_i| = \lambda_i r, \quad \sum_{i=1}^{l} \lambda_i = 1, \quad 0 < \lambda_i < 1 \quad \text{and} \quad \forall i, j \in \{1, \ldots, l\} \quad \mathscr{R}_i \cap \mathscr{R}_j = \emptyset, \tag{3.1}$$

so that $\sum_{i=1}^{l} |\mathscr{R}_i| = r$ and $\bigcup_{i=1}^{l} \mathscr{R}_i = \{1, 2, \ldots, r\}$. We call λ_i the *mass coefficients*, as they define how the image area is divided into the l regions. For example, if $l = 2$ and $\lambda_1 = \lambda_2 = \frac{1}{2}$, the image area is equally split between two regions $\mathscr{R}_1$ and $\mathscr{R}_2$. If $\lambda_1 < \frac{1}{2}$ then $\mathscr{R}_2$ covers a larger part of the image, and vice-versa, if $\lambda_1 > \frac{1}{2}$, then the larger part is reserved for $\mathscr{R}_1$.

Our strategy to obtain robust image descriptors consists of two different approaches, expected to be complementary:

1. **Saliency-driven Perceptual Pooling (*SPP*)**. We define the pooling regions by using a saliency operator. This approach captures the perceptual regularities in the scenes, regardless of their exact position in the scene.
2. **Task-driven Spatial Pooling (*TSP*)**. We define pooling regions with a fixed relative position in the image. This approach captures the spatial regularities of the scenes.

The technical details for the two pooling approaches are provided in the Section 3.3.1 and 3.3.2.

3.3.1 Saliency-driven Perceptual Pooling (SPP)

Traditional spatial encodings, designed to capture the spatial regularities in the scenes, partition the image using a regular grid and pool the features in the resulting patches. Instead of imposing an a-priori segmentation, we would like to let visual-structures emerge from the images, regardless of their exact positions in the scene. Specifically, we aim to obtain a segmentation of the image into two regions $(\mathscr{R}_1, \mathscr{R}_2)$, such that $\mathscr{R}_2$ captures the area of the image with a richer informative content, leaving to $\mathscr{R}_1$ the task to collect the statistics of the remaining part. To this end, we propose to compute a saliency map $\boldsymbol{z} \in \mathbb{R}^r$ for each image and use a threshold $\bar{z}$ to segment the image in two regions:

1. $\mathscr{R}_1 = \{1 \leq i \leq r : z(\boldsymbol{x}_i) \leq \bar{z}\}$,
2. $\mathscr{R}_2 = \{1 \leq i \leq r : z(\boldsymbol{x}_i) > \bar{z}\}$,

where $z(\boldsymbol{x}_i)$ is the value of the saliency map at the local descriptor $\boldsymbol{x}_i$. We propose to select $\bar{z}$ so that $\mathscr{R}_1$ and $\mathscr{R}_2$ satisfy the conditions in equation (3.1) for a given value of the mass coefficient λ_1. Note that, due to the conditions in equation (3.1), λ_2 is obliged to take the value $\lambda_2 = 1 - \lambda_1$. For example, if $\lambda_1 = \frac{1}{2}$, then $\bar{z}$ is the median saliency value of the image, while if $\lambda_1 \neq \frac{1}{2}$, the image is asymmetrically split between $\mathscr{R}_1$ and $\mathscr{R}_2$. To compute the saliency map, we tested two different approaches.

Itti's Saliency

One of the most established and most widely known saliency operators is the one proposed by Itti et al. [1998]. In this model the saliency map $\boldsymbol{z}$ of a given image is computed by performing center-surround operations $O_i(c,s) = |C_i(c) \ominus C_i(s)|$, where c and $s = c + \delta$ are two different scales in a Gaussian pyramid, while $\{C_i\}_{i=1}^3$ are three different image channels: an intensity channel C_1, a color channel C_2 and an orientation channel C_3. The responses O_i from the different channels are then normalized and averaged, to get the final saliency score for each pixel. In our experiments we made use of the implementation of Harel [2006].

SIFT Saliency

Instead of using a saliency operator on the raw pixels data, it could be desirable to design a saliency function able to make use of the rich information already encoded in the pre-computed local descriptors. In this way, the salient / non-salient discrimination could be performed directly on the local descriptors that are to be pooled, assuring a higher consistency between the segmentation and the actual image representation used in the pooling step.

A saliency operator that can enable a feature-based saliency estimation is the *AIM* model (Attention based on Information Maximization) [Bruce and Tsotsos, 2005]. Here, the probability of each pixel is locally estimated by non-parametrically fitting a distribution over the RGB values of the image. Since there is not enough data in an image to reliably estimate the joint distribution of the RGB values, the authors propose to make use of Independent

Component Analysis (ICA) [Hyvärinen and Oja, 2000] to turn the three-dimensional joint distribution estimation problem into a set of three independent one-dimensional estimation problems. Specifically, let $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_t\}$ be a set of t image descriptors with dimensionality d (e.g. $d = 3$ for image pixels), sampled from a training set. The goal of ICA is to find a basis $\boldsymbol{A} \in \mathbb{R}^{d \times d}$ and a matrix of components $\mathbf{S} = [\mathbf{s}_1, \ldots, \mathbf{s}_t]^T \in \mathbb{R}^{t \times d}$ such that $\forall\, i \in \{1, \cdots, t\}$, $\boldsymbol{x}_i = \boldsymbol{A}\mathbf{s}_i$ and the coefficients $(\mathbf{s}_i)_j$ are as statistically independent as possible. Once the basis $\boldsymbol{A}$ has been computed, its inverse $\boldsymbol{W} = \boldsymbol{A}^{-1}$ can be used to project new data into the independent components space.

In this work we propose to apply the AIM technique to the image descriptors extracted from the original images. Specifically, we propose to compute the AIM saliency of the low-level SIFT local descriptors that are to be pooled, and use it to output a low-resolution saliency map. Similarly to AIM, after computing the ICA projection $\tilde{\boldsymbol{X}} = [\tilde{\boldsymbol{x}}_1, \ldots, \tilde{\boldsymbol{x}}_r]^T = \boldsymbol{X}\boldsymbol{W}^\top$ of an image $\boldsymbol{X}$ (in our case a matrix of SIFT local descriptors), we make use of the independence assumption to estimate the local density of the j-th dimension of a descriptor i as

$$p\left((\tilde{x}_i)_j\right) = \frac{1}{r}\sum_{k=1}^{r} K\left((\tilde{x}_i)_j - (\tilde{x}_k)_j\right), \tag{3.2}$$

where $K(x) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}x^2\right)$ is a one-dimensional standard normal probability density function. The saliency of the (projected) local descriptor $\tilde{\boldsymbol{x}}_i$ is then computed as:

$$z(\tilde{\boldsymbol{x}}_i) = -\sum_{j=1}^{d} \log p\left((\tilde{x}_i)_j\right) \tag{3.3}$$

and a first saliency map is obtained by computing the responses for all the r SIFT descriptors of the image. Since the SIFT descriptors are computed on a regular grid with a large spacing (e.g., 8 pixels), this procedure results in a low-resolution [1] saliency map, with sharp variations between neighboring points. A smoother map is finally obtained by convolving the initial response with a Gaussian filter, with $\sigma = 0.04 * \max(h, w)$. This value has recently been shown to provide the best results when predicting human fixations with the original AIM model (Figure 8 of [Hou et al., 2012]), and preliminary experiments confirmed it to be a reasonable choice with our setup as well.

In Figure 3.4 we visualize the 128 SIFT Independent Components (as computed using SIFT patches sampled from one training split of the MIT-Indoor-67 [Quattoni and Torralba, 2009] dataset), together with an example of how a SIFT Saliency map is formed. As expected, this saliency operator is taking into account only the textural information provided by the SIFT features, while disregarding other channels, like color and intensity. For example, the grating on the window results to be as salient as the lamp lit on the night table. While this may not be a problem for our scene recognition goal (i.e. a light source might not be more discriminative than a window grating), it might be of limited use for other tasks, like human fixation

1. For our segmentation and pooling goal we don't need a higher resolution map, since the local descriptors are computed with the same resolution (e.g., one every 8 pixels).

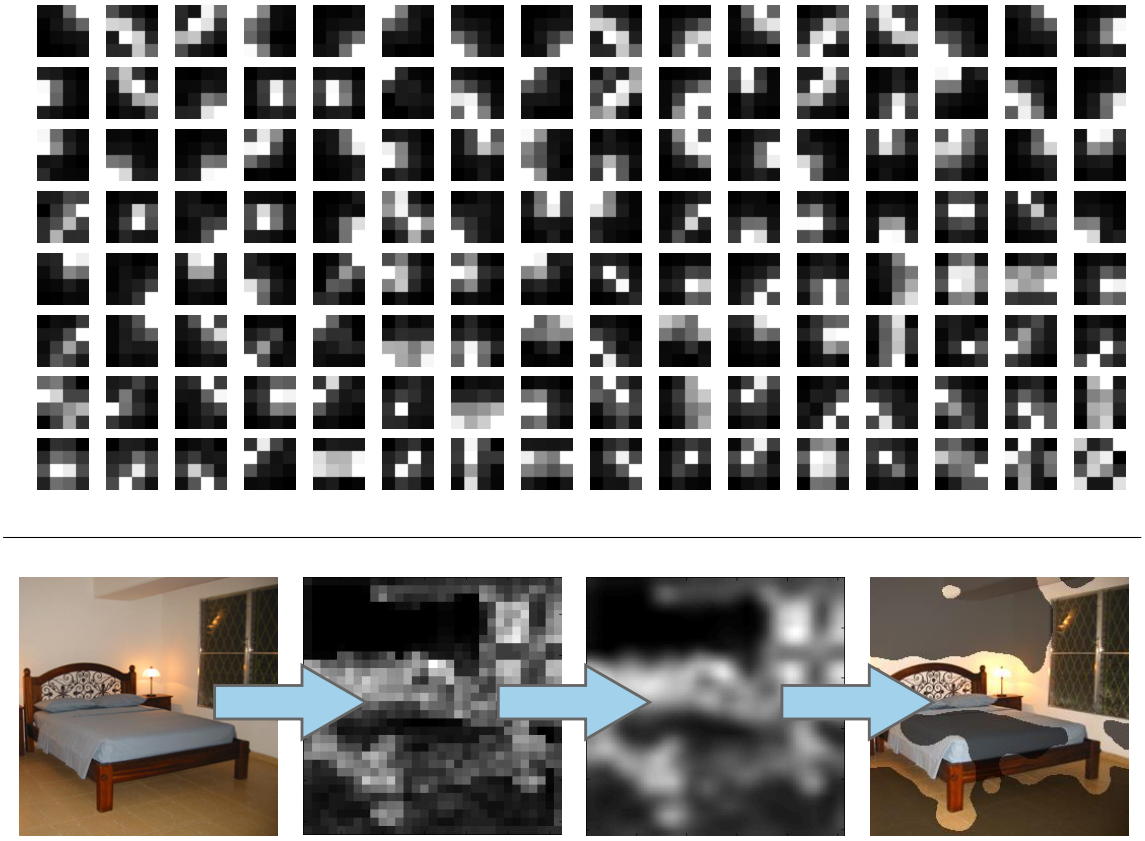

Figure 3.4 – Top: visualization of the 128 SIFT Independent Components, summed over the 8 orientations; white pixels correspond to high ICA (rectified) weights for the gradients in the corresponding area of the SIFT patches. Bottom: Computation of a SIFT saliency map and resulting segmentation using $l = 2$ regions and $\lambda_1 = \lambda_2 = \frac{1}{2}$.

prediction, and we do not claim any biological plausibility for it.

A comparison of the histograms obtained using SPP with Itti's and SIFT saliency is shown in Figure 3.5. In the same Figure we also plot the average number of non-zero visual words in the histograms computed over the salient and the non-salient regions. As it is possible to see, for both Itti's and SIFT saliency, the histograms computed over the salient regions are less peaked, containing a high number of non-zero visual words. On the contrary, the histograms computed over the non-salient regions are peaked around a few active visual words. In other words, the salient regions have a high visual complexity, while the non-salient ones capture more uniform areas that are well described by only a few visual words. In Section 3.4.2 this observation will be empirically confirmed for all the scene recognition datasets.

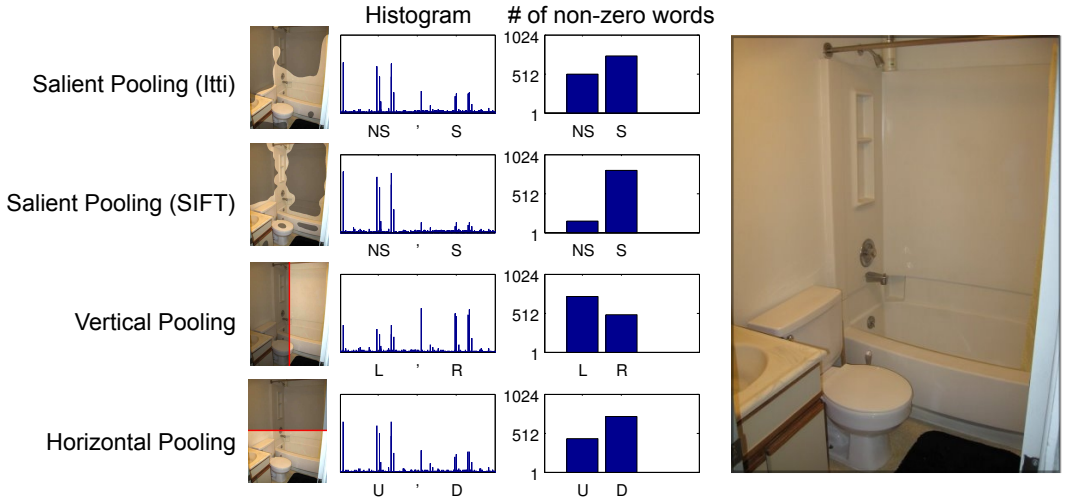

Figure 3.5 – Histograms obtained (with $l = 2$ regions and $\lambda_1 = \lambda_2 = \frac{1}{2}$) using different pooling techniques and number of non-zero visual words in each of the two halves of the histograms: non-salient (NS) and salient (S), left (L) and right (R), up (U) and down (D).

3.3.2 Task-driven Spatial Pooling (TSP)

In the previous Section we defined a pooling strategy conceived to capture perceptually cohesive structures in the images, regardless of their exact position. In this Section we discuss a simple spatial pooling scheme suitable for scene recognition problems.

Indoor scenes, for example, are designed to support human actions and humans have a limited range of spatial mobility. Indeed, humans can usually walk around a room, use objects and appliances within reach, sit on chairs, etc., but they cannot easily move from the floor to the ceiling, or access facilities if they are disposed too low, or too high in the room. This reduces the spatial variability of indoor scenes to lie mostly on the horizontal axis. Due to the effect of gravity, similar considerations may also be drawn for outdoor scenes.

Given this prior, we expect that by pooling features in horizontal bands we will be able to capture the most consistent spatial patterns in scene recognition problems. We instead expect much less robust results by pooling descriptors in vertical bands. To verify this intuition we performed a first set of experiments using only $l = 2$ regions, $\mathscr{R}_1$ and $\mathscr{R}_2$, with $\lambda_1 = \lambda_2 = \frac{1}{2}$:

1. **Horizontal-bands Pooling (*Horizontal*)**. In this settings $\mathscr{R}_1$ is the set of local descriptors lying in the upper 50% of the image, and $\mathscr{R}_2$ is its complement.

2. **Vertical-bands Pooling (*Vertical*)**. In this case $\mathscr{R}_1$ consists of the left-side 50% of the descriptors, and $\mathscr{R}_2$ is its complement.

A visualization of these pooling strategies, with a comparison of the resulting histograms with the ones obtained using SPP is shown in Figure 3.5. The experimental results are presented in Section 3.4.3. Results using $l = 3$ horizontal bands are also provided in Section 3.4.2 and 3.4.3.

3.3.3 Integrating Saliency-driven and Task-driven pooling

Once the saliency-driven and the task-driven image descriptors have been computed, we concatenate them to create a compact image signature that exploits both the perceptual and the spatial consistencies of the scenes. Since our main candidate for the spatial representation is the Horizontal scheme, we only integrate SPP with the Horizontal image descriptor. A multiresolution [Hadjidemetriou et al., 2004] version of our image descriptor is also formed by down-sampling each image by a factor of two, and concatenating the histograms obtained at the two resolutions.

3.4 Experiments

In order to assess the effectiveness of our approach, we perform experiments on the three scene recognition datasets considered throughout this thesis, namely: the MIT-Indoor-67 [Quattoni and Torralba, 2009], the 15-Scenes [Lazebnik et al., 2006] and the UIUC-Sports [Li and Fei-Fei, 2007] datasets. For a complete description of these datasets and their benchmarking procedures please refer to Section 2.1. We compare the performance of our approach with different spatial pooling baselines, such as Horizontal, Vertical and the SPM approach (discussed in Section 2.3.1) with $L \in \{0,1,2,3\}$). We also analyze the relative importance of the components of the proposed approach: the salient and the spatial pooling schemes, as well as the salient and the non-salient regions of the images. In the following we first describe the experimental setup used for all the experiments, we then analyze the various components of the proposed approach, and we finally evaluate the proposed methodology against several pooling baselines on all the three scene recognition datasets.

3.4.1 Experimental setup

In all our experiments we use a common experimental setup. We extract SIFT [2] descriptors on a grid with a spacing of 8 pixels and with a patch size of 16×16 pixels. We also compute multiresolution features, where the spacing and patch size for the down-sampled images are reduced to 6 and 12×12 pixels, respectively. For each resolution, a vocabulary $\boldsymbol{V}$ with $k = 1024$ visual words is obtained by running k-means on a random subset of the training features. The same set of features is used to learn the ICA projection matrix, using the fast-ICA [Hyvärinen and Oja, 2000] algorithm. The intermediate image representation $\boldsymbol{C}$ is then obtained using approximated unconstrained LLC encoding [Wang et al., 2010a] (for a synthetic description of this encoding technique please refer to Section 2.2.2). Since the importance of each visual-word for the reconstruction of a SIFT point is given only by the magnitude of its response, and to avoid cancellation effects with average-pooling, we also perform rectification of the codewords responses, replacing $(c_i)_j$ with $\left|(c_i)_j\right|$.

Since the unconstrained LLC encodings are not normalized, we separately ℓ_1-normalize each

2. In this work we use the implementation of Zhou et al. [2011], made available by the authors.

of the two histograms $\bar{\boldsymbol{c}}_{\mathscr{R}_1}$ and $\bar{\boldsymbol{c}}_{\mathscr{R}_2}$, obtained by average pooling the codes belonging to the two regions $\mathscr{R}_1$ and $\mathscr{R}_2$. This ensures that images with different sizes will contribute to the learning in the same way, and that the two regions will have exactly the same importance in the final representation. This inner-normalization is not performed on the SPM baselines ($L0$, $L1$, $L2$ and $L3$) to avoid emphasizing the histograms computed on small patches; in this case we perform the ℓ_1-normalization on the final vector only. As a similarity measure for all our histograms we make use of the exponential χ^2 kernel [Fowlkes et al., 2004], with γ set to the average pairwise χ^2 distance between the training samples, as in [Gehler and Nowozin, 2009]. The classification is finally performed using SVM [Chang and Lin, 2011], with the regularization parameter tuned using a 5-fold cross-validation procedure. We repeat each experiment on five random training/test splits and we report the average classification accuracy and the standard deviation.

With this setup any single region is represented by a 1024-dimensional histogram. In this way, for example, using a single resolution the standard $L3$ SPM representation results in a 87,040-dimensional descriptor, while a Horizontal+SPP approach with $l = 2$ regions in each part of the representation results in a 4096-dimensional descriptor. Natural baselines for our Horizontal+SPP approach would be the concatenation of the Horizontal and the Vertical image descriptors with $l = 2$ (4096-dimensional) and the $L1$ SPM (5120-dimensional) strategies.

3.4.2 Empirical analysis of the method

Before fully evaluating the proposed pooling approach against other baselines, it is useful to perform a set of preliminary experiments to understand the effect that the salient-pooling approach has on the final image descriptors, the role of the mass coefficients, as well as to compare different ways to integrate the saliency-driven and the spatial representations.

Analysis of the features generated using Saliency-driven pooling

Our first set of experiments consists in analyzing the effect of the saliency-driven pooling techniques on the final image signatures. We quantitatively compare the image descriptors obtained with saliency-driven approaches, with those obtained by the spatial pooling baselines, such as the horizontal and vertical pooling schemes. Specifically, using $l = 2$ and $\lambda_1 = \frac{1}{2}$ for all the representations, we evaluate the following quantities:

1. The number of non-zero visual words in each region (salient/non-salient, top/bottom, left/right) of each representation obtained after the pooling step.
2. The average overlap between salient regions and horizontal/vertical patches (top and left).

The first measure is related to the visual complexity of the area being described: a very complex (part of a) scene is expected to generate a high number of responses to many different visual words, while less complex areas are expected to generate highly peaked histograms, with only

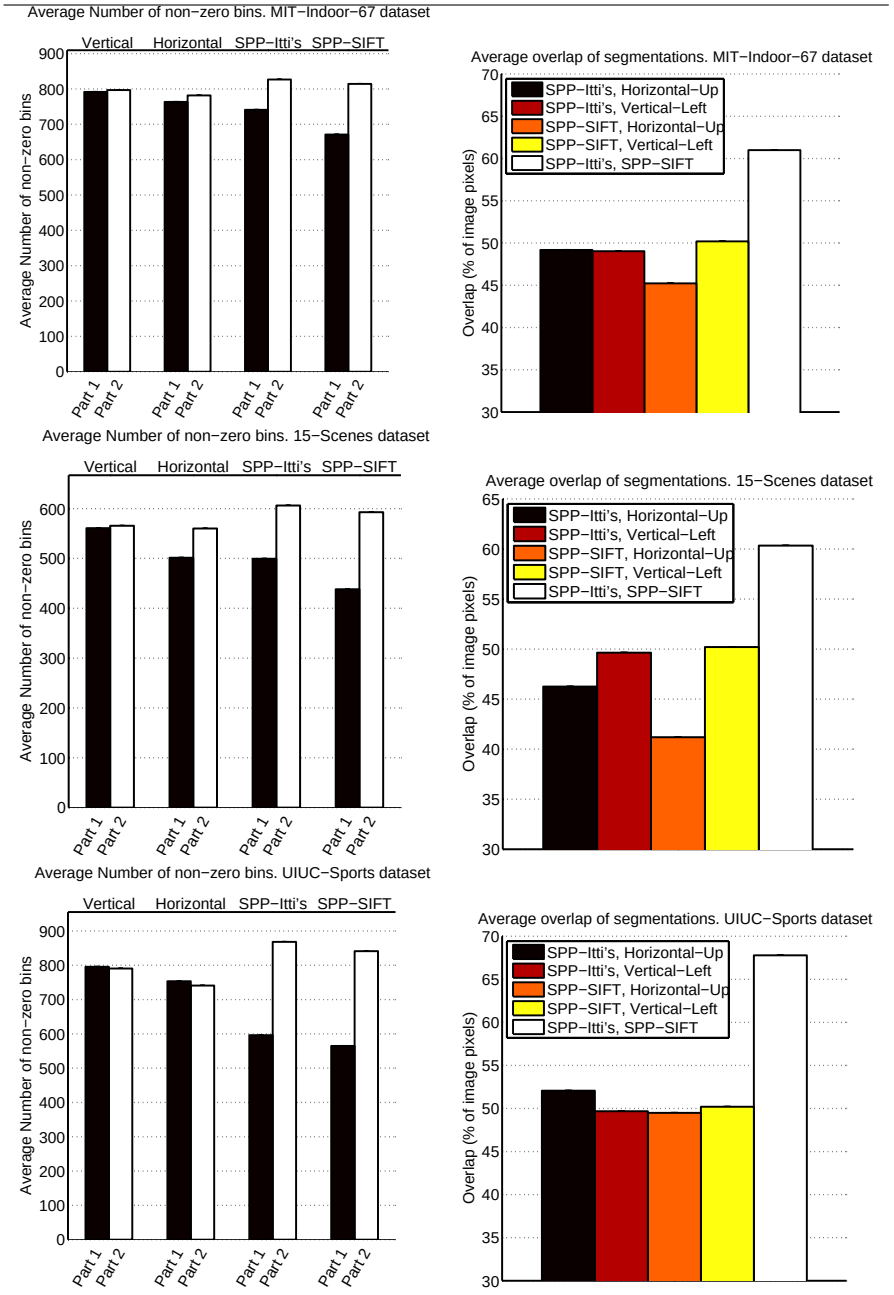

Figure 3.6 – Left: average number of non-zero visual words in each part of the representation, as obtained with different pooling techniques with $l = 2$. For Horizontal pooling Part 1 is the top part of the image, for Vertical pooling Part 1 is the left part of the image, while for SPP Part 1 is the non-salient region. Right: average overlap (in % of the number of pixels) between salient regions and horizontal/vertical patches, compared to the average overlap of the salient regions obtained with the Itti's and the SIFT SPP representations. The plots are obtained with $\lambda_1 = \frac{1}{2}$ on the MIT-Indoor-67 dataset (top), the 15-Scenes dataset (center) and the UIUC-Sports dataset (bottom).

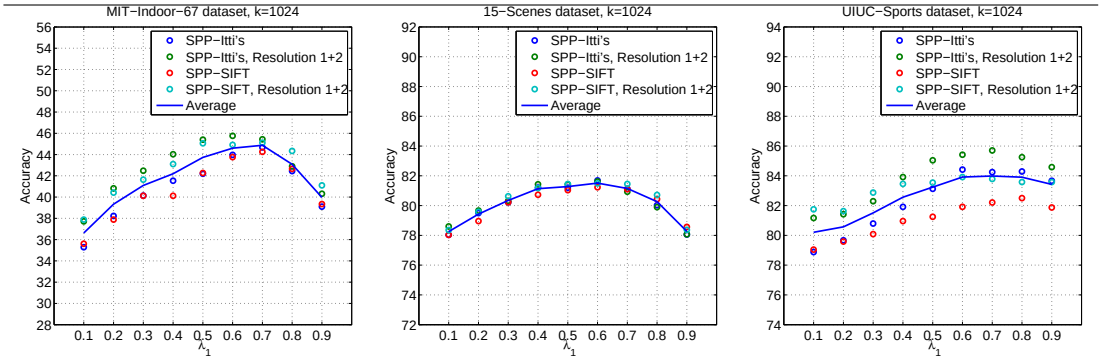

Figure 3.7 – Performance obtained by the saliency-driven pooling approaches when varying the percentage of the image descriptors that are assigned to the non-salient region. The results are provided for single and multiresolution, Itti's and SIFT SPP representations, on the MIT-Indoor-67 (left), the 15-Scenes (center) and the UIUC-Sports (right) datasets. For visualization purposes we also plot the average performance of the four SPP descriptors.

few active visual words. The second quantity is used to evaluate how spatially biased are the salient regions w.r.t. the top/bottom and left/right regions produced by the horizontal and the vertical pooling strategies.

In Figure 3.6 we plot these quantities, as measured on the MIT-Indoor-67 dataset (top), the 15-Scenes dataset (center) and the UIUC-Sports dataset (bottom). As it can be noticed (and similarly to Figure 3.5), on all the considered datasets the saliency-driven pooling approaches produce a representation where the most complex areas of the image (as measured by the number of non-zero visual words) are pooled together in the salient region (Part 2), while the least complex ones end up in the non-salient set (Part 1). This contrasts with the canonical spatial encodings, where the consistency is in the absolute position of the features. In Figure 3.6 we also see that the Itti's and the SIFT saliency operators produce segmentations which overlap for more than 60% of the pixels, thus resulting in quite similar image descriptions. In the same figure we also plot the average overlap of the salient regions (obtained using both the Itti's and the SIFT saliency), with the horizontal/ vertical patches, showing it to be around 50% of the pixels in most of the cases: on average, only half of the salient region overlaps with the upper/lower, or left/right regions. This confirms that the salient pooling captures information that is not spatially biased. Finally, in Figure 3.6 we can also notice that there is a consistency between the relative complexity of a region and its overlap with the saliency region. For example, on both the 15-Scenes and the MIT-Indoor-67 datasets, the average complexity of the upper part of the image results to be slightly lower than the average complexity of the lower part (first two plots on the top of Figure 3.6). Accordingly, also the average overlap of the salient regions with the upper part of the image are reduced (black and red bars in the first two plots on the bottom of Figure 3.6).

Setting the mass threshold

In this Section we analyze the effect of varying the amount of local descriptors that are assigned to each of the two saliency-driven regions, by varying the mass coefficient λ_1. We evaluate the effect of using $\lambda_1 = \frac{1}{2}$ to split the image into the most and the least salient 50% (as in Fornoni and Caputo [2012]), compared to results obtained by varying λ_1 to allow asymmetric splits of the image into regions with a different amount of local descriptors. Specifically, making use of the datasets, the benchmarking protocols and the experimental setup described in Section 2.1, we perform a set of experiments varying λ_1 in $\{0.1, 0.2, \ldots, 0.9\}$.

Figure 3.7 shows the classification accuracy achieved by the Itti's and the SIFT saliency maps on single and multi-resolution features, with different values of λ_1. Although the average performance of the SPP techniques does not excessively oscillate, by using asymmetric splits it is possible to obtain better performance then by simply using a 50%/50% split. Specifically, by assigning 60% to 70% of the image pixels to the non-salient region, the performance of all the saliency-driven representations is consistently incremented for all the datasets. This fact can be explained by separately analyzing the performance of the two components (the salient and the non-salient one) of the SPP representation.

In Figure 3.8 we plot the recognition accuracy obtained when separately using the image signature obtained by pooling single-resolution features only over the salient region, only over the non-salient region, and when concatenating the two representations. As it may be expected, the recognition accuracy of each region increases as the percentage of pixels assigned to it is increased, with the salient region generally performing better for a given amount of pixels. For example, for the Itti's saliency on the MIT-Indoor-67 dataset, the image signature obtained by pooling the local features over the most salient 30% of the image achieves an accuracy close to 35%, while the non-salient descriptor computed over a region with the same area achieves an accuracy lower than 25%. Qualitatively similar results can also be observed for the SIFT saliency and on the other datasets.

The results reported in Figure 3.8 show that salient regions capture visual structures that are more discriminative than those contained in the non-salient ones. Due to this asymmetric behavior of the two image regions, the best recognition performance is always obtained when the non-salient image descriptor is computed using a larger amount of pixels than the salient one. This explains why for all the saliency operators, at both resolutions and on all the datasets, the best performance is always obtained with an asymmetric split assigning more than 50% of the image descriptors to the non-salient region (Figure 3.7 and 3.8). Using this result, we set the mass coefficient λ_1 to 0.6 for all the remaining experiments. Please note that, although this is a reasonable value for both saliency operators and on all datasets (outperforming $\lambda_1 = \lambda_2 = \frac{1}{2}$ used in Fornoni and Caputo [2012]), a better performance may be obtained by tuning λ_1 for each saliency operator and each task separately. In Figure 3.8 it is also possible to note how the best performance is always obtained by combining the salient and the non-salient image representations (except for the SIFT saliency on the Sports dataset). This confirms our

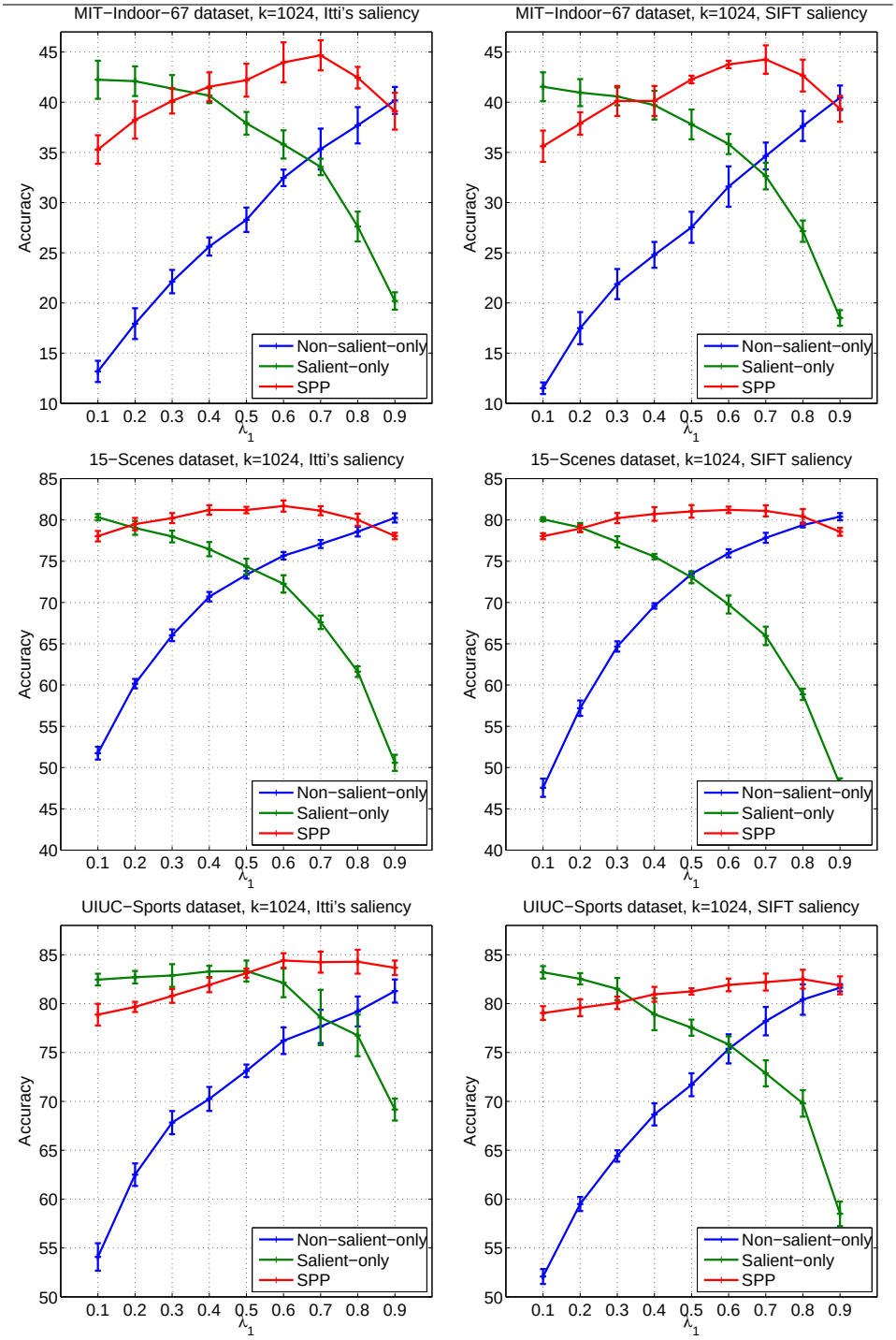

Figure 3.8 – Performance obtained by separately using the single-resolution features pooled over the salient and non-salient region, and when concatenating the two representations. The mass coefficient λ_1 is varied from 0.1 to 0.9 and results are reported for the MIT-Indoor-67 (top), the 15-Scenes (middle) and the UIUC-Sports (bottom) datasets.

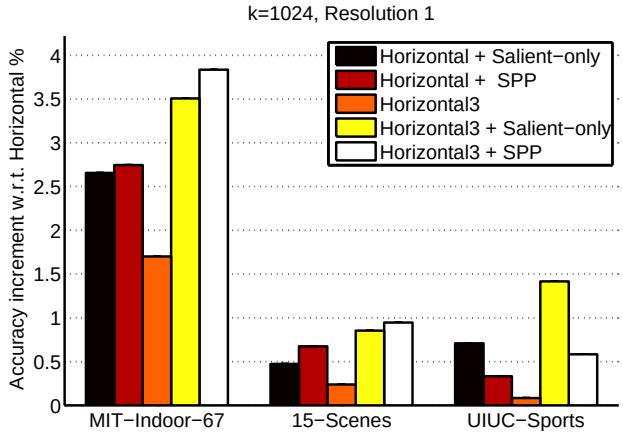

Figure 3.9 – Relative performance of different pooling strategies w.r.t. the Horizontal baseline (with $l = 2$), using single-resolution descriptors on the MIT-Indoor-67 (left), the 15-Scenes (center) and the UIUC-Sports (right) datasets. "Horizontal3" stands for the horizontal bands pooling with $l = 3$ and $\lambda_1 = \lambda_2 = \lambda_3 = \frac{1}{3}$.

hypothesis that for scene recognition problems, both the salient and the non-salient areas are important for classification, and the feature extracted from both regions should be preserved in the final image representation.

Combining Saliency-driven and Task-driven pooling

Our last preliminary study focuses on the combination of the saliency-driven and the task-driven descriptors. According to our previous analysis, the salient-only image descriptor is more effective than the non-salient one, while a representation of the non-salient area is nevertheless important to obtain high recognition accuracies. With respect to this last point, we note that when the saliency-driven representations are combined with a spatial pooling, the visual words extracted from the non-salient areas are already captured and represented by the spatial pooling image descriptor (though without being separately pooled). We thus consider the question whether it is really necessary to explicitly represent these features by computing a description of the non-salient area of the image, when the saliency-driven pooling and the spatial pooling are combined together.

To try to answer this question we measure the relative performance of the combined pooling strategies using the Salient-only and the SPP representation, compared to the performance of the Horizontal baseline (with $l = 2$). The obtained results of are reported in Figure 3.9. While on two out of three datasets a separate representation of the non-salient region further increases the recognition accuracy, the improvement w.r.t. the Horizontal + Salient-only approach is only modest. Given this result we propose to evaluate whether it would be advantageous to replace the explicit representation of the non-salient area with a more fine grained horizontal pooling. Specifically, we evaluate the effect of increasing the resolution at which the spatial information is captured, by using three horizontal regions instead of two (the *Horizontal3*

baseline, computed using $l = 3$ and $\lambda_1 = \lambda_2 = \lambda_3 = \frac{1}{3}$), and the effect of concatenating the Salient-only and the SPP image descriptors to this higher resolution spatial pooling. As it can be noted, a small to medium performance improvement is obtained when increasing the spatial resolution of the horizontal pooling, while a consistently large improvement can be obtained by concatenating this image descriptor with a representation of the salient region (obtaining the Horizontal3+Salient-only image descriptor). As before, a further concatenation of the descriptor computed from the non-salient area (to obtain the Horizontal3+SPP representation) only modestly increases the performance, on two out of three datasets.

In conclusion, this preliminary analysis shows that an alternative approach to the Horizontal+SPP approach consists in using only the salient part of the SPP representation, and combine it with a horizontal pooling with $l = 3$ bands. This does not increase the dimensionality of the final representation, while improving the performance. The two approaches are extensively evaluated and compared to the SPM baselines in the next Section.

3.4.3 Experimental results

In this Section we perform a detailed performance analysis of different pooling methods on all the scene recognition datasets, using both single and multiresolution features. Specifically, we compare the recognition accuracy obtained by the standard SPM approaches at different levels ($L0$, $L1$, $L2$ and $L3$), the horizontal (Horizontal), the vertical (Vertical) and the horizontal + vertical (Horizontal+Vertical) spatial partitions, the saliency-driven pooling (SPP, using both Itti's and SIFT saliency) and the concatenation of the saliency-driven image descriptors to the descriptors obtained using a horizontal partition (Horizontal+SPP). As discussed in Section 3.4.2 we also consider the horizontal partition with $l = 3$ (Horizontal3) and its concatenation with the salient-only pooling approach (Horizontal3+Salient-only).

Results on MIT-Indoor-67

The results of our benchmark for the MIT-Indoor-67 dataset are reported in Figure 3.10. From the Figure we can note that there is a large difference in performance between the horizontal and the vertical pooling strategies (+14.8% relative improvement, using single resolution features). This is not surprising considering the spatial structure of the problem. The SPP strategies perform better than $L0$ and the vertical pooling (up to +9.2% relative improvement, using single resolution features), but still worse than the horizontal one. On the other hand, when combined with horizontal pooling, the saliency-driven approaches always outperform Horizontal3, Horizontal+Vertical and $L1$, $L2$, $L3$ baselines, with the best results obtained by using the Horizontal3+Salient-only image representation. In this case, the absolute performance improvement w.r.t. the best performing spatial pooling (e.g. the $L2$ spatial pyramid) is around 2% using both single and multiresolution features. Indeed, the 4096-dimensional Horizontal3+Salient-only Itti's image representation at a single resolution performs only 0.5% worse (in absolute terms) than the best-performing spatial descriptor

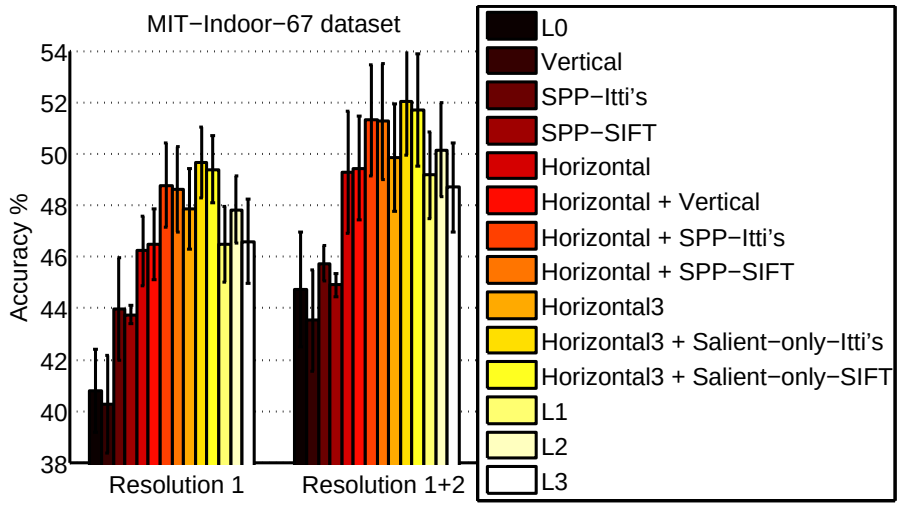

Figure 3.10 – Performances of the different pooling strategies on the MIT-Indoor-67 dataset.

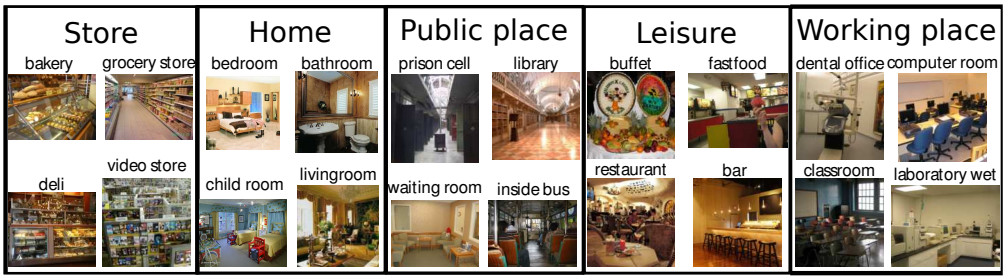

Figure 3.11 – Example of images from some of the classes in each scene group of the MIT-Indoor-67 dataset.

using two resolutions, namely the 43,008-dimensional *L2* feature. On this dataset the Itti's and the SIFT saliency-driven pooling schemes perform similarly, with the former slightly outperforming the latter.

Results on indoor scene groups

Quattoni and Torralba [2009] divided the classes of the MIT-Indoor-67 dataset into five big scene groups: *Store*, *Home*, *Public place*, *Leisure* and *Working place*. Each of these scene groups contains images from 11 to 15 classes, sharing semantic and/or perceptual properties. In Figure 3.11 we report examples of images extracted from some of the classes in each scene group. As it can be noted, images from classes belonging to the same group tend to be visually more similar than images from classes belonging to different groups, while a large visual variability is still present within each group. Inspired by this observation we performed two additional sets of experiments meant to stress the robustness and discriminative power of the proposed approach:

Task-1 (High-level classification). In this task we measure the performance obtained when

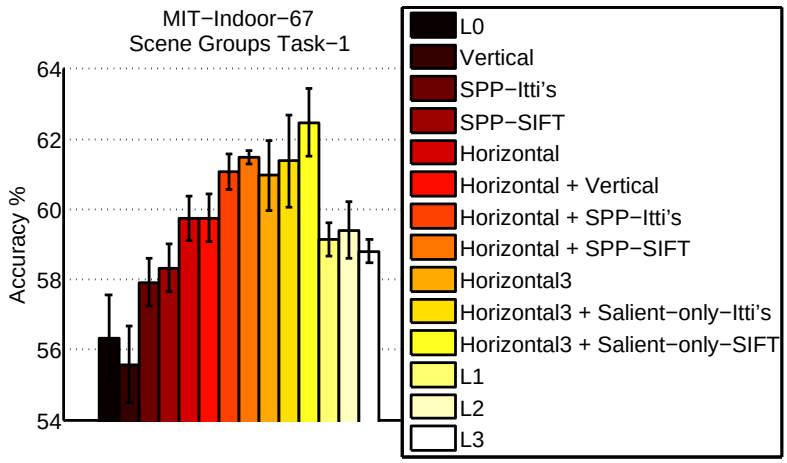

Figure 3.12 – Accuracy obtained when using single-resolution descriptors to classify the images from MIT-Indoor-67 with respect to which of the five scene groups (Store, Home, Public place, Leisure, or Working place) they belong to.

classifying the images according to which scene group they belong to. This is a five-class problem where the algorithm has to assign an image to one of the five scene groups: Store, Home, Public place, Leisure, or Working place. Given the very rich diversity of the images composing each scene group, this benchmark can be used to measure the robustness of an image representation when dealing with scene classes with a pronouncedly high intra-class variability.

Task-2 (Fine-grained classification). In this task we measure the performance when classifying images w.r.t. the sub-classes in each of the five scene groups. For this task we thus perform a different multi-class experiment to separate the classes in each of the five scene groups. Given the above-average visual similarity of the classes within each scene group, this benchmark is useful to measure the discriminative ability of an image representation when having to discriminate between tightly related classes.

Following the benchmarking protocol of the original MIT-Indoor-67 task, both for Task-1 and Task-2 we select 80 images for training and 20 for testing for each of the sub-categories of each scene group. The training and testing sets of a given scene group are then constructed by joining the training/testing sets of the categories of interest. As before, each experiment is repeated five times on five different training/testing splits and we plot the average accuracy and the standard deviation measured on the five splits.

The results obtained by using single-resolution descriptors on Task-1 are reported in Figure 3.12. As it may be expected this is a very challenging task. Indeed, even if it is only a five-class problem with a number of training images per class that is more than 10 times higher than in the regular MIT-Indoor-67 task (since each scene group is composed of images from at least 11 sub-categories), the accuracies achieved by all the pooling methods are quite low. Once again, the horizontal pooling scheme (Horizontal) seems to be more robust than the vertical pooling (Vertical). However, differently from the original MIT-Indoor-67 task, none of

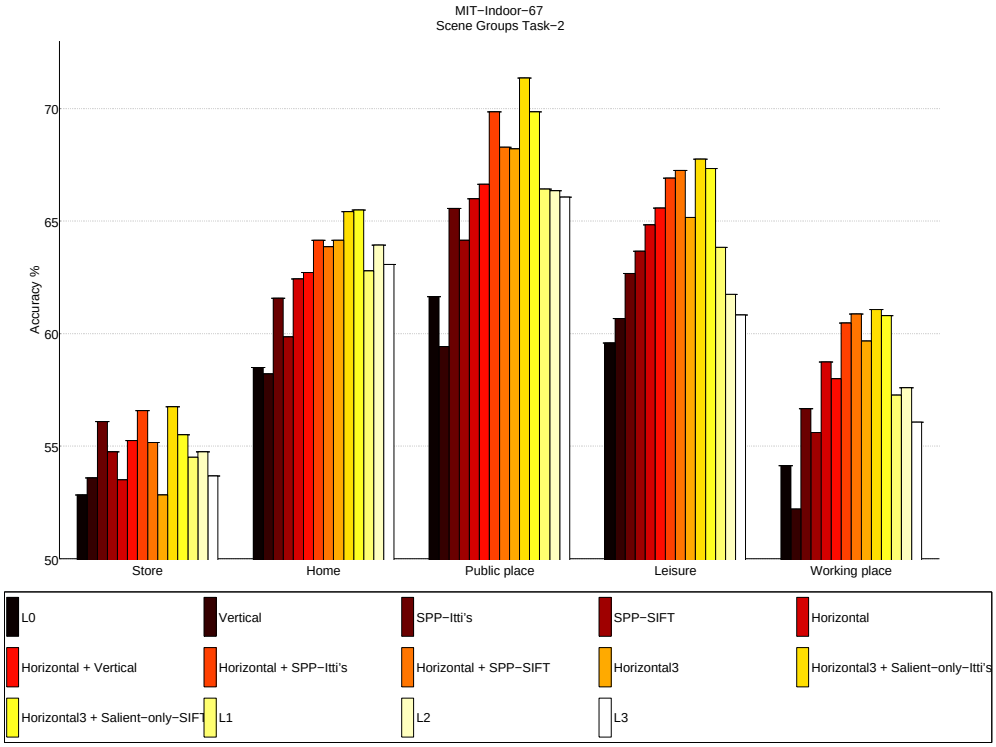

Figure 3.13 – Analysis of the performance of the spatial and saliency-driven pooling approaches on the five scene groups of the MIT-Indoor-67 dataset.

the spatial pyramid approaches is able to outperform the simple horizontal partitioning. This is probably because the large intra-class spatial variability of the scene groups strongly limits the usefulness of rigid and fine-grained spatial partitioning schemes. On this task the SIFT saliency consistently outperforms the one by Itti, while as for the regular MIT-Indoor-67 task, the performance of the SPP approaches lie in between the performance of vertical pooling and the performance of the horizontal pooling. As before, all the top recognition accuracies are obtained by combining a horizontal pooling approach (Horizontal, or Horizontal3) with a saliency-driven one (SPP, or Salient-only, respectively). This confirms the robustness of this type of representations when dealing with problems with a very high intra-class variability.

In Figure 3.13 we report the performance of different algorithms when addressing Task-2, requiring to finely classify images within each of the scene groups. For clarity of visualization in this plot we do not report the standard deviations. As previously discussed, given the above-average visual similarity of the classes in each scenes group, this task is useful to measure the discriminative power of a given representation when dealing with problems with a high inter-class similarity. We note a considerable variability in the difficulty of finely classifying images in different scene groups, with stores being the most difficult to tell apart, and public places being the easiest ones. Still, for all the scene groups, from the easiest to the hardest, the best performance is consistently obtained by combining the horizontal pooling approaches (Horizontal, or Horizontal3) with a saliency-driven pooling (SPP, or Salient-only, respectively).

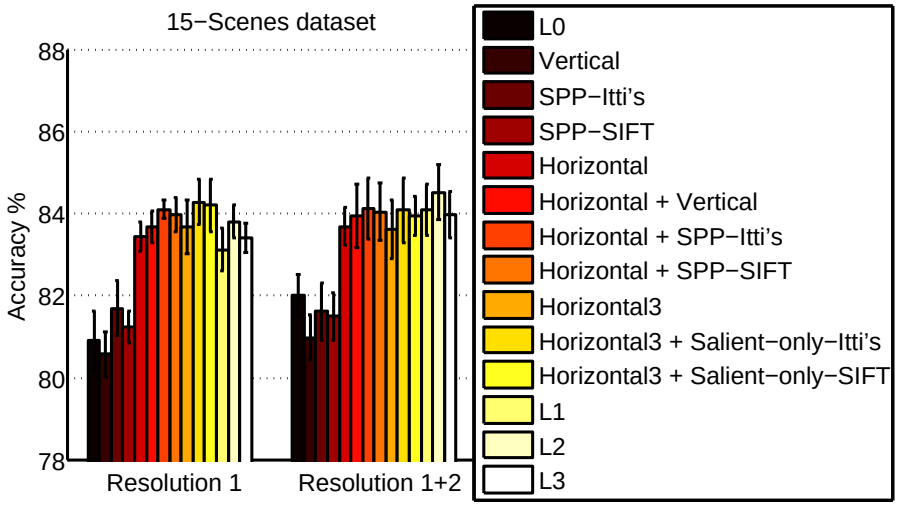

Figure 3.14 – Performances on the 15-Scenes dataset.

It is also interesting to note that on the most difficult scene group (namely the "Store" group) the SPP-Itti's alone outperforms all the spatial pooling approaches, from Horizontal to $L3$. Moreover the same SPP-Itti's image descriptor performs very well also on the easiest scene group (namely the "Public place" group), where the relative performance improvement of the combined representations (Horizontal/Horizontal3 + SPP/Salient-only) seem to be particularly marked. The good performance of the saliency-driven pooling approaches on this group may be explained by looking at the segmentation masks obtained by using Itti's saliency on this scene group, in Figure B.1 of Appendix B. As it can be seen, some of the groups, like "Museum", or "Cloister", or "Library" contain very salient and specific visual structures (such as artworks, columns, or bookshelves) that are easily captured by the saliency-driven pooling schemes. The consistently good performance of the saliency-driven pooling approaches on all the scenes group in this benchmark confirms the robustness of this type of representations when dealing with fine-grained indoor scene recognition problems, with high inter-class similarity.

Results on other scene categorization tasks

As discussed before, our approach is well suited to address the high intra-class variabilities and inter-class similarities of indoor scenes. In this Section we are going to verify experimentally how well this approach generalizes to other scene recognition problems. To this end, we will make use of the 15-Scenes dataset and the UIUC-Sports dataset, described in Section 2.1.

For the 15-Scenes dataset we followed the standard benchmarking protocol, which consists in randomly selecting 100 training images per class and using the remaining ones for the test. For the UIUC-Sports dataset the standard procedure consists instead in selecting 70 images per class for the training set and 60 for the test set. As before, we repeat each experiment five times and we report the average classification accuracy, in Figure 3.14 for the 15-Scenes dataset and in Figure 3.15 for UIUC-Sports.

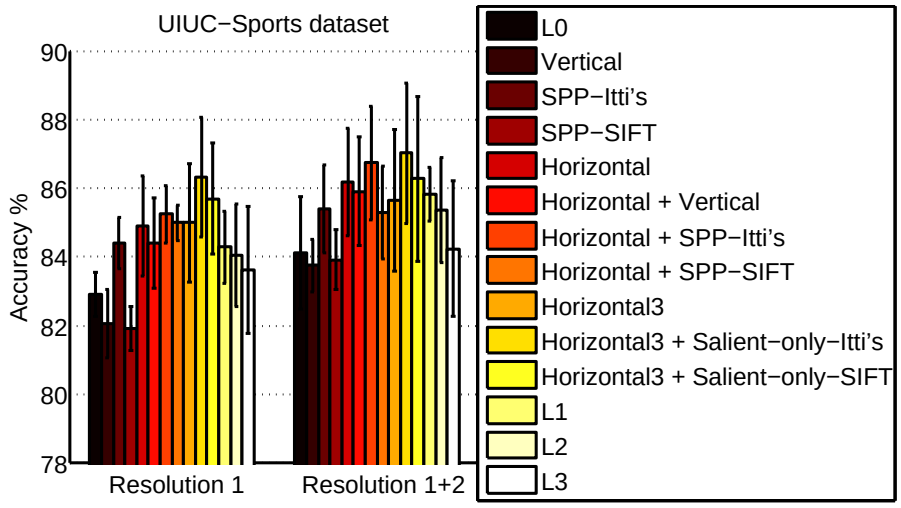

Figure 3.15 – Performances on the UIUC-Sports dataset.

The single-resolution results on the 15-Scenes dataset are consistent with the ones obtained on the MIT-Indoor-67 dataset, although with lower absolute performance gains. On the other hand, on this dataset the performance advantage of the Horizontal+SPP approaches vanishes when using multiresolution features, and the best performance is achieved by the $L2$ spatial pyramid approach. On the UIUC-Sports dataset the SPP-Itti's representation alone seems to be particularly effective, with performance comparable to those of the higher-dimensional spatial pyramid approaches ($L1$, $L2$ and $L3$). Due to the high standard deviations, the performance gains do not seem to be statistically significant on this benchmark. Still, large performance gains are observed when using horizontal pooling approaches (Horizontal, or Horizontal3) in combination with the Itti's saliency-driven representations (SPP, or Salient-only, respectively). On this dataset, the SPP-SIFT image descriptor seems to be performing consistently worse than the one based on Itti's saliency. One possible explanation might be that the former operator only takes into account the textural property of the scene, which might not be useful for consistently separating the scene foreground (e.g., a group of people rowing), from a highly textured natural background (e.g., vegetation, water and buildings). It should also be noted that although this dataset is very different from the MIT-Indoor-67 and the 15-Scenes databases, the top-performing representations always make use of a saliency-driven pooling approach. This outcome is consistent with the results obtained on all the considered tasks for the MIT-Indoor-67 dataset and for the single-resolution features on the 15-Scenes dataset, confirming the advantage of our image representation w.r.t. the traditional spatial pooling schemes.

In Table 3.1 we compare the performance of our image descriptors to those obtained by other recently proposed approaches. Although in the very recent literature there are a few methods that achieve higher recognition accuracies on some of the datasets, they all make use of image descriptors that are from one to two orders of magnitude larger than the ones proposed in this work. Indeed, if we constrain the comparison only to representations having

Method	Dim.	MIT-Indoor-67	15-Scenes	UIUC-Sports
Quattoni and Torralba [2009] (Gist)	384	25.05	-	-
Wu and Rehg [2011] (PACT)	1,302	36.88	83.88	78.25
Kwitt et al. [2012]	1,407 (up to)	44.0	82.3	83.0
Parizi et al. [2012]	3,200	37.93	78.6	-
Juneja et al. [2013] (BoP)	16,750	46.10	-	-
Yang et al. [2009] (Sparse-Coding)	21,504	-	80.4	-
Juneja et al. [2013] (LLC)	42,000	53.03	-	-
Li et al. [2010] (Object-Bank)	44,604	37.6	80.9	76.3
Singh et al. [2012]	70,350	38.1	-	-
Xie et al. [2014] (FV)	65,536	61.22	-	-
Doersch et al. [2013]	67,000	**64.03**	-	-
Juneja et al. [2013] (FV)	204,800	60.77	-	-
Sun and Ponce [2013]	517,230	51.40	**86.08**	86.40
Horizontal + SPP-Itti's	4,096	48.81	84.12	85.25
Horizontal3 + Salient-only-Itti's	4,096	49.70	84.30	86.33
MultiR. Horizontal + SPP-Itti's	8,192	51.34	84.12	86.75
MultiR. Horizontal3 + Salient-only-Itti's	8,192	52.07	84.09	**87.04**

Table 3.1 – Performance comparison with previous approaches using a single image descriptor. For each approach we also report the dimensionality of the representation used.

a dimensionality comparable to that of our approach (top part of the table), it is possible to see that our proposed image descriptors obtain the best performance on all the datasets.

3.5 Discussion

In this Chapter we introduced SPP, a saliency-driven pooling technique able to capture perceptually coherent structures, independently from their positions in the scene. We made use of the well-known Itti's saliency operator and proposed a SIFT-based saliency function, operating on the rich textural information encoded in the local descriptors. The saliency maps are used to segment each image into the most and the least salient areas. Local image features are subsequently pooled from each region, and the resulting descriptors concatenated. By combining this image representation with a simple horizontal-bands spatial descriptor, we obtain very compact image signatures achieving competitive performances on all the considered datasets.

Amongst the two considered saliency operators, Itti's saliency resulted to be the most robust for the considered scene recognition tasks. This may be due to the fact that SIFT saliency takes into account only the textural information encoded in the SIFT features, disregarding other channels, like color and intensity. For both Itti's and SIFT saliency, the salient area of the image is shown to be on average more complex than the non-salient one (as measured by the number of non-zero visual words in each descriptor). Performance-wise the salient area resulted also to be the most discriminative one, being particularly effective to recognize sports scenes. Still, the best recognition performance was always obtained by combining the representations

extracted from the two regions (salient and non-salient). When integrating the two regions, due to their asymmetric discriminative contents, it becomes important to tune the saliency threshold. Differently from Fornoni and Caputo [2012], we considered saliency thresholds between 0.1 and 0.9 and showed that the scene recognition accuracy can be consistently improved by using thresholds larger than 0.5. Thresholds lower than 0.5 never resulted in performance improvements. Intuitively, the asymmetric discriminative power of the two regions needs to be balanced by assigning larger portions of the image to the non-salient region and lower-portion of the image to the salient one.

Our final image signatures are constructed by combining the representations obtained using the proposed SPP pooling and a simple spatial pooling. The resulting image signatures are up to one order of magnitude more compact than the ones obtained by using popular SPM encodings [Lazebnik et al., 2006]. The proposed signatures are particularly effective on the most difficult scene recognition problems, outperforming the best SPM encodings on two of the three considered datasets. The robustness of the representation is also proved on the challenging scene groups classification task (Task-1), and when solving the most difficult fine-grained tasks (Task-2), such as categorizing stores scenes.

Compared to other approaches using representations with a similar dimensionality (top part of Table 3.1), our image signatures obtain the best performance on all the datasets. Significantly higher performances are obtained only by approaches using very high-dimensional image signatures, as the ones obtained using Fischer Vector encodings (FV) [Perronnin et al., 2010]. Please note that, while we have chosen to use the LLC encoding [Wang et al., 2010a], the SPP pooling technique proposed in this Chapter is not bound to any specific feature encoding technique. An interesting research direction would thus be to apply SPP to Fischer vector encodings. While this would result in much larger descriptors, defeating our quest for compact representations, it may also result in image signatures achieving state of the art scene recognition performance. Another direction that would be interesting to investigate is related to the saliency operators. For example, by combining the low-level channels used by Itti's saliency, with the rich textural information exploited by the SIFT saliency, we might obtain more robust segmentations, specifically adapted to our task. Other saliency operators could be considered as well.

With respect to the goals stated at the beginning of this thesis, the main limitation of the method proposed in this Chapter lies in the classification step. Indeed, we made use of a performing, but computationally expensive, exponential χ^2 kernel SVM. In order to address this problem, in the next Chapter we propose a multi-component classifier, able to achieve results competitive with the ones obtained by kernelized SVMs, using only a fraction of the computational resources required by the latter.

4 ML3 - A Multiclass Latent Locally Linear SVM algorithm

Current state of the art scene recognition algorithms either make use of computationally expensive non-linear kernel classifiers, or require very high-dimensional non-sparse image representations. The most prominent representative of the first category of approaches is the kernelized Support Vector Machine (SVM), a non-linear classification algorithm that has been extensively used to deliver state of the art performance on many problems. Unfortunately, the practical use of the SVM classifier is constrained by its training complexity, which grows super-linearly with the number of training samples. Moreover, its memory footprint and testing complexity grows linearly with the number of support vectors. In order to retain the performance of non-linear kernel classifiers, without significantly increasing the size of the representation, a growing, promising alternative is represented by multi-component classifiers [Felzenszwalb et al., 2010; Wang et al., 2011b; Gu et al., 2012; Hoai and Zisserman, 2013]. The approaches belonging to this class are characterized by their ability to learn a set of sub-categorical models for each class (named *components* of the classifier) and to select the most suitable one(s) to classify each sample. Their training complexity may not grow super-linearly with the number of training samples, while their testing complexity and memory footprint do not depend on the number of training samples.

In this Chapter we propose a new multi-component classifier, based on a latent locally linear SVM formulation. The proposed classifier makes use of a set of linear models that are locally linearly combined using sample and class specific weights. Thanks to the latent formulation, the mixing coefficients are modeled as latent variables. We allow soft combinations and we also provide a closed-form solution for their estimation. This novel formulation allows to learn the classifier components in a principled and efficient way, using a CCCP optimization procedure.

An extensive empirical evaluation on ten standard UCI machine learning datasets, three characters and digit recognition databases and one large binary dataset, show the advantages of the proposed formulation over previously proposed multi-component algorithms. Experiments on the three scene recognition datasets considered throughout this thesis [Quattoni and Torralba, 2009; Lazebnik et al., 2006; Li and Fei-Fei, 2007] prove the proposed algorithm to

be able to partially fill in the gap between linear classifiers and non-linear kernel classifiers, using a fraction of the computational resources required by the latter.

4.1 Introduction

As discussed in the previous Chapters, scene recognition is a difficult problem characterized by a combination of high-intra class variability and high inter-class similarity. Moreover, for a range of scene recognition applications, such as automatic image annotation and robot vision, efficiency of the recognition system is a fundamental requirement. In Chapter 3, we proposed to tackle these problems by designing a compact bottom-up image representation, able to discover visual structures regardless of their positions in the scene. Another important way to address such problems lies in the design of efficient classification algorithms capable of representing classes with a rich sub-categorical structure [Zhu et al., 2012]. Over the last 15 years, kernelized Support Vector Machines (SVMs) have become the de facto standard to address difficult classification problems, requiring to learn complex non-linear decision boundaries. Still, kernelized SVMs (and learning with kernels in general) do not scale well with the number of samples. Linear SVMs, on the other hand, have a training complexity that is linear w.r.t. the number of training samples [Joachims, 2006], and a testing complexity that depends only on the dimensionality of the data. Furthermore, when coupled with very high-dimensional representations, as the ones obtained using Fisher Vectors encodings [Perronnin et al., 2010], Linear SVMs have been shown to provide very competitive performance. Unfortunately, high-dimensional image representations greatly increase the training/testing complexity and the memory footprint of the algorithm, to the point that the training data itself may not fit into memory anymore [Chatfield et al., 2011]. On low-dimensional representations, the performance of Linear SVMs is often disappointing.

To try to address these issues, multi-component SVM-based methods have received increasing attention, both in the kernel learning community [Gönen and Alpaydin, 2008] and in the linear learning community [Zhang et al., 2006; Yu et al., 2009; Felzenszwalb et al., 2010; Ladicky and Torr, 2011; Wang et al., 2011b; Hoai and Zisserman, 2013]. A key feature of such methods is the ability to exploit the structure of the data by learning specific models in different zones of the input space. Performance-wise, when coupling multi-component methods with infinite dimensional kernels, the improvement over non-local versions of the algorithms is usually relatively small [Gönen and Alpaydin, 2008], because the boundary is already flexible enough to separate any training set. However, when combined with linear classifiers they can lead to large improvements, thanks to the increased flexibility of the separation surface between the classes [Gönen and Alpaydin, 2008].

The work presented in this Chapter contributes to this research thread. Our focus is on enhancing linear algorithms to obtain the complex decision functions traditionally given by kernels. We propose a new multi-component learning algorithm based on a latent SVM formulation. For each sample and class, during training as well as during testing, our algorithm

automatically selects a different weighted combination of linear models (named *components* of the classifier). The sample and class specific weights are treated as latent variables of the scoring function [Felzenszwalb et al., 2010] and are obtained by locally maximizing the confidence of the class model on the sample. As opposed to previous methods, we do not require a two-stages formulation, i.e. our approach does not require to first learn an encoding-based representation using a reconstruction (or soft-assignment) technique and then learn a linear SVM on this representation, nor any nearest-neighbor search. Our algorithm is trained in a winner-take-all competitive multi class fashion: each class tries to maximize its score on each sample by using an optimal combination of components, competing with the others in the training process. Moreover, compared to standard latent SVM implementations, our formulation allows to use soft combinations of components, where the sparsity of the combinations, and thus the smoothness of the solution, is controlled using a p-norm constraint. The solution of the p-norm constrained score maximization problem is shown to be efficiently computable in closed-form, and using this analytic solution we obtain a simple prediction rule in which the local weights do not need to be explicitly computed. We call our method *Multiclass Latent Locally Linear Support Vector Machine* (*ML3*).

Experiments on real and synthetic data illustrate how ML3 behaves as the p-norm constraints and the number of components are varied. We also compare its performance and speed to previously proposed approaches, on ten UCI machine learning datasets [Frank and Asuncion, 2010] (for the binary case), three character recognition databases (MNIST [LeCun et al., 1998], USPS [Hull, 1994] and LETTER [Frank and Asuncion, 2010]) and a large dataset with more than 500,000 samples [Joachims, 2006]. Results consistently show the value of our method. Finally, using the SPP representations introduced in Chapter 3, the ML3 algorithm is evaluated on the three scene recognition datasets considered throughout this thesis.

An outline of this Chapter is as follows. In Section 4.2 we discuss how the proposed classifier relates to the other multi-component and monolithic approaches, as introduced in Section 2.4.2. In Section 4.3 we provide more details about the most important works that will serve as building blocks for the approach presented in this Chapter. In the first part of Section 4.4 we define locally linear functions and we introduce a simple encoding scheme specifically devised for this class of functions. Using this encoding as a motivation, the second part of Section 4.4 introduces the ML3 algorithm, discusses its properties, and its optimization procedure. In Section 4.5 we use ML3 to generate explicit feature maps and to provide several visualizations of the internal functioning of the algorithm, using two synthetic datasets. In Section 4.6 we report additional experiments on synthetic data showing the behavior of ML3 when varying the parameter p. In Section 4.7 we show the results of benchmarking ML3 against other approaches, analyzing again the behavior of the algorithm w.r.t. its hyper-parameters. We conclude in Section 4.8, pointing out some possible future avenues for research.

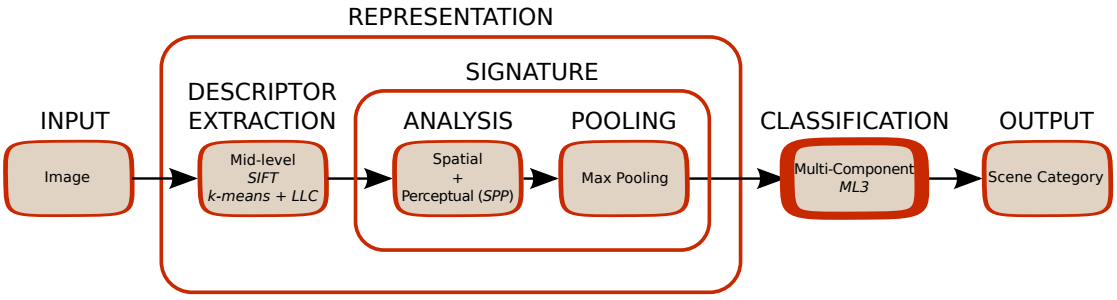

Figure 4.1 – The scene recognition pipeline proposed in this Chapter. The component of the pipeline related to the main contribution of this Chapter is highlighted by a thick border.

4.2 Related works

As in the previous Chapter, the works related to our approach are presented by instantiating the scene recognition pipeline introduced in Chapter 2 for the method discussed here. A visualization of the specific scene recognition pipeline used in this Chapter is reported in Figure 4.1. As it can be seen, the scene recognition pipeline used in this work is very similar to the one used in Chapter 3. However, since this Chapter focuses on locally linear classifiers, we replace the average-pooling step with a max-pooling approach, which has been shown to be more suitable for linear classifiers (see Section 2.3.3).

With respect to this pipeline, the main contribution of this work is related to the classification step. As it is often the case, the classifier presented in this Chapter is related to many of the classes of approaches described in Section 2.4.2. It is related to *local classifiers*, in that, similarly to Cheng et al. [2010] and to Segata and Blanzieri [2010], the algorithm performs a (hard to soft) clustering of the input space, assigning each sample to one (or more) components of the model. It is related to *ensemble methods*, since, similarly to the adaptive mixtures of local experts of Jacobs et al. [1991], our model learns non-linear functions as local linear combinations of linear ones (with the definition of locality used by Jacobs et al. [1991]). It is related to the class of *manifold learning methods*, since similarly to Yu et al. [2009] and Zhang et al. [2011], the scoring function used by our algorithm can be justified from a function approximation point of view (see Section 4.4.1). It is related to *multi-hyperplane classifiers*, as similarly to adaptive multi-hyperplane machines [Aiolli and Sperduti, 2005; Wang et al., 2011b] and existing Latent SVM implementations [Felzenszwalb et al., 2010] it assigns a given query sample the hyperplane (or the linear combination of hyperplanes) producing the maximal score for that sample. Our model can even be related to *non-linear monolithic classifiers* such as kernelized SVMs, in that (as shown in Section 4.5) during the classification step it implicitly projects the samples to a high-dimensional feature space where the linear separability of the samples is increased. Finally, it can also be related to *nearest-neighbor classifiers* since, as it will be discussed in Chapter 5, under certain conditions it can be regarded as a way to learn a representative set of prototypes to be used in a modified NN algorithm.

While being related to many classes of approaches, the method proposed in this Chapter is also unique from many perspectives. Differently from the local classifiers and the manifold approaches, ML3 does not use any unsupervised method (such as the Nearest Neighbor distance, or k-means clustering) to pre-assign a sample to one or more components. Indeed, rather than using only the distribution of the sample points, the sample-to-component coefficients of ML3 take also into account the supervised information provided by the labels of the problem and are obtained by directly minimizing the classification error. With respect to ensemble methods [Jacobs et al., 1991; Gönen and Alpaydin, 2008], the objective function of ML3 is simpler, as it does not make use of any additional gating function (with its additional parameters). Indeed, the only parameters of the model are the linear components, while the sample-to-component coefficients are efficiently computed using a closed-form expression. Moreover, no coefficients at all need to be explicitly computed during testing. Hence, as we will show in Section 4.7, ML3 achieves state of the art performances, with efficient training and testing procedures. Finally, as opposed to the multi-hyperplane classifiers and the latent SVM implementations in Wang and Mori [2009] and Felzenszwalb et al. [2010], ML3 allows for soft combinations of the linear components, resulting in smooth decision boundaries.

This Chapter is based on the work presented in Fornoni et al. [2013].

4.3 Preliminaries

This Section describes in more details the works on which the approach proposed in this Chapter is based, namely: the *Latent SVM* framework [Yu and Joachims, 2009], the *Constrained Concave Convex Procedure (CCCP)* [Yuille and Rangarajan, 2003; Smola et al., 2005; Sriperumbudur and Lanckriet, 2012] and the *Locally Linear SVM* (*LLSVM*) classifier [Ladicky and Torr, 2011]. These works constitute the building blocks that will be used to construct the ML3 algorithm presented in this Chapter.

4.3.1 Latent SVM

Latent SVMs were initially motivated and introduced in the field of computer vision to solve object detection [Felzenszwalb et al., 2008] and action recognition [Wang and Mori, 2009] tasks. They were subsequently adapted to address general structured prediction problems in Yu and Joachims [2009].

Suppose we are given a set of training examples $(\boldsymbol{x}_i, y_i) \in \mathscr{X} \times \mathscr{Y}$, where $i = \{1, \dots, n\}$, $\mathscr{X} \subseteq \mathbb{R}^d$ is the input space and $\mathscr{Y} = \{1, \dots, c\}$ is the output (and the decision) space. A latent SVM makes use of decision functions of the form [Yu and Joachims, 2009]

$$\begin{aligned} \hat{y}_i(\boldsymbol{w}) &\triangleq \arg\max_{y \in \mathscr{Y}} s_{\boldsymbol{w}}(\boldsymbol{x}_i, y), \\ s_{\boldsymbol{w}}(\boldsymbol{x}_i, y) &\triangleq \max_{\boldsymbol{\beta} \in \mathscr{B}(\boldsymbol{x}_i)} \boldsymbol{w}^\top \phi(\boldsymbol{x}_i, y, \boldsymbol{\beta}), \end{aligned} \tag{4.1}$$

where $s_{\boldsymbol{w}}\colon \mathscr{X} \times \mathscr{Y} \mapsto \mathbb{R}$ provides the score of a model $\boldsymbol{w} \in \mathbb{R}^b$, for a given sample and class, while $\mathscr{B}(\boldsymbol{x}_i)$ defines the feasible set for the latent variable $\boldsymbol{\beta}$, given the sample $\boldsymbol{x}_i$. Finally, $\phi\colon \mathscr{X} \times \mathscr{Y} \times \mathscr{B}(\boldsymbol{x}_i) \mapsto \mathbb{R}^b$ defines a feature mapping that depends on the candidate class y and the latent variable $\boldsymbol{\beta}$.

The main idea of latent SVM is to find, for each instance $\boldsymbol{x}_i$, a feature mapping ϕ maximizing the confidence of the model $\boldsymbol{w}$ on the sample. For example, for object detection the model $\boldsymbol{w}$ would likely include a set of components (specific sub-models) corresponding to different poses (frontal, lateral, etc.), with the scoring function (4.1) being responsible for selecting the best fitting one.

Let $\mathbf{1}(u)$ be the indicator function of the predicate u. The 0/1 classification error for a sample $(\boldsymbol{x}_i, y_i)$ can be written as $\mathbf{1}\left(\hat{y}_i(\boldsymbol{w}) \neq y_i\right)$, and a piece-wise linear upperbound can be formed by [Crammer and Singer, 2001]

$$\mathbf{1}\left(\hat{y}_i(\boldsymbol{w}) \neq y_i\right) \leq \max_{y \in \mathscr{Y}} \left[\mathbf{1}\left(y \neq y_i\right) + s_{\boldsymbol{w}}\left(\boldsymbol{x}_i, y\right)\right] - s_{\boldsymbol{w}}\left(\boldsymbol{x}_i, y_i\right) \tag{4.2a}$$

$$= \left| 1 + \max_{y \in \mathscr{Y} \setminus \{y_i\}} s_{\boldsymbol{w}}(\boldsymbol{x}_i, y) - s_{\boldsymbol{w}}(\boldsymbol{x}_i, y_i) \right|_+ . \tag{4.2b}$$

A multi class latent SVM objective function can then be obtained by averaging the upperbound on the 0/1 classification error for the n training samples, and combining it with a regularizer $h\colon \mathbb{R}^b \mapsto \mathbb{R}$, measuring the complexity of the model $\boldsymbol{w}$:

$$\min_{\boldsymbol{w}} \lambda\, h(\boldsymbol{w}) + \frac{1}{n} \sum_{i=1}^{n} \left| 1 + \max_{y \in \mathscr{Y} \setminus \{y_i\}} s_{\boldsymbol{w}}(\boldsymbol{x}_i, y) - s_{\boldsymbol{w}}(\boldsymbol{x}_i, y_i) \right|_+ , \tag{4.3}$$

where $\lambda \in \mathbb{R}^+$ sets the trade-off between regularization and training error minimization. This objective function can also be rewritten as a constrained optimization problem:

$$\min_{\boldsymbol{w}, \boldsymbol{\xi}} \lambda\, h(\boldsymbol{w}) + \frac{1}{n} \sum_{i=1}^{n} \xi_i \tag{4.4a}$$

s.t.

$$1 + \max_{y \in \mathscr{Y} \setminus \{y_i\}} s_{\boldsymbol{w}}(\boldsymbol{x}_i, y) - s_{\boldsymbol{w}}(\boldsymbol{x}_i, y_i) \leq \xi_i, \qquad i = 1, \ldots, n \tag{4.4b}$$

$$\xi_i \geq 0, \qquad i = 1, \ldots, n, \tag{4.4c}$$

where the equivalence with (4.3) can be quickly verified by a case analysis on the ξ_i [1].

Due to the maximization, $s_{\boldsymbol{w}}$ is strictly convex w.r.t. $\boldsymbol{w}$ so that the constraints in (4.4b) have the form of a difference of convex functions and are thus non-convex. A general way to optimize such problems is to use the Constrained Concave Convex Procedure (CCCP) [Yuille and Rangarajan, 2003; Smola et al., 2005; Sriperumbudur and Lanckriet, 2012].

1. If for a given sample $\boldsymbol{x}_i$, the LHS of (4.4b) is less then zero, then the minimal ξ_i is equal to 0, due to the constraints in (4.4c). If on the contrary for a given sample $\boldsymbol{x}_i$, the LHS of (4.4b) is grater then or equal to zero, the minimal ξ_i is simply $\xi_i = 1 + \max_{y \in \mathscr{Y} \setminus \{y_i\}} s_{\boldsymbol{w}}(\boldsymbol{x}_i, y) - s_{\boldsymbol{w}}(\boldsymbol{x}_i, y_i)$.

Algorithm 1 Constrained Concave Convex Procedure

1: Initialize $t = 0$ and $\boldsymbol{w}^t$ with a random value
2: **repeat**
3: $\quad \boldsymbol{w}^{t+1} = \arg\min_{\boldsymbol{w}} f_0(\boldsymbol{w}) - t_{\{g_0, \boldsymbol{w}^t\}}(\boldsymbol{w})$
4: $\quad\quad$ s.t. $f_i(\boldsymbol{w}) - t_{\{g_i, \boldsymbol{w}^t\}}(\boldsymbol{w}) \leq c_i \quad \forall i = 1, \ldots, n$
5: **until** a stopping criterion is satisfied

4.3.2 The Constrained Concave Convex (CCCP) procedure.

Suppose we need to optimize a problem of the form

$$\min_{\boldsymbol{w}} f_0(\boldsymbol{w}) - g_0(\boldsymbol{w}), \tag{4.5a}$$

$$\text{s.t. } f_i(\boldsymbol{w}) - g_i(\boldsymbol{w}) \leq c_i \quad \forall i = 1, \ldots, n, \tag{4.5b}$$

where f_i and g_i are real valued convex functions on a vector space $\mathcal{X}$. A possible minimization approach is to apply the CCCP optimization procedure [Smola et al., 2005] reported in Algorithm 1, where $t_{\{g_i, \boldsymbol{w}^t\}}$ is the first-order Taylor approximation of g_i around $\boldsymbol{w}^t$. The main idea of the CCCP algorithm is to iteratively minimize a convex majorization of (4.5). This is obtained by replacing each g_i with a linearization, so that $f_i - t_{\{g_i, \boldsymbol{w}^t\}}$ becomes convex and, since also each g_i is convex, $f_i(\boldsymbol{w}) - t_{\{g_i, \boldsymbol{w}^t\}}(\boldsymbol{w}) \geq f_i(\boldsymbol{w}) - g_i(\boldsymbol{w})$. Any feasible point in Algorithm 1 is thus also a feasible point of problem (4.5). If f_0 is continuous and strictly convex, $\{f_i\}_{i=1}^n$ are continuous, while $\{g_i\}_{i=0}^n$ are continuously differentiable, the algorithm can be shown to converge to a stationary point of (4.5) (Theorem 5 of Sriperumbudur and Lanckriet [2012]) [2].

4.3.3 Locally Linear SVMs

Locally Linear Support Vector Machines were introduced by Ladicky and Torr [2011] as a method to learn smooth non-linear classifiers, without using kernels. The algorithm is based on the idea that, for any smooth classifier "*in a sufficiently small region the decision boundary is approximately linear and the data is locally linearly separable*" [Ladicky and Torr, 2011]. Following this intuition, the authors propose the idea of locally linear classification functions. Informally, this can be described as a set of hypotheses that locally behave as linear operators, computing the inner product $\boldsymbol{w}^\top \boldsymbol{x}$ between a vector $\boldsymbol{w} = \omega(\boldsymbol{x})$ and a sample $\boldsymbol{x}$, where the dependency of the hyper-plane $\boldsymbol{w}$ on the sample $\boldsymbol{x}$ is described by a set of real valued functions $\{\omega_1(\cdot), \omega_2(\cdot), \ldots\}$ (one for each dimension of $\boldsymbol{x}$). A formal definition of a locally linear function is provided in Section 4.4.1.

The goal of LLSVMs is to learn locally linear binary classifiers using decision functions of the form $\text{sign}\left(\omega(\boldsymbol{x})^\top \boldsymbol{x}\right)$. Following Yu et al. [2009], the authors assume $\omega(\boldsymbol{x})$ to be a Lipschitz

2. As noted in Sriperumbudur and Lanckriet [2012], previous proofs of convergence of CCCP to local minima in Yuille and Rangarajan [2003]; Smola et al. [2005] are incomplete. Hence, the best currently known result for CCCP is just the convergence to a stationary point. As stated in Section 5 in Sriperumbudur and Lanckriet [2012], finding conditions for the local convergence of CCCP remains an open problem, beyond the scope of this work.

smooth function and the locally linear function $\omega(\boldsymbol{x})^\top \boldsymbol{x}$ is therefore approximated with a local linear combination of m linear models: $\omega(\boldsymbol{x})^\top \boldsymbol{x} \approx \beta(\boldsymbol{x})^\top \boldsymbol{W}\boldsymbol{x}$. The mixture coefficients $\beta(\boldsymbol{x})$ are obtained in advance using an unsupervised manifold learning procedure supposed to provide localized mixing coefficients $\beta(\boldsymbol{x})$. Ladicky and Torr [2011] implement this idea by using k-means clustering and fixing $\beta(\boldsymbol{x})$ using inverse Euclidean distances, solved only for the closest 8 cluster centers, an idea similar to [Wang et al., 2010a]. The authors state that this simple approach results to be competitive with more principled encoding techniques [Yu et al., 2009] based on the minimization of reconstruction and localization errors.

More formally, their formulation reads as follows. Given a set of training examples $(\boldsymbol{x}_i, y_i) \in \mathscr{X} \times \mathscr{Y}$ and their corresponding coefficient vectors $\beta(\boldsymbol{x}_i)$, where $i = \{1, \ldots, n\}$, $\mathscr{X} \subseteq \mathbb{R}^d$ is the input space and $\mathscr{Y} = \{-1, 1\}$ is a binary output space, the objective function of a locally linear SVM is defined as

$$\min_{\boldsymbol{W}} \frac{\lambda}{2} \|\boldsymbol{W}\|_F^2 + \frac{1}{n} \sum_{i=1}^{n} \left|1 - y_i \beta(\boldsymbol{x}_i)^\top \boldsymbol{W}\boldsymbol{x}_i\right|_+ .$$

This objective function strictly resembles the objective function of a Support Vector Machine, it is convex and it can be efficiently optimized using stochastic gradient descent algorithms [Shalev-Shwartz et al., 2011].

4.4 The proposed approach

This Section provides a detailed description of the approach proposed in this Chapter. We first introduce a formal definition of locally linear functions and propose a simple and efficient *Locally Linear Coding* (*L2C*) approach, designed to approximate this class of functions. Supported by this initial analysis, we then introduce the main contribution of this Chapter: the *Multiclass Latent Locally Linear SVM* (*ML3*) classification algorithm. This algorithm targets the same class of functions considered by Locally Linear SVMs, while making use of the Latent SVM classification framework introduced above (see in Section 4.2). As for Latent SVM, our model is trained using the CCCP procedure discussed in Section 4.2, while each of the CCCP subproblems is solved using stochastic gradient descent (SGD), with an adapted version of the algorithm of Shalev-Shwartz et al. [2011].

4.4.1 Locally Linear Coding (L2C)

In this Section we describe a simple and efficient coding approach, specifically designed for Locally Linear Support Vector Machines. The theory presented in this Section follows the idea of Yu et al. [2009], adapting it to the class of functions used by Locally Linear SVMs. The main idea of Yu et al. [2009] is that non-linear functions with certain smoothness properties can be well approximated by projecting the samples into a high-dimensional space, using an encoding scheme ensuring some degree of locality of the generated codes. A simple linear learning in this projected space is then sufficient to accurately approximate the non-linear

function in the input space. We develop this idea for the class of locally linear functions.

Please note that the aim of this Section is not to design a complex and competitive encoding technique. On the contrary, the main purpose of this work is to analyze the properties of a simple and very efficient encoding approach that is directly related to the ML3 algorithm presented in this Chapter.

We start our discussion with the definition of *locally linear functions*, as proposed by Ladicky and Torr [2011] [3]. We chose to analyze this class of functions since it gives rise to approximations of the form $\beta(\boldsymbol{x})^\top \boldsymbol{W}\boldsymbol{x}$ used by locally linear SVMs, such as the ML3 algorithm proposed in this Chapter.

Def (Locally linear function) 1. A function $f : \mathscr{X} \subseteq \mathbb{R}^d \rightarrow \mathbb{R}$ is said to be *locally linear* if for every $\boldsymbol{x} \in \mathscr{X}$ it can be written as [Ladicky and Torr, 2011]:

$$f(\boldsymbol{x}) = \omega(\boldsymbol{x})^\top \boldsymbol{x} = \sum_{i=1}^{d} \omega(\boldsymbol{x})_i x_i \tag{4.6}$$

where $w : \mathscr{X} \rightarrow \mathscr{X}$ is a vector-valued function that can be broken down into d separate parts $\{\omega_i : \mathscr{X} \rightarrow \mathbb{R}\}_{i=1}^{d}$, each one mapping the input $\boldsymbol{x}$ to a point in the i-th dimension of $\mathscr{X}$.

Simple examples of functions naturally belonging to this category are linear functions $f(\boldsymbol{x}) = \boldsymbol{a}^\top \boldsymbol{x}$ (obtained by fixing $\omega(\boldsymbol{x}) \triangleq \boldsymbol{a}$) and quadratic functions $f(\boldsymbol{x}) = \boldsymbol{x}^\top \boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}^\top \boldsymbol{x}$, obtained by using ω of the form $\omega(\boldsymbol{x}) \triangleq \boldsymbol{A}^\top \boldsymbol{x} + \boldsymbol{b}$.

As shown by Yu et al. [2009], an important concept to obtain well-behaving encoding schemes is the notion of Lipschitz smoothness. We make use of the generalization of this concept to vector valued functions.

Def (Lipschitz smoothness) 2. A vector-valued function $f : \mathscr{X} \subseteq \mathbb{R}^d \rightarrowtail \mathbb{R}^k$ is *α-Lipschitz smooth* on $\mathscr{X}$, with respect to a given norm $\|\cdot\|$, if for every $\boldsymbol{x}, \boldsymbol{x}' \in \mathscr{X}$, there exists an $\alpha \geq 0$ s.t. $\left\| f(\boldsymbol{x}) - f(\boldsymbol{x}') \right\| \leq \alpha \left\| \boldsymbol{x} - \boldsymbol{x}' \right\|$.

Intuitively, Lipschitz smooth vector-valued functions are functions that are limited in how fast they can change. Indeed, there exists a non-negative real number α such that, for every pair of points $\boldsymbol{x}$ and $\boldsymbol{x}'$, the value of the slope $\frac{\|f(\boldsymbol{x}) - f(\boldsymbol{x}')\|}{\|\boldsymbol{x} - \boldsymbol{x}'\|}$ is not greater than α. The Lipschitz smoothness of locally linear functions is characterized by Proposition 2 in Appendix A.

We can now state the following Lemma, motivating a simple technique to approximate any locally linear function with a α-Lipschitz smooth $\omega(\cdot)$.

Lemma (Linearization) 3. *Let $\mathscr{X}_\rho \equiv \{\boldsymbol{x} \in \mathbb{R}^d : \|\boldsymbol{x}\|_2 \leq \rho\}$ and $\omega : \mathscr{X}_\rho \rightarrow \mathbb{R}^d$ be a vector-valued function with d dimensions $\{\omega(\cdot)_i\}_{i=1}^{d}$. Let $f : \mathscr{X}_\rho \rightarrowtail \mathbb{R}$ be a locally linear function of the form*

3. For simplicity in this work we skip the bias term $b(\boldsymbol{x})$, used in Ladicky and Torr [2011]. This term can still be recovered by increasing the dimensionality of $\mathscr{X}$, concatenating a 1 to each instance.

$f(\boldsymbol{x}) = \omega(\boldsymbol{x})^\top \boldsymbol{x}$, and let $\boldsymbol{V} = \begin{bmatrix} \boldsymbol{v}_1 & \boldsymbol{v}_2 & \cdots & \boldsymbol{v}_m \end{bmatrix}^\top \in \mathbb{R}^{m \times d}$ be an arbitrary $m \times d$ matrix. If $\omega(\cdot)$ is α-Lipschitz smooth on $\mathcal{X}_\rho$ w.r.t. the ℓ_2-norm, then for all $\boldsymbol{x} \in \mathcal{X}_\rho$ and $\boldsymbol{\beta} \in \mathbb{R}^m$ s.t. $\boldsymbol{\beta} \geq 0$ and $\|\boldsymbol{\beta}\|_1 = 1$, the following inequality holds, for some $\gamma \geq 0$ that depends on the specific form of $\omega(\cdot)$:

$$\left| f(\boldsymbol{x}) - \sum_{i=1}^{m} \beta_i \omega(\boldsymbol{v}_i)^\top \boldsymbol{x} \right| \leq \rho\gamma \left\| \boldsymbol{x} - \boldsymbol{V}^\top \boldsymbol{\beta} \right\|_2 + \rho\left(\gamma + \alpha\right) \sum_{i=1}^{m} \beta_i \left\| \boldsymbol{x} - \boldsymbol{v}_i \right\|_2 . \tag{4.7}$$

Proof. A proof is provided by Lemma 3 in Appendix A. □

The vectors $\boldsymbol{v}_i$ are referred to as *anchor points* [Yu et al., 2009]. The LHS term of equation (4.7) can be understood as the error obtained when using $\sum_{i=1}^{m} \beta_i \omega(\boldsymbol{v}_i)^\top \boldsymbol{x}$ to approximate $f(\boldsymbol{x})$. The first term on the RHS of equation (4.7) is the *reconstruction error* obtained by using $\boldsymbol{V}^\top \boldsymbol{\beta}$ to approximate $\boldsymbol{x}$. The second term is instead called the *localization error* [Yu et al., 2009], as it promotes selecting vectors $\boldsymbol{v}_i$ that are close to $\boldsymbol{x}$.

This bound motivates a simple way to approximate locally linear functions, using a linear combination of linear ones. Indeed, suppose that we want to learn an approximation of an unknown locally linear function $f : \mathcal{X}_\rho \to \mathbb{R}$, satisfying the hypotheses of Lemma 3. Given a set of anchor points $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_m$, f can be approximated by computing the inner product of the sample $\boldsymbol{x}$ and a linear combination of a fixed set of vectors $\{\omega(\boldsymbol{v}_1), \omega(\boldsymbol{v}_2), \ldots, \omega(\boldsymbol{v}_m)\}$. We can thus create one parameter $\boldsymbol{w}_i = \omega(\boldsymbol{v}_i)$ for each unknown but fixed vector $\omega(\boldsymbol{v}_i)$, and use the following parametric approximation of the function f:

$$f(\boldsymbol{x}) \approx \sum_{i=1}^{m} \beta_i \boldsymbol{w}_i^\top \boldsymbol{x} = \boldsymbol{\beta}^\top \boldsymbol{W} \boldsymbol{x} = \mathrm{Tr}\left(\boldsymbol{W}^\top \left(\boldsymbol{\beta} \boldsymbol{x}^\top\right)\right) = \boldsymbol{W} \cdot \psi(\boldsymbol{x}, \boldsymbol{\beta}), \tag{4.8a}$$

$$\psi(\boldsymbol{x}, \boldsymbol{\beta}) \triangleq \boldsymbol{\beta} \boldsymbol{x}^\top, \tag{4.8b}$$

where $\boldsymbol{W} = \begin{bmatrix} \boldsymbol{w}_1 & \boldsymbol{w}_2 & \ldots & \boldsymbol{w}_m \end{bmatrix}^\top$ is the matrix obtained by stacking all the m $\boldsymbol{w}_i$ together, $\psi(\boldsymbol{x}, \boldsymbol{\beta})$ is the *L2C code* of the sample $\boldsymbol{x}$ using $\boldsymbol{\beta}$, while $\boldsymbol{W} \cdot \psi(\boldsymbol{x}, \boldsymbol{\beta})$ indicates the Frobenius inner product between the matrices $\boldsymbol{W}$ and $\psi(\boldsymbol{x}, \boldsymbol{\beta})$.

Note that, although $f(\boldsymbol{x})$ is possibly a non-linear function of $\boldsymbol{x}$, its approximation using Lemma 3 is a linear function of the parameter $\boldsymbol{W}$. For this reason, Lemma 3 is called *Linearization Lemma*.

Practical minimization of the bound

The quality of the approximation of f depends on how tight is the upper-bound on the approximation error. A correct approach would be to directly minimize the RHS of eq (4.7) w.r.t the matrix $\boldsymbol{V}$ and the vector $\boldsymbol{\beta}$. This is the approach used by Yu et al. [2009]. While this is optimal from the point of view of the minimization of the bound, it would also lead to a

complex encoding procedure requiring to solve a LASSO problem [Yu et al., 2009]. In this Section we discuss a very simple technique to upper-bound both the reconstruction and the localization error, making it possible to efficiently obtain the solution for $\boldsymbol{\beta}$ in a closed form.

As shown by Proposition 4 in Appendix A, under the constraints $\boldsymbol{\beta} \geq 0$ and $\left\|\boldsymbol{\beta}\right\|_1 = 1$, required by Lemma 3, it is possible to upper-bound the localization term on the RHS of equation (4.7) with

$$\sum_{i=1}^{m} \beta_i \left\|\boldsymbol{x} - \boldsymbol{v}_i\right\|_2 \leq \left(\left\|\boldsymbol{x}\right\|_2^2 + \left\|\boldsymbol{V}\right\|_F^2 - 2\boldsymbol{\beta}^\top \boldsymbol{V}\boldsymbol{x}\right)^{\frac{1}{2}}, \tag{4.9}$$

where $\boldsymbol{V} = \begin{bmatrix} \boldsymbol{v}_1 & \boldsymbol{v}_2 & \dots & \boldsymbol{v}_m \end{bmatrix}^\top$. Moreover, whenever $\left\|\boldsymbol{\beta}\right\|_p \leq 1$, with $p \leq 2$ (which includes the case $\left\|\boldsymbol{\beta}\right\|_1 = 1$), also the reconstruction error on the RHS of equation (4.7) can be upper-bounded by

$$\left\|\boldsymbol{x} - \boldsymbol{V}^\top \boldsymbol{\beta}\right\|_2 \leq \left(\left\|\boldsymbol{x}\right\|_2^2 + \left\|\boldsymbol{V}\right\|_F^2 - 2\boldsymbol{\beta}^\top \boldsymbol{V}\boldsymbol{x}\right)^{\frac{1}{2}}, \tag{4.10}$$

as shown by Proposition 4 in Appendix A.

Consequently, the approximation error in equation (4.7), can be further upper-bounded by:

$$\left| f(\boldsymbol{x}) - \sum_{i=1}^{m} \beta_i \omega(\boldsymbol{v}_i)^\top \boldsymbol{x} \right| \leq \rho\left(2\gamma + \alpha\right)\left(\left\|\boldsymbol{x}\right\|_2^2 + \left\|\boldsymbol{V}\right\|_F^2 - 2\boldsymbol{\beta}^\top \boldsymbol{V}\boldsymbol{x}\right)^{\frac{1}{2}}. \tag{4.11}$$

It is important to note that, while the constraint $\left\|\boldsymbol{\beta}\right\|_1 = 1$ is required for the RHS of (4.9) to be a valid upper-bound of the localization error, the relaxed constraint $\left\|\boldsymbol{\beta}\right\|_p \leq 1$ with $p \leq 2$ is sufficient for the same quantity to be a valid upper-bound on the reconstruction error (4.10). Indeed, as it will be shown in the following, using the constraint $\left\|\boldsymbol{\beta}\right\|_p \leq 1$ with $p > 1$ may lead to a decrease (or, in extreme cases, to a non-increase) of the reconstruction error. Following this idea and ignoring the constants, the objective function for the minimization of the approximation error in (4.11) can be defined as

$$L(\boldsymbol{\beta}, \boldsymbol{V}, \boldsymbol{x}) = \frac{1}{2}\left\|\boldsymbol{V}\right\|_F^2 - \boldsymbol{\beta}^\top \boldsymbol{V}\boldsymbol{x},$$

where, for what we have discussed before, the minimization w.r.t. $\boldsymbol{\beta}$ is performed in the (relaxed) convex set

$$\Omega_p \triangleq \{\boldsymbol{\beta} \in \mathbb{R}^m : \boldsymbol{\beta} \geq 0, \left\|\boldsymbol{\beta}\right\|_p \leq 1\}, \tag{4.12}$$

with $1 \leq p \leq 2$.

Minimization w.r.t. the coefficients β

In order to minimize $L(\boldsymbol{\beta}, \boldsymbol{V}, \boldsymbol{x})$ w.r.t. $\boldsymbol{\beta}$, for any given $\boldsymbol{V}$ we need to compute

$$\beta_{\boldsymbol{V}}(\boldsymbol{x}) \triangleq \underset{\boldsymbol{\beta}\in\Omega_p}{\arg\min} \frac{1}{2}\|\boldsymbol{V}\|_F^2 - \boldsymbol{\beta}^\top \boldsymbol{V}\boldsymbol{x} = \underset{\boldsymbol{\beta}\in\Omega_p}{\arg\max}\, \boldsymbol{\beta}^\top \boldsymbol{V}\boldsymbol{x}, \tag{4.13}$$

where $p \in [1,2]$. The optimization problem in equation (4.13) resembles a linear program, with p-norm ball constraints and its solution can be computed in closed form. When $p > 1$, a solution is provided by the following Lemma.

Lemma (Solution for $\beta_{\boldsymbol{V}}(\boldsymbol{x})$ and $1 < p < \infty$) 4. *Let $\boldsymbol{V} = \begin{bmatrix} \boldsymbol{v}_1 & \boldsymbol{v}_2 & \cdots & \boldsymbol{v}_m \end{bmatrix}^\top \in \mathbb{R}^{m\times d}$ and $\beta_{\boldsymbol{V}}(\boldsymbol{x})$ be defined as in equation (4.13). If $1 < p < \infty$, for every $\boldsymbol{x} \in \mathbb{R}^d$ s.t. $\left\|(\boldsymbol{V}\boldsymbol{x})^+\right\|_q > 0$, with $q = p/(p-1)$, the j-th element of the optimal vector $\beta_{\boldsymbol{V}}(\boldsymbol{x})$ is given by*

$$\beta_{\boldsymbol{V}}(\boldsymbol{x})_j = \left(\frac{\left|\boldsymbol{v}_j^\top \boldsymbol{x}\right|_+}{\left\|(\boldsymbol{V}\boldsymbol{x})^+\right\|_q} \right)^{q-1}. \tag{4.14}$$

Furthermore, $\beta_{\boldsymbol{V}}(\boldsymbol{x})^\top \boldsymbol{V}\boldsymbol{x} = \left\|(\boldsymbol{V}\boldsymbol{x})^+\right\|_q$.

Proof. A proof is provided in Lemma 1 in Appendix A. □

For the points $\boldsymbol{x} \in \mathbb{R}^d$ s.t. $\left\|(\boldsymbol{V}\boldsymbol{x})^+\right\|_q = 0$ (i.e. points on which $\boldsymbol{v}_i^\top \boldsymbol{x} \le 0,\ \forall i \in \{1,\dots,m\}$) a trivial solution maximizing the objective function in equation (4.13) is $\beta_{\boldsymbol{V}}(\boldsymbol{x}) = \boldsymbol{0}$. Using this solution, we have once again $\beta_{\boldsymbol{V}}(\boldsymbol{x})^\top \boldsymbol{V}\boldsymbol{x} = \left\|(\boldsymbol{V}\boldsymbol{x})^+\right\|_q = 0$.

This closed form solution is valid for any $p \in (1,\infty)$. As it is possible to see, varying p in $(1,\infty)$ allows to move from the case where only the anchor point $\boldsymbol{v}_j$ with the highest (positive) projection on $\boldsymbol{x}$ is assigned weight 1 (when $p \to 1$, $q \to \infty$), to the case where all the positively projecting anchor points are assigned the same weight (when $p \to \infty$, $q \to 1$).

We now provide an exact solution also for the case $p = 1$. As said before, we do not consider the case $p < 1$, as the ℓ_p-norm becomes non-convex.

Solution for $p = 1$. For every $\boldsymbol{\beta} \in \Omega_1$ it is possible to write $\boldsymbol{\beta}^\top \boldsymbol{V}\boldsymbol{x} = \sum_j \beta_j \boldsymbol{v}_j^\top \boldsymbol{x} \le \max_k \boldsymbol{v}_k^\top \boldsymbol{x} \sum_j \beta_j \le \max_k \boldsymbol{v}_k^\top \boldsymbol{x}$. Consequently, if $\left|\max_k \boldsymbol{v}_k^\top \boldsymbol{x}\right|_+ \neq 0$ an optimal solution is simply given by

$$\beta_{\boldsymbol{V}}(\boldsymbol{x}) = \begin{bmatrix} 0 & \cdots & 0 & 1 & 0 & \cdots & 0 \end{bmatrix}^\top, \tag{4.15}$$

where the only 1 is in the position $\arg\max_k \boldsymbol{v}_k^\top \boldsymbol{x}$. Whenever $\arg\max_k \boldsymbol{v}_k^\top \boldsymbol{x}$ is not unique (i.e. the maximum is achieved by more than one element of the vector), any convex combination of those elements would provide the same maximal value of the objective function. In this case,

amongst the possible solutions we again chose one with the form of (4.15). Finally, as before if $\left|\max_k \boldsymbol{w}_k^\top \boldsymbol{x}\right|_+ = 0$ (i.e. $\boldsymbol{w}_i^\top \boldsymbol{x} \leq 0$, $\forall i \in \{1,\ldots,m\}$) an optimal solution is given by $\beta_{\boldsymbol{V}}(\boldsymbol{x}) = \boldsymbol{0}$. Using this solution for $p = 1$, the value of the objective function at the optimum can once again be written as $\beta_{\boldsymbol{V}}(\boldsymbol{x})^\top \boldsymbol{V}\boldsymbol{x} = \left\|(\boldsymbol{V}\boldsymbol{x})^+\right\|_q = 0$, with $q = \infty$.

Using the closed form solution for $\beta_{\boldsymbol{V}}(\boldsymbol{x})$ we obtain a more compact objective function, depending only on $\boldsymbol{V}$

$$L(\boldsymbol{V},\boldsymbol{x}) = \frac{1}{2}\|\boldsymbol{V}\|_F^2 - \left\|(\boldsymbol{V}\boldsymbol{x})^+\right\|_q. \tag{4.16}$$

By the equivalence of the norms, if $p_2 \geq p_1$ (e.g. $p_1 = 1$ and $p_2 = 1.5$), implying $q_2 \leq q_1$, we have $\left\|(\boldsymbol{V}\boldsymbol{x})^+\right\|_{q_2} \geq \left\|(\boldsymbol{V}\boldsymbol{x})^+\right\|_{q_1}$. As anticipated above, this means that for any fixed $\boldsymbol{V}$, the (upper-bound of the) reconstruction error in (4.10) can be further minimized by setting $p > 1$. Unfortunately, as noted above, with this setting the RHS of (4.11) is not a valid upper-bound of the localization error anymore. This means that the localization error might increase. Consequently, by tuning $1 \leq p \leq 2$, one may achieve a trade-off between the minimization of the localization term and the reconstruction term in equation (4.7).

Minimization w.r.t. V

Suppose we are given a set of training examples $\{\boldsymbol{x}_i\}_{i=1}^n$, with $\boldsymbol{x}_i \in \mathscr{X} \subseteq \mathbb{R}^d$. By averaging $L(\boldsymbol{V},\boldsymbol{x}_i)$ over the n training samples and minimizing w.r.t. $\boldsymbol{V}$, we obtain an objective function of the form: $\min_{\boldsymbol{V}} \frac{1}{2}\|\boldsymbol{V}\|_F^2 - \frac{1}{n}\sum_{i=1}^n \left\|(\boldsymbol{V}\boldsymbol{x}_i)^+\right\|_q$. Alternatively, taking inspiration from the dictionary learning literature [Mairal et al., 2009], we can also define the optimization problem:

$$\begin{aligned}&\min_{\boldsymbol{V}} -\frac{1}{n}\sum_{i=1}^n \left\|(\boldsymbol{V}\boldsymbol{x}_i)^+\right\|_q \\ &\text{s.t.}\, \boldsymbol{v}_j^\top \boldsymbol{v}_j \leq 1,\end{aligned}$$

where the bound on the squared norm of the columns of $\boldsymbol{V}$ guarantees that $\|\boldsymbol{V}\|_F^2 \leq m$, so that $L(\boldsymbol{V},\boldsymbol{x})$ can be upper-bounded by $m - \left\|(\boldsymbol{V}\boldsymbol{x})^+\right\|_q$. This optimization problem has the form of a difference of convex functions (where the convex part in the objective and the concave part in the constraints is the function 0) and can be solved using the CCCP procedure discussed in Section 4.3.2.

4.4.2 Multiclass Latent Locally Linear SVM (ML3)

Motivated by the L2C approach discussed before, this Section introduces our multi-component classification algorithm, which constitutes the core contribution of this Chapter.

Suppose we are given a set of training examples $(\boldsymbol{x}_i, y_i) \in \mathscr{X} \times \mathscr{Y}$, where $i = \{1,\ldots,n\}$, y_i is a class label associated to $\boldsymbol{x}_i$, $\mathscr{X} \subseteq \mathbb{R}^d$ is the input space and $\mathscr{Y} = \{1,\ldots,c\}$ is the output (and the

decision) space. Suppose also that, for each class y, there exists an unknown locally linear scoring function $s_y : \mathscr{X} \mapsto \mathbb{R}$, satisfying the hypotheses of Lemma 3 and providing a measure of the confidence that the sample $\boldsymbol{x}_i$ belongs to class y. As discussed above, $s_y(\boldsymbol{x}_i)$ can be learned using a parametric approximation of the form:

$$s_y(\boldsymbol{x}_i) \approx \beta_{\boldsymbol{V}}(\boldsymbol{x}_i)^\top \boldsymbol{W}_y \boldsymbol{x}_i = \boldsymbol{W}_y \cdot \psi\left(\boldsymbol{x}_i, \beta_{\boldsymbol{V}}(\boldsymbol{x}_i)\right),$$

where $\boldsymbol{W}_y$ is a matrix of parameters that can be learned in a discriminative fashion (e.g. as in LLSVMs), while the matrix of anchor points $\boldsymbol{V}$ and the L2C codes $\psi\left(\boldsymbol{x}_i, \beta_{\boldsymbol{V}}(\boldsymbol{x}_i)\right)$ can be obtained by minimizing $L\left(\boldsymbol{\beta}, \boldsymbol{V}, \boldsymbol{x}_i\right)$. The main limitation of this approach (and of the LLSVM and LCC approaches adopted by Ladicky and Torr [2011] and Yu et al. [2009]) is that the matrix $\boldsymbol{V}$ (and thus the codes $\psi\left(\boldsymbol{x}_i, \beta_{\boldsymbol{V}}(\boldsymbol{x}_i)\right)$) has to be learned in advance, using an unsupervised learning procedure. If we consider $\boldsymbol{W}_y$ as a multi-component model with m components $\{\boldsymbol{w}_{1y}, \ldots, \boldsymbol{w}_{my}\}$, we can see that with this choice the components of $\boldsymbol{W}_y$ are allocated to a given sample $\boldsymbol{x}_i$ only taking into account the distribution of the instances, without considering the labels. This is sub-optimal from a classification point of view. Similarly to Zhang et al. [2011], instead of using only one single dictionary for all the samples, we may thus opt to construct a set of class-specific dictionaries $\{\boldsymbol{V}_1, \ldots, \boldsymbol{V}_c\}$, each one specialized to reduce the approximation error for the samples in its class. Moreover, instead of having a separate procedure to learn the matrices $\boldsymbol{W}_y$ and the dictionaries $\boldsymbol{V}_y$, we may want to train the matrices $\boldsymbol{W}_y$ and $\boldsymbol{V}_y$ jointly. This can be achieved by simply enforcing $\boldsymbol{V}_y = \boldsymbol{W}_y$, resulting in an approach similar to the Discriminatively Learned Dictionaries of Mairal et al. [2008]. Although the components $\boldsymbol{w}_{iy}$ of $\boldsymbol{W}_y$ are sub-optimal from a function approximation point of view, they can be directly trained to minimize the classification error. Moreover, for each sample $\boldsymbol{x}_i$, the function approximation error can still be minimized by computing the optimal $\beta_{\boldsymbol{W}_y}(\boldsymbol{x}_i)$ for each sample. With this choice, the parametric approximation of the scoring function $s_y(\boldsymbol{x}_i)$ can thus be written as

$$s_y(\boldsymbol{x}_i) \approx \beta_{\boldsymbol{W}_y}(\boldsymbol{x}_i)^\top \boldsymbol{W}_y \boldsymbol{x}_i = \max_{\boldsymbol{\beta} \in \Omega_p} \boldsymbol{\beta}^\top \boldsymbol{W}_y \boldsymbol{x}_i$$

In the following we show how this function can be cast in the Latent SVM framework introduced in Section 4.3.1, resulting in the ML3 learning algorithm.

The ML3 model

We now introduce the ML3 algorithm, casting the learning problem within the Latent SVM framework discussed in Section 4.3.1. We use m-dimensional latent variables $\boldsymbol{\beta}$, with a feasible set Ω_p defined (for all the samples $\boldsymbol{x}_i$) as in equation (4.12). Using this feasible set, we define the feature mapping $\phi : \mathscr{X} \times \mathscr{Y} \times \Omega_p \mapsto \mathbb{R}^{mc \times d}$ as

$$\phi\left(\boldsymbol{x}_i, y, \boldsymbol{\beta}\right) \triangleq \left[\underbrace{\boldsymbol{0}}_{1} \quad \ldots \quad \boldsymbol{0} \quad \underbrace{\psi\left(\boldsymbol{x}_i, \boldsymbol{\beta}\right)^\top}_{y} \quad \boldsymbol{0} \quad \ldots \quad \underbrace{\boldsymbol{0}}_{c}\right]^\top \in \mathbb{R}^{mc \times d},$$

where in this case $\mathbf{0}$ indicates a $d \times m$ zero-valued matrix and $\psi\left(\boldsymbol{x}_i, \boldsymbol{\beta}\right)$ is the L2C code of $\boldsymbol{x}_i$ using $\boldsymbol{\beta}$, defined in (4.8b).

Our multi class model $\boldsymbol{W}$ is then defined as the block matrix

$$\boldsymbol{W} \triangleq \begin{bmatrix} \boldsymbol{W}_1^\top & \boldsymbol{W}_2^\top & \dots & \boldsymbol{W}_c^\top \end{bmatrix}^\top \in \mathbb{R}^{m\,c \times d}, \tag{4.17a}$$

$$\boldsymbol{W}_y \triangleq \begin{bmatrix} \boldsymbol{w}_{1y} & \boldsymbol{w}_{2y} & \cdots & \boldsymbol{w}_{my} \end{bmatrix}^\top \in \mathbb{R}^{m \times d} \tag{4.17b}$$

where each block $\boldsymbol{W}_y$ is a class-specific model (for class y) and, according to the terminology used in this thesis, we call the m rows of $\boldsymbol{W}_y$ the components of the model $\boldsymbol{W}_y$.
Using this notation, we define the prediction of the ML3 algorithm as

$$\hat{y}_i\left(\boldsymbol{W}\right) \triangleq \underset{y \in \mathscr{Y}}{\arg\max}\, s_{\boldsymbol{W}}\left(\boldsymbol{x}_i, y\right), \tag{4.18a}$$

$$s_{\boldsymbol{W}}\left(\boldsymbol{x}_i, y\right) \triangleq \max_{\boldsymbol{\beta} \in \Omega_p} \boldsymbol{W} \cdot \phi\left(\boldsymbol{x}_i, y, \boldsymbol{\beta}\right) \tag{4.18b}$$

$$= \max_{\boldsymbol{\beta} \in \Omega_p} \mathrm{Tr}\left(\boldsymbol{W}_y^\top \boldsymbol{\beta} \boldsymbol{x}_i^\top\right) \tag{4.18c}$$

$$= \max_{\boldsymbol{\beta} \in \Omega_p} \boldsymbol{\beta}^\top \boldsymbol{W}_y \boldsymbol{x}_i \tag{4.18d}$$

where $s_{\boldsymbol{W}} : \mathscr{X} \times \mathscr{Y} \mapsto \mathbb{R}$ is the multi class scoring function of ML3. To keep the notation light, we may simply use $\hat{y}_i$ as a shortcut of $\hat{y}_i\left(\boldsymbol{W}\right)$, whenever $\boldsymbol{W}$ is clear from the context.

Note that the non-negativity constraints in Ω_p prevent the coefficients $\boldsymbol{\beta}$ from inverting the sign of the predictions of the components of $\boldsymbol{W}_y$. Indeed, as showed by Lemma 4 in Section 4.4.1, the negatively scoring components of $\boldsymbol{W}_y$ are always assigned a coefficient 0. This means that for any class y and any value of p, the scoring function of ML3 performs a form of component selection, suppressing the outputs of the components that provide a negative prediction on $\boldsymbol{x}_i$. As it will be discussed in the following, this also results in the fact that only the components that provide a positive prediction are updated using $\boldsymbol{x}_i$ (if necessary). For the ML3 algorithm we thus let p to vary in the full range $p \in [1, \infty)$. As shown by Lemma 4 (and as it will be empirically shown later) this allows to move from the case where only the most positively scoring component is contributing to the prediction of each sample, to the case where all the positively scoring components tend to contribute in the same way. This extends the latent SVM implementations in Wang and Mori [2009]; Felzenszwalb et al. [2010], that were limited to use only the single most confident component for a given sample.

Please note that $s_{\boldsymbol{W}}\left(\boldsymbol{x}_i, y\right)$ is a convex function of $\boldsymbol{W}$. This comes from the facts that:

1. $\boldsymbol{\beta}^\top \boldsymbol{W}_y \boldsymbol{x}_i$ is a linear function of $\boldsymbol{W}$ for every $\boldsymbol{\beta}$, so that $\sup_{\boldsymbol{\beta} \in \Omega_p} \boldsymbol{\beta}^\top \boldsymbol{W}_y \boldsymbol{x}_i$ is a convex function of $\boldsymbol{W}$ [Boyd and Vandenberghe, 2004];
2. since $\boldsymbol{\beta}^\top \boldsymbol{W}_y \boldsymbol{x}_i$ is a continuous function of $\boldsymbol{\beta}$, and Ω_p is non-empty and closed, we also have $\sup_{\boldsymbol{\beta} \in \Omega_p} \boldsymbol{\beta}^\top \boldsymbol{W}_y \boldsymbol{x}_i = \max_{\boldsymbol{\beta} \in \Omega_p} \boldsymbol{\beta}^\top \boldsymbol{W}_y \boldsymbol{x}_i$.

Note also that if we define the vector-valued function $\omega(\boldsymbol{x}_i, y) \triangleq \beta_{\boldsymbol{W}_y}(\boldsymbol{x}_i)^\top \boldsymbol{W}_y$ to indicate the lin-

ear model $\omega(\boldsymbol{x}_i, y)$ induced by the optimal mixing vector $\beta_{\boldsymbol{W}_y}(\boldsymbol{x}_i)$ (defined as in (4.13)), the prediction function of the ML3 algorithm can compactly be written as $\hat{y}_i = \arg\max_{y \in \mathcal{Y}} \omega(\boldsymbol{x}_i, y)^\top \boldsymbol{x}_i$. The prediction $\hat{y}_i$ is thus computed using linear models $\omega(\boldsymbol{x}_i, y)$ that vary depending on the location of the point $\boldsymbol{x}_i$ in the feature space. Similarly to Ladicky and Torr [2011], the proposed prediction function can be thus considered a locally linear function. However, differently from the approach used by Ladicky and Torr [2011] (where $\beta(\boldsymbol{x}_i)$ is independent of the class label and fixed in advance via manifold learning), in this work the sample and class specific mixing vector $\beta_{\boldsymbol{W}_y}(\boldsymbol{x}_i)$ is directly chosen to maximize the score $s_{\boldsymbol{W}}(\boldsymbol{x}_i, y)$, within a multiclass latent SVM framework. The proposed approach is thus named *Multiclass Latent Locally Linear SVM (ML3)*.

Following the latent SVM framework (equation (4.4)), we define the objective function of the ML3 algorithm, as

$$\min_{\boldsymbol{W}, \boldsymbol{\xi}} \frac{\lambda}{2} \|\boldsymbol{W}\|_F^2 + \frac{1}{n} \sum_{i=1}^{n} \xi_i \tag{4.19a}$$

$$\text{s.t. } 1 + \max_{y \in \mathcal{Y} \setminus \{y_i\}} s_{\boldsymbol{W}}(\boldsymbol{x}_i, y) - s_{\boldsymbol{W}}(\boldsymbol{x}_i, y_i) \leq \xi_i, \qquad i = 1, \ldots, n \tag{4.19b}$$

$$\xi_i \geq 0, \qquad i = 1, \ldots, n \tag{4.19c}$$

where $\|\cdot\|_F$ is the Frobenius norm. The objective function (4.19) is not convex w.r.t. $\boldsymbol{W}$ because, as noted above, $-s_{\boldsymbol{W}}(\boldsymbol{x}_i, y_i)$ is a concave function of $\boldsymbol{W}_{y_i}$, and not just a linear one. However, we can decompose (4.19b) into the difference of two convex functions: $1 + \max_{y \in \mathcal{Y} \setminus \{y_i\}} s_{\boldsymbol{W}}(\boldsymbol{x}_i, y) - \xi_i$ and $s_{\boldsymbol{W}}(\boldsymbol{x}_i, y_i)$. To solve our problem we can thus make use of the CCCP optimization algorithm discussed in Section 4.3.2.

CCCP Iteration for ML3. The first step to solve the ML3 learning problem using CCCP is to compute the linearization of $s_{\boldsymbol{W}}(\boldsymbol{x}_i, y_i)$ (as a function of $\boldsymbol{W}$) around an arbitrary point $\boldsymbol{C} \in \mathbb{R}^{mc \times d}$

$$t_{\{s_{\boldsymbol{W}}(\boldsymbol{x}_i, y_i), \boldsymbol{C}\}} = s_{\boldsymbol{C}}(\boldsymbol{x}_i, y_i) + \nabla s_{\boldsymbol{C}}(\boldsymbol{x}_i, y_i) \cdot (\boldsymbol{W} - \boldsymbol{C}) \ .$$

Using the fact that Ω_p is closed, $\boldsymbol{\beta}^\top \boldsymbol{W}_y \boldsymbol{x}_i$ is a convex function of $\boldsymbol{W}$ for every $\boldsymbol{\beta}$, and applying Danskin's theorem [Bertsekas, 1999], one can see that

$$\nabla s_{\boldsymbol{C}}(\boldsymbol{x}_i, y)^\top = \Big[\underbrace{\boldsymbol{0}}_{1} \quad \cdots \quad \boldsymbol{0} \quad \underbrace{\boldsymbol{x}_i \beta_{\boldsymbol{C}_y}(\boldsymbol{x}_i)^\top}_{y} \quad \boldsymbol{0} \quad \cdots \quad \underbrace{\boldsymbol{0}}_{c} \Big],$$

where, as for $\boldsymbol{W}$, $\boldsymbol{C}_y$ indicates the y-th block of $\boldsymbol{C}$, while $\beta_{\boldsymbol{C}_y}(\boldsymbol{x}_i)$ is defined as in equation (4.13) and obtained by using the closed form solutions provided in Section 4.4.1 (see Lemma 4). We

can thus write

$$\begin{aligned} t_{\{s_{\boldsymbol{W}}(\boldsymbol{x}_i,y),\boldsymbol{C}\}} &= s_{\boldsymbol{C}}\left(\boldsymbol{x}_i,y\right)+\operatorname{Tr}\left(\boldsymbol{x}_i\beta_{\boldsymbol{C}_y}(\boldsymbol{x}_i)^\top\left(\boldsymbol{W}_y-\boldsymbol{C}_y\right)\right)\\ &= s_{\boldsymbol{C}}\left(\boldsymbol{x}_i,y\right)+\beta_{\boldsymbol{C}_y}(\boldsymbol{x}_i)^\top\boldsymbol{W}_y\boldsymbol{x}_i-\beta_{\boldsymbol{C}_y}(\boldsymbol{x}_i)^\top\boldsymbol{C}_y\boldsymbol{x}_i\\ &= \beta_{\boldsymbol{C}_y}(\boldsymbol{x}_i)^\top\boldsymbol{W}_y\boldsymbol{x}_i. \end{aligned}$$

Finally, by replacing $s_{\boldsymbol{W}}(\boldsymbol{x}_i,y_i)$ in (4.19) with $t_{\{s_{\boldsymbol{W}}(\boldsymbol{x}_i,y_i),\boldsymbol{W}^t\}}$ and switching back to the unconstrained formulation of (4.19), we can now compactly write the objective function of the $t+1$ CCCP iteration for the ML3 algorithm as

$$\boldsymbol{W}^{t+1}=\operatorname*{arg\,min}_{\boldsymbol{W}}\frac{\lambda}{2}\left\|\boldsymbol{W}\right\|_F^2+\frac{1}{n}\sum_{i=1}^{n}\ell\left(\boldsymbol{W},\boldsymbol{W}^t,\boldsymbol{x}_i,y_i\right), \tag{4.20a}$$

$$\ell\left(\boldsymbol{W},\boldsymbol{W}^t,\boldsymbol{x}_i,y_i\right)=\left|1+\max_{y\in\mathcal{Y}\setminus\{y_i\}}s_{\boldsymbol{W}}(\boldsymbol{x}_i,y)-t_{\{s_{\boldsymbol{W}}(\boldsymbol{x}_i,y_i),\boldsymbol{W}^t\}}\right|_+. \tag{4.20b}$$

Implementation. In order to efficiently optimize (4.20) we opt to make use of just one epoch of SGD, using an algorithm similar to the one in Shalev-Shwartz et al. [2007], and some known strategies to accelerate convergence. Although one epoch of SGD is not guaranteed to minimize the objective function (which is needed for the convergence of CCCP), we observed that in practice it is enough and its efficiency is especially compelling for large-scale problems.

Following Shalev-Shwartz et al. [2007], at each step of the SGD procedure we draw at random a training sample $(\boldsymbol{x}_i,y_i)$, replace the objective in (4.20) with its stochastic approximation using only $\left(\boldsymbol{x}_i,y_i\right)$:

$$\frac{\lambda}{2}\left\|\boldsymbol{W}\right\|_F^2+\ell\left(\boldsymbol{W},\boldsymbol{W}^t,\boldsymbol{x}_i,y_i\right), \tag{4.21}$$

and perform a sub-gradient descent step. Applying again Danskin's theorem [Bertsekas, 1999], the subgradient $\tilde{\nabla}_{i,\boldsymbol{W}_y}$ of (4.21) w.r.t. $\boldsymbol{W}_y$ can be written as

$$\lambda\boldsymbol{W}_y+\mathbf{1}\left(\ell\left(\boldsymbol{W},\boldsymbol{W}^t,\boldsymbol{x}_i,y_i\right)>0\right)\left(\mathbf{1}\left(y=\check{y}_i\right)\beta_{\boldsymbol{W}_{\check{y}_i}}(\boldsymbol{x}_i)-\mathbf{1}\left(y=y_i\right)\beta_{\boldsymbol{W}^t_{y_i}}(\boldsymbol{x}_i)\right)\boldsymbol{x}_i^\top,$$

where $\check{y}_i\triangleq\operatorname{arg\,max}_{y\in\mathcal{Y}\setminus\{y_i\}}s_{\boldsymbol{W}}\left(\boldsymbol{x}_i,y\right)$. As it can be seen, only the components of $\boldsymbol{W}_{\check{y}_i}$ and $\boldsymbol{W}_{y_i}$ for which the associated element of $\beta_{\boldsymbol{W}_{\check{y}_i}}(\boldsymbol{x}_i)$ and $\beta_{\boldsymbol{W}^t_{y_i}}(\boldsymbol{x}_i)$ is different from zero are updated using $\boldsymbol{x}_i$. These components are updated with a weight provided by $\beta_{\boldsymbol{W}_{\check{y}_i}}(\boldsymbol{x}_i)$ and $\beta_{\boldsymbol{W}^t_{y_i}}(\boldsymbol{x}_i)$, which in turn is related to their confidence on $\boldsymbol{x}_i$. Each component tends thus to be trained only with samples for which its confidence is higher than the other components for the same class.

A strategy to accelerate the convergence of SGD is to bound the norm of the optimal classifier and use it to normalize the solution during training [Shalev-Shwartz et al., 2007]. Specifically, let O^* be the value of the objective function in (4.20) obtained with the optimal classifier $\boldsymbol{W}^*$. Then $O^*\geq\frac{\lambda}{2}\left\|\boldsymbol{W}^*\right\|_F^2$; moreover $O^*\leq 1$ (the value of the objective function evaluated in $\boldsymbol{W}=\mathbf{0}$),

Algorithm 2 Stochastic Gradient Descent for the $t+1$ CCCP iteration of ML3

Input: $\boldsymbol{X}$, $\boldsymbol{y}$, $\boldsymbol{W}^t$, λ, s_0, $lastIteration$
Output: $\boldsymbol{W}^{t+1}$, $\bar{\boldsymbol{W}}$

$\boldsymbol{W}^{t+1} \leftarrow \boldsymbol{W}^t$
$\bar{\boldsymbol{W}} \leftarrow \boldsymbol{0}$
for $s = 1 \ldots, n$ **do**
  $\eta \leftarrow \frac{1}{\lambda(s+s_0)}$
  $\boldsymbol{W}_y^{t+1} \leftarrow \boldsymbol{W}_y^{t+1} - \eta \tilde{\nabla}_{s,\boldsymbol{W}_y^{t+1}},\ \forall y = 1, \ldots, c$
  $\boldsymbol{W}^{t+1} \leftarrow \boldsymbol{W}^{t+1} \min\left\{1, \frac{\sqrt{2}/\sqrt{\lambda}}{\|\boldsymbol{W}^{t+1}\|_F}\right\}$
  if $lastIteration$ **then**
    $\bar{\boldsymbol{W}} \leftarrow \frac{(s-1)\bar{\boldsymbol{W}} + \boldsymbol{W}^{t+1}}{s}$
  end if
end for

so that we have $\|\boldsymbol{W}^*\|_F \leq \sqrt{2/\lambda}$. The norm of the optimal classifier is thus bounded, and a projection rule of the form

$$\boldsymbol{W} \leftarrow \boldsymbol{W} \min\left\{1, \frac{\sqrt{2}/\sqrt{\lambda}}{\|\boldsymbol{W}\|_F}\right\}$$

enforces this condition.

Secondly, Bordes et al. [2009] proposed to use an additional constant term, s_0, in the learning rate, to prevent the first updates from producing matrices $\boldsymbol{W}$ with an implausibly large norm. As a side effect, this also allows to use $\boldsymbol{W}^t$ to initialize the algorithm, when computing $\boldsymbol{W}^{t+1}$. Also, as underlined in Felzenszwalb et al. [2010], a careful initialization of $\boldsymbol{W}^0$ might be necessary to avoid selecting unreasonable values for $\beta_{\boldsymbol{W}_{y_i}^0}(\boldsymbol{x}_i)$ in the first iteration. To this end, we propose the following procedure: 1) randomly initialize $\tilde{\beta}_{\boldsymbol{W}_y^0}(\boldsymbol{x}_i) \in \Omega_p$ for all training samples and classes; 2) keeping all the latent variables for all classes fixed to $\tilde{\beta}_{\boldsymbol{W}_y^0}(\boldsymbol{x}_i)$, initialize $\boldsymbol{W}^0$ with one epoch of stochastic gradient descent; 3) fix $s_0 = 2n$. Although still random, this procedure forces the CCCP procedure to start from a relatively good solution, increasing the chances to converge to a good local minimum. To speed up convergence, at the end of each CCCP iteration we also increment s_0 by $2n$. Finally, in the last CCCP iteration we take the average of all the generated solutions and use it as the final solution. The complete training algorithm for one CCCP iteration is summarized in Algorithm 2. Its complexity is $\mathrm{O}(n\,d\,m\,c)$.

Prediction. As discussed in Section 4.4.1, for any model $\boldsymbol{W}$ and any sample $\boldsymbol{x}$, the optimal mixing vector $\beta_{\boldsymbol{W}_y}(\boldsymbol{x}_i)$ can be obtained using an analytical solution. Specifically, when predicting the score $s_{\boldsymbol{W}}(\boldsymbol{x}_j, y)$ for a test sample j and candidate class y, $\beta_{\boldsymbol{W}_y}(\boldsymbol{x}_j)$ is computed according to equation (4.14), or (4.15) (depending to the value of p). Differently from the manifold learning approaches, the only parameter of the model is thus the matrix $\boldsymbol{W}$. Moreover, for prediction purposes the explicit computation of $\beta_{\boldsymbol{W}_y}(\boldsymbol{x}_j)$ is unnecessary. Indeed, as shown

by Lemma 4, plugging the analytical expression of $\beta_{\boldsymbol{W}_y}(\boldsymbol{x}_i)$ back into $s_{\boldsymbol{W}}\left(\boldsymbol{x}_j, y\right)$ we can directly obtain the value of $s_{\boldsymbol{W}}\left(\boldsymbol{x}_j, y\right)$ at the optima:

$$s_{\boldsymbol{W}}\left(\boldsymbol{x}_j, y\right) = \left\| \left(\boldsymbol{W}_y \boldsymbol{x}_j\right)^+ \right\|_q, \tag{4.22}$$

where, again, $q = p/(p-1)$ and $\boldsymbol{a}^+$ is the element-wise maximum between the vectors $\boldsymbol{a}$ and $\boldsymbol{0}$. This provides us with a very efficient prediction rule, whose complexity/sample is $\mathrm{O}(d\,m\,c)$.

An alternative interpretation of the ML3 algorithm. It is worth noting that although the mixing vectors $\boldsymbol{\beta}$ have completely disappeared from (4.22), the prediction rule still has an intuitive interpretation. Suppose indeed that we are given a vector of scores $\boldsymbol{W}_y \boldsymbol{x}_j$, obtained by applying a set of models $\boldsymbol{W}_y$ to a sample $\boldsymbol{x}_j$. A possible way to measure the confidence of this set of models on $\boldsymbol{x}_j$ could be to compute the norm of $\boldsymbol{W}_y \boldsymbol{x}_j$, using $\left\| \boldsymbol{W}_y \boldsymbol{x}_j \right\|_q$ as a (non-linear) scoring function. However, with this choice also the negative values in $\boldsymbol{W}_y \boldsymbol{x}_j$ would positively contribute to the score of $\boldsymbol{W}_y$ on $\boldsymbol{x}_j$, resulting in a wrong estimation of the confidence. A workaround to the problem could be to define the confidence of the set of models $\boldsymbol{W}_y$ on $\boldsymbol{x}_i$ as the norm of only the positive elements of $\boldsymbol{W}_y \boldsymbol{x}_j$, simplifying also the interpretation of the decision function and of the learned models. This corresponds exactly to the closed form scoring function that we obtained in (4.22), confirming the importance of the non-negativity constraints in Ω_p.

4.5 Explicit feature maps and visualizations

In this Section we make use of two synthetic 2D datasets (a XOR dataset and the "Banana" [Frank and Asuncion, 2010] dataset) to provide some insights on the internal functioning of the ML3 algorithm. We explicitly compute the optimal value of the latent variable for the predicted class of each sample,

$$\{\hat{y}_i, \beta_{\boldsymbol{W}_{\hat{y}_i}}(\boldsymbol{x}_i)\} = \underset{\substack{y \\ \boldsymbol{\beta} \in \Omega_p}}{\arg\max}\, \boldsymbol{\beta}^\top \boldsymbol{W}_y \boldsymbol{x}_i$$

and use it to output the feature map $\psi\left(\boldsymbol{x}_i, \beta_{\boldsymbol{W}_{\hat{y}_i}}(\boldsymbol{x}_i)\right)$. In this respect, we use ML3 as a feature extractor and visualize, together with the decision boundary in the input space, the optimal beta and the associated feature map for each sample. Please note that while the ML3 algorithm computes a different $\beta_{\boldsymbol{W}_y}(\boldsymbol{x}_i)$ - and thus a different feature map - for each class, we will plot only $\psi\left(\boldsymbol{x}_i, \beta_{\boldsymbol{W}_{\hat{y}_i}}(\boldsymbol{x}_i)\right)$. This makes the feature mappings easier to visualize (as there is only a single explicit feature representation for any given sample), while retaining the predictive power of the full set of mappings used by the original algorithm. The reason for this lies in the fact that for any $y \neq \hat{y}_i$, by construction

$$\beta_{\boldsymbol{W}_{\hat{y}_i}}(\boldsymbol{x}_i)^\top \boldsymbol{W}_y \boldsymbol{x}_i \leq \beta_{\boldsymbol{W}_y}(\boldsymbol{x}_i)^\top \boldsymbol{W}_y \boldsymbol{x}_i \leq \beta_{\boldsymbol{W}_{\hat{y}_i}}(\boldsymbol{x}_i)^\top \boldsymbol{W}_{\hat{y}_i} \boldsymbol{x}_i,$$

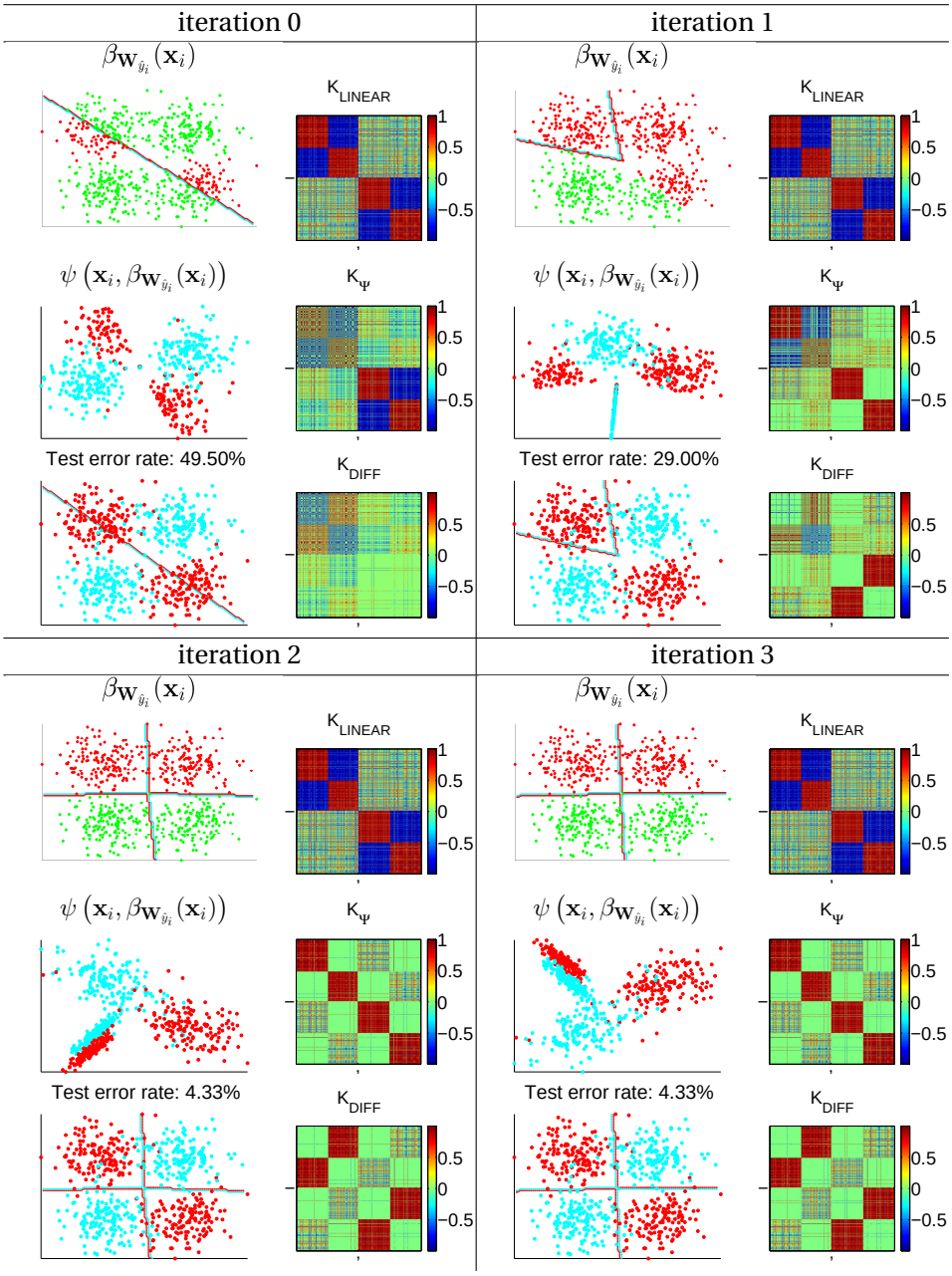

Figure 4.2 – Training sequence on a synthetic XOR dataset, using two components and $p = 1$. For each experiment we color encode the sample-to-component assignments (first row of the first column), with the RGB values set according to the first three components of $\beta_{\boldsymbol{W}_{\hat{y}_i}}(\boldsymbol{x}_i)$. We also plot a 2D projection of $\psi\left(\boldsymbol{x}_i, \beta_{\boldsymbol{W}_{\hat{y}_i}}(\boldsymbol{x}_i)\right)$ with the ground-truth label color encoded in red and cyan (second row of the first column of each experiment). In the third row of the first column we plot the resulting decision boundary in the original input space, with the ground-truth label color encoded again in red and cyan. Finally, on the second column of each experiment we plot the normalized Gramian matrices computed using the original data (first row), using $\psi\left(\boldsymbol{x}_i, \beta_{\boldsymbol{W}_{\hat{y}_i}}(\boldsymbol{x}_i)\right)$ (second row) and difference between the latter and the former (third row). In the Gramian matrices the samples are ordered according to their ground-truth labels.

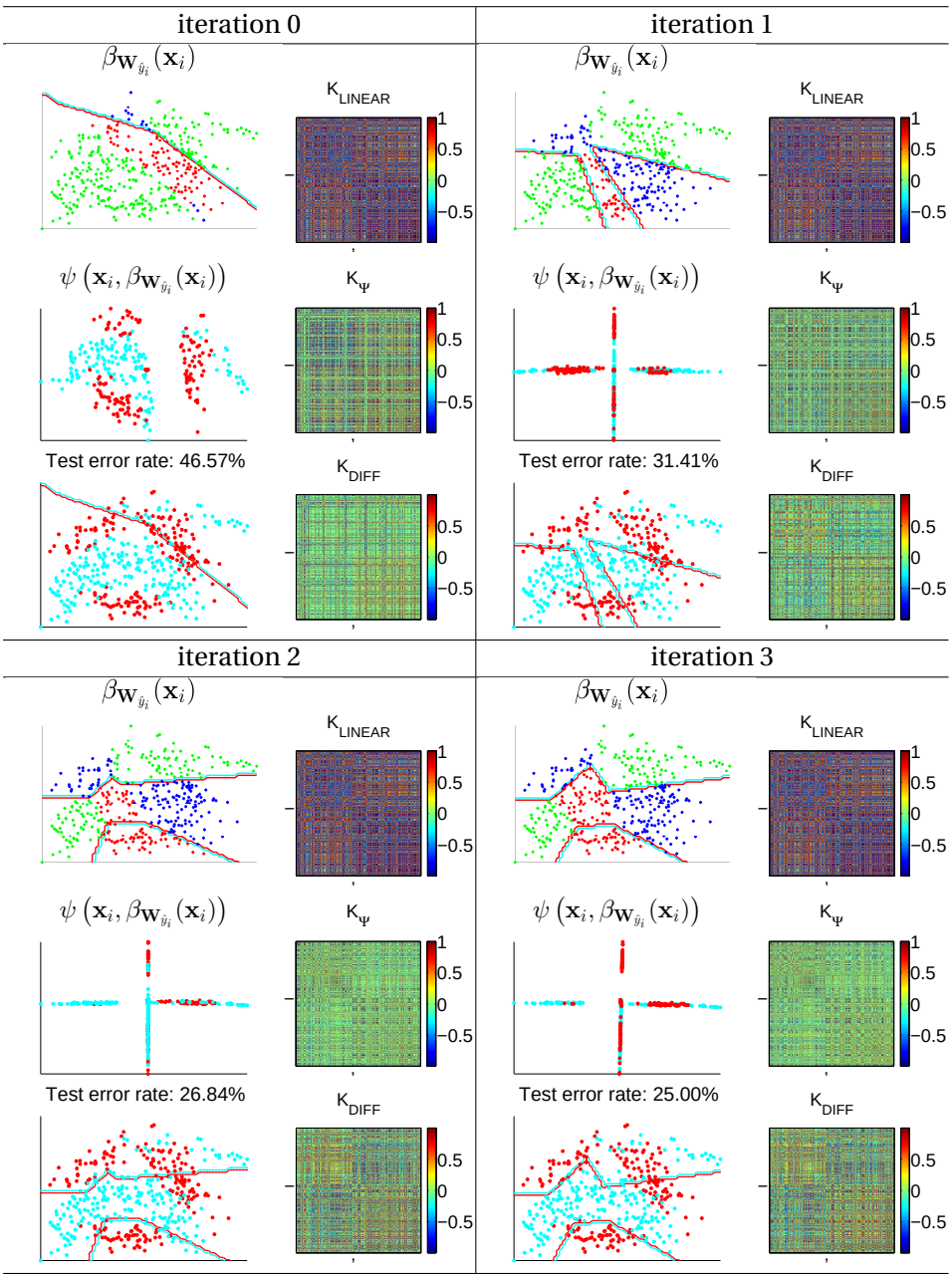

Figure 4.3 – Training sequence on the Banana dataset, using three components and $p = 1$. For each experiment we color encode the sample-to-component assignments (first row of the first column), with the RGB values set according to the first three components of $\beta_{\boldsymbol{W}_{\hat{y}_i}}(\boldsymbol{x}_i)$. We also plot a 2D projection of $\psi\left(\boldsymbol{x}_i, \beta_{\boldsymbol{W}_{\hat{y}_i}}(\boldsymbol{x}_i)\right)$ with the ground-truth label color encoded in red and cyan (second row of the first column of each experiment). In the third row of the first column we plot the resulting decision boundary in the original input space, with the ground-truth label color encoded again in red and cyan. Finally, on the second column of each experiment we plot the normalized Gramian matrices computed using the original data (first row), using $\psi\left(\boldsymbol{x}_i, \beta_{\boldsymbol{W}_{\hat{y}_i}}(\boldsymbol{x}_i)\right)$ (second row) and difference between the latter and the former (third row). In the Gramian matrices the samples are ordered according to their ground-truth labels.

Therefore, by fixing the optimal beta for each sample $\boldsymbol{x}_i$ to $\beta_{\boldsymbol{W}_{\hat{y}_i}}(\boldsymbol{x}_i)$, the value of $\beta_{\boldsymbol{W}_{\hat{y}_i}}(\boldsymbol{x}_i)^\top \boldsymbol{W}_y \boldsymbol{x}_i$ will be maximal for $y = \hat{y}_i$ and the prediction of the ML3 algorithm will be preserved.

A visualization of a short learning sequence using XOR and Banana datasets, is shown in Figures 4.2 and 4.3 (where $p = 1$ and $m = 2$ for XOR and $m = 3$ for Banana). In each experiment, corresponding to a different CCCP iteration, we plot $\beta_{\boldsymbol{W}_{\hat{y}_i}}(\boldsymbol{x}_i)$, a 2D projection of $\psi\left(\boldsymbol{x}_i, \beta_{\boldsymbol{W}_{\hat{y}_i}}(\boldsymbol{x}_i)\right)$, the resulting decision boundary in the input space, the normalized Gramian matrices obtained using the input features ($\mathrm{K}_{\mathrm{LINEAR}}$) and using $\psi\left(\boldsymbol{x}_i, \beta_{\boldsymbol{W}_{\hat{y}_i}}(\boldsymbol{x}_i)\right)$ (K_Ψ), and the difference between the latter and the former ($\mathrm{K}_{\mathrm{DIFF}}$).

As it can be seen, as the ML3 training progresses, the components tend to specialize to specific parts of the space and the sample-to-component assignments cluster accordingly (with each class-specific cluster determined by a row of $\boldsymbol{W}_y$). As a result, the samples in the feature space ψ become increasingly linearly separable, while K_Ψ exhibits an increased intra-class sample similarity and a reduced inter-class similarity. After two CCCP iterations on the XOR dataset, the ML3 algorithm has learned a model with two components per class, each one covering a well defined region of the input space. With this specialization of the linear components, the XOR problem becomes linearly separable in the feature space $\psi\left(\boldsymbol{x}_i, \beta_{\boldsymbol{W}_{\hat{y}_i}}(\boldsymbol{x}_i)\right)$, and the global decision boundary in the input space almost perfect. Similar results are obtained on the Banana dataset, where it is also possible to note how already after the initialization procedure, the decision boundary shows some form of non-linearity. This is due to the fact that $\boldsymbol{W}^0$ is trained with a set of fixed but random $\tilde{\beta}_{\boldsymbol{W}_y^0}(\boldsymbol{x}_i)$ (a different one for each sample), resulting in a non-linear mapping. Moreover, since during prediction we make use of $\beta_{\boldsymbol{W}_y^0}(\boldsymbol{x}_i)$ (instead of the randomly fixed $\tilde{\beta}_{\boldsymbol{W}_y^0}(\boldsymbol{x}_i)$), the model produce also an early clustering of the input space. As the learning progresses, each class-specific cluster moves towards more discriminative positions. For example, for the red class of the Banana dataset, the three components learned by ML3 roughly correspond to each of the three clusters that can be spotted by a visual inspection of the sample distribution for this class. On this dataset it is also possible to note, especially in Figure 4.4, how as the training progresses, the Gramian matrices K_Ψ computed using $\psi\left(\boldsymbol{x}_i, \beta_{\boldsymbol{W}_{\hat{y}_i}}(\boldsymbol{x}_i)\right)$ tend to become block diagonal, in agreement with the ground-truth labels.

4.6 Hyper-parameters setting

In this Section we provide a brief discussion of the role of the parameter p in the ML3 algorithm. For a complete discussion of the role of this parameter from a function approximation point of view please refer to Section 4.4.1.

When $p = 1$, for each class y the optimal $\beta_{\boldsymbol{W}_y}(\boldsymbol{x}_i)$ assigns all the available weight to the single most confident positively scoring component (see (4.15)). This enforces a hard-clustering of the input space into well separated regions covered by a single component. The boundary between clusters is sharp and the classification boundary non-smooth. Similarly, when $p \to \infty$

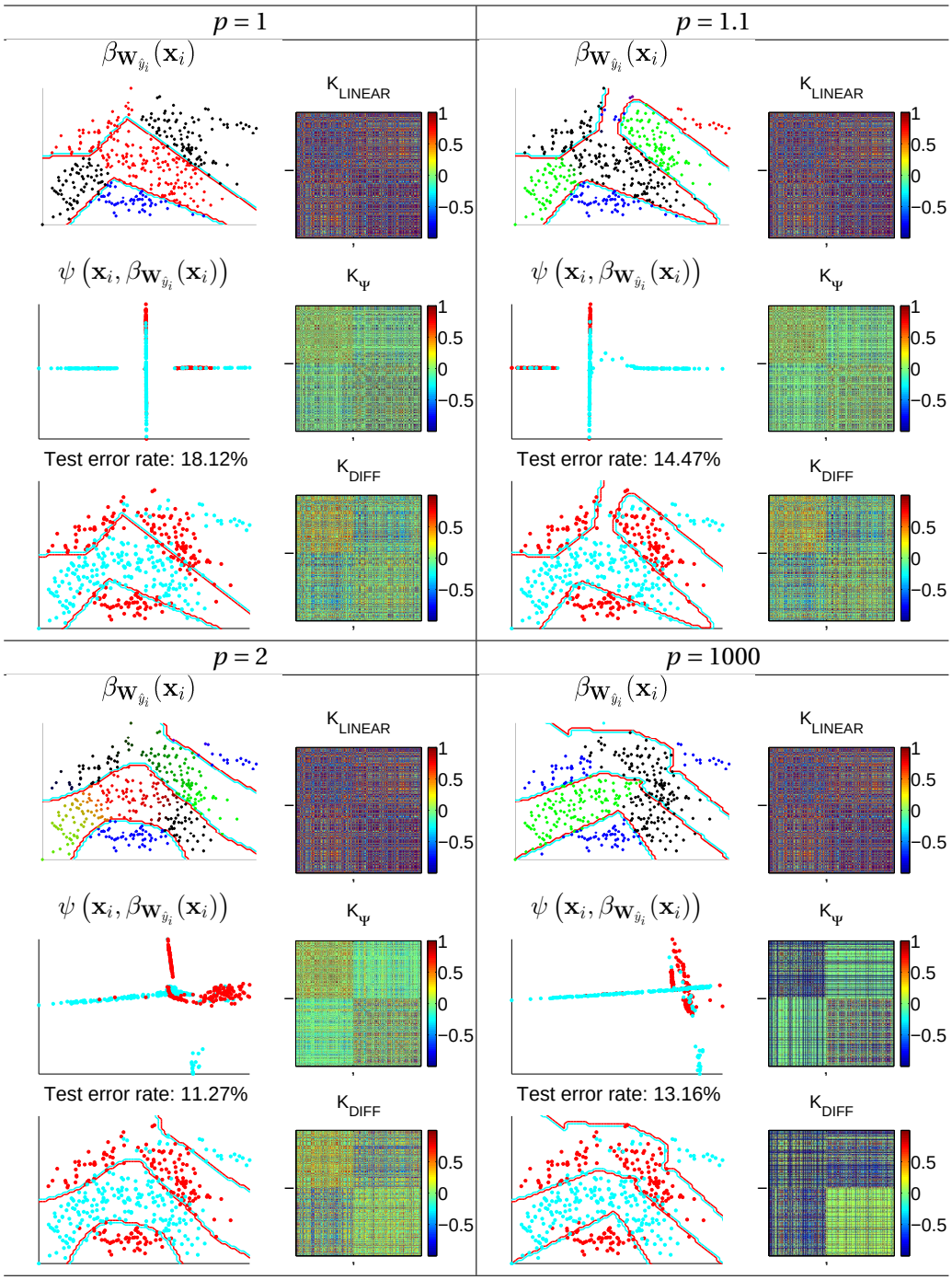

Figure 4.4 – Effect of varying the parameter p in the set $\{1, 2, 1000\}$, using $m = 5$ on Banana dataset. As in Figures 4.2 and 4.3, for each experiment we color encode the sample-to-component assignments, we plot a 2D projection of $\psi(\boldsymbol{x}_i, \beta_{\boldsymbol{W}_{\hat{y}_i}}(\boldsymbol{x}_i))$, the resulting decision boundary and the normalized Gramian matrices computed using the original data, using $\psi(\boldsymbol{x}_i, \beta_{\boldsymbol{W}_{\hat{y}_i}}(\boldsymbol{x}_i))$ and difference between the latter and the former.

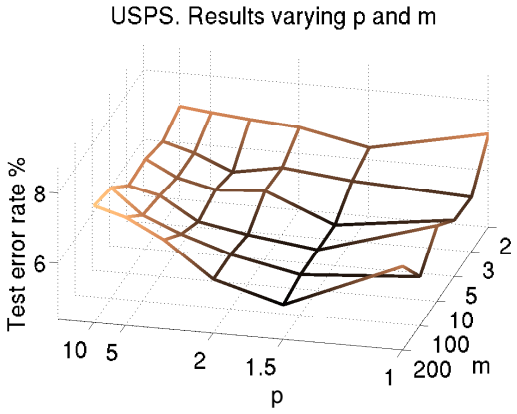

Figure 4.5 – Performance on USPS, when varying both p and the number of components m.

all the weights tend to 1, except for those predicting negatively, which receive a sharp 0 (see (4.14)). This again results in hard boundaries between clusters, with sharply defined intersecting areas and a non-smooth decision boundary. Finally, when $p = 2$, each component is given a weight proportional to its confidence, resulting in smooth transitions between components, and smooth decision boundaries. A visualization of this effect on the Banana dataset can be seen in Figure 4.4, where as before we plot the sample-to-component assignments, the classification results, the feature mappings $\psi\left(\boldsymbol{x}_i, \beta_{\boldsymbol{W}_{\hat{y}_i}}(\boldsymbol{x}_i)\right)$ and the resulting kernel matrices, for $p \in \{1, 1.1, 2, 1000\}$.

Although $p = 2$ is a reasonable candidate to produce smooth decision boundaries, a lower value of p will tend to emphasize the locality of the components. Indeed, as discussed in Section 4.4.1 the L2C codes obtained with values of p in $[1, 2]$ achieve a compromise between minimization of the localization error and minimization of the reconstruction error. This may be important, for example, whenever the number of components is very low and it is thus necessary to obtain components that are well localized in space. Moreover, also in case of a very high number of components, the locality-induced sparsity of $\beta_{\boldsymbol{W}_y}(\boldsymbol{x}_i)$ might help to reduce the noise due to predictions of components with low-confidence. In Figure 4.5 we plot the testing performances of the ML3 algorithm when p varies between 1 and 10, on the USPS character recognition task (see Section 4.7), while also varying the number of components. As we can see, setting $p = 1.5$ results in the best performance, especially when m is large. Moreover, as it is possible to see, with values of p in $(1, 2]$ the performance of ML3 does not severely degrade when the number of components is increased. For these reasons, we use $p = 1.5$ as our default value.

4.7 Experiments

We assess our approach by running experiments on ten standard UCI machine learning datasets [Frank and Asuncion, 2010], three character recognition datasets [Hull, 1994; Frank and Asuncion, 2010; LeCun et al., 1998] and one large binary dataset [Joachims, 2006]. This set

of datasets largely overlaps with the ones used in [Gönen and Alpaydin, 2008; Yu and Zhang, 2010; Ladicky and Torr, 2011; Yu et al., 2009; Zhang et al., 2011], with a similar scale in terms of number of samples and classes. Furthermore, we evaluate the performance of our approach on the scene recognition datasets used in this thesis: the MIT-Indoor-67, the 15-Scenes and the UIUC-Sports datasets.

We compare our algorithm against state of the art manifold learning techniques, such as Locally Linear SVM (LLSVM) [Ladicky and Torr, 2011], General OCC (*G-OCC*) and Class-specific OCC (*C-OCC*) [Zhang et al., 2011]. Moreover, we compare against Adaptive Multi-hyperplane Machines (*AMM*) [Wang et al., 2011b] and against explicit feature map approximation of the Gaussian kernel using random features (*R-Feat*) [Rahimi and Recht, 2007]. Finally, we compare against standard learning algorithms, such as: linear SVM with a one vs all multiclass extension (*Linear*), multiclass linear SVM (*MC-Linear*) [Crammer and Singer, 2001] and SVM with Gaussian kernel (*Gaussian*). For completeness of results we also perform experiments using the L2C coding approach described in Section 4.4.1 (i.e. fixing the mixing coefficients using the L2C codes) and by using the dictionaries produced by the L2C codes to initialize the models of the ML3 algorithm (indicated as *ML3+I*).

In all our benchmarks the best regularization parameter for each algorithm is selected by performing 5-fold cross-validation on each training split. All the multi-component classifiers are compared using the same number of components, except where explicitly mentioned. Although for best performance the value of p should be tuned for each considered task, for the ML3 algorithm and the L2C coding we use the default value $p = 1.5$ (discussed in Section 4.6)for all the following experiments, except when we explicitly analyze how the performance varies w.r.t. p. For the experiments with the L2C encodings we use the same multiclass implementation of ML3 , but we fix the mixing coefficients in advance by performing the dictionary learning procedure described in Section 4.4.1. As for the other algorithms, ML3 is developed using a mixed Maltab/C++ implementation, with the main algorithm being implemented in a mex file [4]. For LLSVM we use the same manifold learning settings as in Ladicky and Torr [2011] (i.e. encoding based on inverse Euclidean distances to the 8 nearest cluster centers), with an implementation privately provided by the authors. For G-OCC and C-OCC we use the implementations available on the website of the authors. As underlined in Zhang et al. [2011], the manifold learning step of OCC consists of learning a set of basis. This limits the maximum number of components used by OCC to be equal to the rank of the data matrix. We will specifically remark the cases in which this limitation results in a different number of components with respect to ML3, or other baselines. For AMM we use the implementation freely available on the web, with the component pruning threshold and iterations set to $c = 10$ and $k = 10,000$, as recommended by the authors [Wang et al., 2011b] (for a brief description of the AMM algorithm, please refer to Section 2.4.2). Please note that due to the component pruning technique used by AMM we do not have explicit control on the actual number of components used by AMM. For this algorithm we can only limit the

4. The software for the ML3 algorithm is freely available at https://github.com/idiap/ML3.

Table 4.1 – Average test error rate and ranking on the UCI benchmark datasets.

	ML3+I	Gauss	ML3	R-Feat	G-OCC	C-OCC	LLSVM	L2C	AMM	L-MKL	Linear
Banana	11.24	**10.76**	11.25	12.18	34.56	35.51	12.63	13.48	16.11	11.69	47.53
German	**23.35**	23.59	23.65	24.55	24.13	25.57	27.75	25.15	35.39	27.19	24.61
Heart	16.56	16.89	18.00	18.44	26.78	24.56	**15.78**	17.56	16.56	18.67	16.78
Ionosphere	12.31	10.09	11.88	**9.66**	10.85	11.79	14.02	16.58	12.22	14.70	14.87
Liver	33.04	33.30	32.17	32.70	**29.57**	31.91	35.13	33.13	34.35	33.04	33.13
PIMA	**22.11**	23.28	22.50	23.48	24.80	26.88	23.01	23.44	22.97	25.16	22.50
Ringnorm	7.48	**1.56**	7.59	1.82	19.07	18.72	3.02	8.87	11.73	13.03	23.30
Sonar	29.29	30.57	29.86	29.14	29.86	**28.29**	31.29	32.00	32.43	32.86	32.86
Spambase	11.45	**8.42**	11.23	10.24	18.95	15.21	23.21	11.02	26.92	9.39	11.84
WDBC	**9.83**	9.87	10.86	11.93	11.46	11.20	12.58	12.40	11.33	13.56	12.92
Avg. Rank	3.20	3.50	4.10	4.50	6.60	6.70	7.00	7.10	7.40	7.80	8.10

maximum number of components used (i.e. not to exceed the number of components used by the other algorithms). To train the linear SVM we use LIBLINEAR [Fan et al., 2008], while for the Gaussian SVM we use LIBSVM [Chang and Lin, 2011], with the scaling parameter of the kernel set using the inverse of the average pairwise distance of a subset of the training data. For R-Feat we use our implementation, with the same scaling parameter as for the Gaussian SVM. For this algorithm, the dimensionality of the approximation is set to to $m \times d$ (where d is the dimensionality of the input features for a given dataset), so that the number of parameters in the model is approximatively the same as the number of parameters used by the other multi-component algorithms.

4.7.1 Benchmark datasets.

The first benchmark of this Chapter consists of a set of ten two-class datasets from the UCI collection [Frank and Asuncion, 2010]. For this benchmark we also compare against Localized Multiple Kernel Learning (L-MKL) [Gönen and Alpaydin, 2008], using the MATLAB implementation available on the website of the authors and m linear kernels. For the Banana dataset we use 400 samples for training and the remaining for testing. For all the other datasets two thirds of the samples are used as a training set, while the remaining third is used as a test set. As explained before, each training set is divided in five folds that are used to select the regularization parameter. Each experiment was repeated ten times on ten different training / testing splits [5] and the average test error rate is reported in Table 4.1. For these experiments the number of components was fixed to $m = 10$ and, in order to learn a bias for each components for L2C, ML3 and OCC, we concatenated 1 to each instance vector. For the other algorithms (Linear, MC-Linear, AMM, LLSVM and L-MKL) we use the bias support directly provided by the default implementation. Please note that the dimensionality of Banana, Liver, PIMA and WDBC is lower than 10, resulting in a reduced number of components (2, 6, 8 and 9, respectively) for the OCC encodings.

5. For L-MKL we had to reduce the number of splilts and CV-folds, as it resulted to be unable to complete the benchmark in a reasonable time. For similar reasons the algorithm was not tested on larger datasets.

Table 4.2 – Average training times (in seconds) on the UCI benchmark datasets.

	ML3+I	Gauss	ML3	R-Feat	G-OCC	C-OCC	LLSVM	L2C	AMM	L-MKL	Linear
Banana	0.082	0.025	0.089	0.009	0.011	0.013	0.006	0.063	0.082	55.16	0.000
German	0.205	0.049	0.173	0.055	0.076	0.152	0.043	0.144	0.204	29.76	0.003
Heart	0.059	0.003	0.057	0.004	0.013	0.027	0.010	0.043	0.054	0.535	0.000
Ionosphere	0.067	0.004	0.062	0.008	0.030	0.063	0.013	0.062	0.076	1.059	0.001
Liver	0.071	0.015	0.060	0.004	0.009	0.015	0.007	0.044	0.056	4.839	0.000
PIMA	0.136	0.011	0.138	0.008	0.025	0.041	0.018	0.091	0.140	5.968	0.000
Ringnorm	1.118	0.694	1.077	0.622	0.452	0.840	0.203	0.886	1.439	4349	0.005
Sonar	0.060	0.003	0.056	0.030	0.032	0.068	0.007	0.060	0.052	2.374	0.000
Spambase	1.094	3.151	0.941	0.621	0.687	1.226	0.173	0.868	1.307	475.1	0.015
WDBC	0.093	0.009	0.102	0.010	0.021	0.035	0.014	0.085	0.113	3.591	0.000
Total	2.985	3.963	2.754	1.371	1.355	2.481	0.494	2.346	3.524	4928	0.026

Table 4.3 – Average testing times (in seconds) on the UCI benchmark datasets.

	ML3+I	Gauss	ML3	R-Feat	G-OCC	C-OCC	LLSVM	L2C	AMM	L-MKL	Linear
Banana	0.007	0.025	0.006	0.005	0.002	0.001	0.004	0.011	0.002	0.319	0.001
German	0.001	0.014	0.001	0.003	0.001	0.001	0.001	0.001	0.001	0.018	0.000
Heart	0.000	0.001	0.000	0.001	0.000	0.000	0.001	0.001	0.000	0.006	0.000
Ionosphere	0.000	0.002	0.000	0.002	0.001	0.001	0.001	0.001	0.000	0.012	0.000
Liver	0.000	0.001	0.000	0.001	0.000	0.000	0.001	0.001	0.000	0.012	0.000
PIMA	0.001	0.005	0.001	0.001	0.000	0.001	0.001	0.001	0.000	0.015	0.000
Ringnorm	0.004	0.110	0.004	0.019	0.003	0.004	0.004	0.007	0.002	0.537	0.001
Sonar	0.000	0.002	0.000	0.002	0.001	0.001	0.001	0.001	0.000	0.006	0.000
Spambase	0.004	0.323	0.004	0.037	0.006	0.006	0.004	0.005	0.003	0.297	0.001
WDBC	0.001	0.002	0.001	0.001	0.000	0.001	0.001	0.001	0.000	0.009	0.000
Total	0.019	0.485	0.018	0.070	0.014	0.016	0.020	0.028	0.010	1.233	0.004

In Table 4.1 we report the performance of all the considered algorithms on all the UCI benchmark datasets, ordering the algorithms (form left to right) according to their average ranking in the benchmark. Please note that in order to make these results easily comparable with those reported in other papers, the performance measure used for these experiments is the error rate, and not the accuracy, as in the other Chapters of this thesis.

As expected, all the multi-component methods outperform (in terms of average ranking) the single-component linear SVM, with Localized MKL and Adaptive Multi-hyperplane Machines being the weakest multi-component performers. For the AMM model, the reason for this low performance could lie in the fact that this algorithm is optimized for very large scale data and the weight pruning technique can result in reduced performances. Amongst the encoding based methods (G-OCC, C-OCC, LLSVM and L2C), the most robust performer seems to be G-OCC, while LLSVM and L2C do not seem to perform particularly good. The best performers on this benchmark consistently result to be the ML3 algorithm, the Gaussian SVM and, partially, also the Random Features. As it can be seen, if the dictionaries obtained by L2C are used to initialize the ML3 models (the ML3+I baseline), the performance of the ML3 algorithm on this benchmark increases even above that of the Gaussian kernel.

	p=1	p=1.2	p=1.4	p=1.6	p=1.8	p=2
Banana	13.97	13.21	**11.74**	19.49	14.65	23.69
German	25.51	**23.83**	24.46	24.73	24.79	25.36
Heart	22.11	18.89	17.78	**17.11**	18.00	17.22
Ionosphere	16.15	15.47	16.41	16.15	**15.21**	15.73
Liver	36.26	34.61	33.65	**32.87**	33.39	34.00
PIMA	25.82	24.45	23.71	23.48	23.63	**23.12**
Ringnorm	11.89	12.27	9.91	8.18	7.01	**6.38**
Sonar	33.14	**29.43**	33.29	35.57	35.57	38.86
Spambase	14.86	11.72	**11.14**	11.20	11.17	11.17
WDBC	**11.76**	12.40	12.83	12.53	13.05	12.45
Avg. Rank	4.50	3.40	3.20	3.20	3.40	3.30

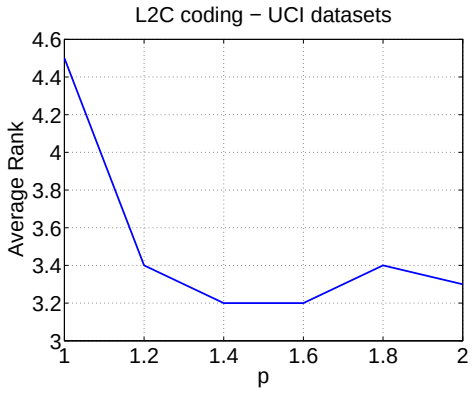

Figure 4.6 – Performance of the L2C coding on the UCI benchmark datasets. Left: average test error rate and ranking, varying p between 1 and 2. Right: average ranking, varying p between 1 and 2.

In Table 4.2 we report the average training times of all the considered methods, as obtained by averaging the times measured on the ten random splits, on a single thread of an Intel(R) Core(TM) i7-2600K, with 16GB of RAM. The training times for the multi-component classifiers (AMM, LLSVM, OCC, L2C and ML3) are measured using 100 training epochs (or CCCP iterations). When comparing the training and testing times of different methods please consider that the L-MKL classifier is implemented in MATLAB, while all the other methods have a C++ implementation (at least for the classifier). The training times for the Gaussian SVM and for R-Feat include the time required to estimate the kernel parameter (by computing the average pairwise distance on a subset of the training data).

As it can be seen, the fastest classifier is the Linear SVM, which can be trained on all the dataset in less than $\frac{1}{10}$ of a second. The multi-component method with the lowest training time results to be the LLSVM classifier (which can be trained on all the datasets in less than half a second), while other competitive approaches result to be C-OCC and R-Feat. On these small datasets the training times for the Gaussian SVM are typically lower than those measured for ML3, or C-OCC. However the total time required to train the Gaussian SVM on the full benchmark results to be higher than the one measured for the ML3 algorithm. This is due to the fact that on the largest datasets of the benchmark (i.e. Ringnorm and Spambase) the training times of the ML3 algorithm become close to (or lower than) the training times of the Gaussian SVM.

In Table 4.3 we also report the testing times for all the algorithms. As it can be seen, the fastest algorithm is once again linear SVM. The multi-component algorithms perform all very close to each other (except for L-MKL), with the fastest approach being AMM. As it can be seen, the testing times of the ML3 algorithm and of the other multi-component classifiers are consistently lower then those of the Gaussian SVM. Indeed, the total time required to test the ML3 algorithm results to be more than one order of magnitude lower than the total time required to test the Gaussian SVM.

Table 4.4 – Error rate and associated training and testing time (in seconds) of different algorithms. The results taken from other papers are reported with the citation. The results for multi-component approaches (LLSVM, OCC, L2C, AMM, ML3, ML3+I) are obtained by training the algorithms for 30 epochs (or CCCP iterations) with $p = 1.5$ and $m = 80$ for USPS, $m = 16$ for LETTER, $m = 90$ for MNIST, $m = 54$ for COVTYPE.

	USPS			LETTER			MNIST			COVTYPE		
Gaussian	4.88%	2.55s	1.88s	**3.15**%	3.20s	2.81s	1.78%	327s	284s	**9.30**%	$> 10^5$s	1576s
R-Feat	4.53%	46.7s	1.75s	5.45%	7.02s	0.06s	-	-	-	-	-	-
SGD-QN [Bordes et al., 2009]	9.57%	0.30s	-	41.77%	0.20s	-	12.00%	1.50s	-	-	-	-
Linear	8.52%	1.50s	0.02s	29.93%	0.74s	0.01s	8.21%	64.0s	0.38s	23.25%	2.16s	0.04s
MC-Linear	7.77%	0.97s	0.02s	21.48%	112s	0.01s	7.01%	8.19s	0.38s	22.63%	250s	0.04s
LCC [Yu et al., 2009]	-	-	-	-	-	-	1.90%	-	-	-	-	-
Imp. LCC [Yu and Zhang, 2010]	4.38%	-	-	4.12%	-	-	2.28%	-	-	-	-	-
LLSVM	6.78%	5.21s	0.11s	17.25%	1.99s	0.02s	3.81%	120s	1.65s	17.36%	9.34s	0.32s
G-OCC	5.03%	48.1s	1.00s	7.90%	3.99s	0.08s	1.65%	1365s	16.9s	18.58%	178s	0.89s
C-OCC	**4.19**%	93.6s	1.03s	7.93%	8.19s	0.09s	1.72%	2641s	17.4s	20.14%	349s	0.94s
L2C	7.87%	26.5s	0.38s	8.65%	3.54s	0.15s	2.46%	774s	9.45s	22.63%	253s	0.73s
AMM	6.28%	4.38s	0.03s	11.15%	3.71s	0.01s	3.63%	94.7s	0.49s	19.2%	101s	0.13s
ML3+I	4.98%	32.0s	0.19s	3.30%	13.0s	0.10s	**1.59**%	822s	2.76s	18.48%	304s	0.53s
ML3	5.43%	29.5s	0.20s	**3.15**%	13.2s	0.11s	**1.59**%	745s	2.78s	14.84%	225s	0.44s

As an additional experiment on this benchmark, we also evaluated the performance of L2C coding, when varying p between 1 and 2. In Figure 4.6 we report the performance of each configuration on all the UCI datasets. In the same Figure we plot the average ranking of each configuration as well. As it can be seen, the best performance is obtained in most of the cases with $1 < p < 2$.

4.7.2 Character recognition.

The USPS [Hull, 1994] dataset consists of 7,291 training and 2,007 testing 16×16 gray-scale images of US postcodes, where each label corresponds to a digit between 0 and 9. LETTER [Frank and Asuncion, 2010] is a dataset composed of 16,000 training and 4,000 testing images of the 26 capital letters in the English alphabet; each image being compactly represented by a 16-dimensional vector. MNIST [LeCun et al., 1998] is a dataset comprising 70,000 28×28 gray-scale images of hand-written digits, from 0 to 9. This dataset has one official training split with 60,000 samples and an associated test set with 10,000 samples. As a preprocessing step for the last two databases we center the data. Finally, COVTYPE [Joachims, 2006] is class 1 in the Covertype dataset 2 of Blackard, Jock & Dean, and it consists of 522,911 fifty-four-dimensional training samples and 58,101 test samples. Although it is not a character recognition dataset, we included it here as an example of a large (in terms of number of samples) problem.

Following Zhang et al. [2011] we set $m = 90$ for MNIST, $m = 80$ for USPS and $m = 16$ for LETTER. For COVTYPE we set $m = 54$, which is the maximum allowed by OCC. The number of CCCP iterations (or SGD epochs, for LLSVM and OCC) is set to 30 and, as before, for L2C, ML3 and OCC we concatenate a 1 to each instance vector. The experimental results obtained with this settings are summarized in Table 4.4. Please note that in order to make these results easily comparable with those reported in other papers, the performance measure used for these experiments is the error rate, and not the accuracy, as in the other Chapters of this thesis.

We divide the Table in four parts, corresponding to: Gaussian methods, Linear SVM approaches, manifold learning approaches, and multi-component approaches that modify the sample-to-component assignments during the discriminative training. For this benchmark, we also report the results achieved by Yu et al. [2009] and Yu and Zhang [2010] using other manifold learning techniques (*LCC* and *Imp. LCC*) and we report the performance obtained by Bordes et al. [2009] using a stochastic quasi-Newton method (*SGD-QN*) for linear SVM.

As before, the L2C coding seems to perform similarly to the simple encoding scheme used by LLSVM (using the inverse Euclidean distances w.r.t. the 8 nearest neighbors), with more complicated coding approaches (as G-OCC C-OCC) outperforming them on the majority of the datasets. For LETTER, MNIST and COVTYPE, the ML3 algorithm obtains the state of the art for the class of multi-component SVMs algorithms, with performances close to the ones achieved by the Gaussian SVM [6]. Moreover, on COVTYPE the training and testing times of ML3 result to be several order of magnitude lower than the ones obtained by Gaussian-SVM. For USPS, the results are on par with the majority of the multi-component algorithms and with the Gaussian-SVM, with Improved LCC and C-OCC obtaining better results. It is worth noting that although L2C and ML3 utilize the same coding technique (i.e. the same method to produce the mixing coefficients, given a dictionary), there is a large performance gap between them. This confirms the importance of jointly training the components and the mixing coefficients, rather than fixing the coefficients in advance. Finally, we also note that on this benchmark using L2C to initialize the models of ML3 does not seem to significantly increase the performance.

The timings reported in Table 4.4 are measured by averaging the measurements of five different runs, on a single thread of an Intel(R) Core(TM) i7-2600K, with 16GB of RAM. Amongst the locally linear SVMs algorithms, the one with the lowest training and testing times results to be once again LLSVM. This is likely due to the fact that the encoding used by this algorithm makes use of only the eight closest anchor points, forcing all the other coefficients to be zero and thus saving many computations. On the other hand, even though ML3 computes class-specific weights for each sample, while solving the original multiclass problem, its training times are often on par or lower then those of G-OCC. The only major exception happens on the LETTER dataset, which is the one with the highest number of classes. Still, on this dataset the training times of ML3 are much lower than those obtained by MC-Linear, with an error-rate that is also almost one order of magnitude lower. The training times of ML3 result to be also comparable

6. The performance of R-Feat on MNIST and COVTYPE could not be measured, as with the considered settings the expanded features did not fit into the memory (16 GB) of the benchmarking machine.

or lower than the ones measured using C-OCC. This could be due to the fact that in OCC the manifold is trained using SVD, whose complexity is $O\left(\min\{n\,d^2, d\,n^2\}\right)$. Finally, we also note that in all the experiments the testing times of ML3 result to be from one to four orders of magnitude lower than those of Gaussian-SVM.

In Figure 4.7 we plot the testing error and the objective function (for the ML3 algorithm) as a function of the number of CCCP iterations, fixing m to 100. Note that, since LETTER and COVTYPE datasets consist of 16 and 54-dimensional instances, the maximum number of orthogonal coordinates are, respectively, 16 and 54. The OCC plots for these datasets are thus obtained with $m = 16$ and $m = 54$. As it can be seen, while the testing error of the ML3 algorithm reduces relatively fast on all the character recognition datasets, the algorithm exhibits a slow convergence on the COVTYPE dataset.

In Figure 4.8 we analyze the behavior of the ML3 algorithm and of the other locally linear SVM algorithms when varying the number of components m between 10 and 100, while keeping the other parameters as in Table 4.4. As outlined before, the maximum number of components for the OCC algorithms remains limited to 16 for LETTER and 54 for COVTYPE. As it can be noted, with m in the range [10, 100] all the algorithms tend to behave in a similar manner: the performance improves when the number of components is increased, while some signs of overfitting may appear when the number of components is chosen to be too high. We also notice that the performance tend to vary smoothly as m is changed, without any significant oscillation.

In Figure 4.9 we analyze the performance of ML3 when varying the parameter p in the set $\{1, 2, 3, 5, 10, 1000\}$, together with the best performing p in the set {1.1,1.2,1.3,1.4,1.5, 1.6,1.7,1.8,1.9}. For each dataset the number of components m is kept fixed to the value used to produce Table 4.4. While, as previously discussed, the ML3 algorithm seem to be only mildly sensitive to the choice of the number of components m, the choice of p seem to have a more crucial role. For example, the best performance is never achieved using $p = 1$. In other words, using only the single most confident component - as in the standard latent SVM implementation and in AMM - is easily outperformed by using a linear combination of them. As observed for L2C on the UCI datasets, there is a well marked minima between $p = 1$ and $p = 2$ on three datasets out of four. On these three datasets we also note a small performance difference between the best performing p (e.g. $p = 1.2$ for USPS) and the results obtained using the default value for p (i.e. the results in Table 4.4, obtained with $p = 1.5$). While this supports our default value for p, the experiments on COVTYPE also show that this might not always be a good choice and, to get even better performance, a cross-validation procedure might be desirable.

Finally, in Figure 4.10 we plot the testing error rate w.r.t. the amount of training time necessary to achieve a given performance. Each point is obtained with a different number of components m or, for R-Feat, with a different number of random features. The performance of AMM was not assessed in this benchmark, as the number of components for this algorithm is automatically determined. As before, for each point we perform a separate a five-fold cross-validation

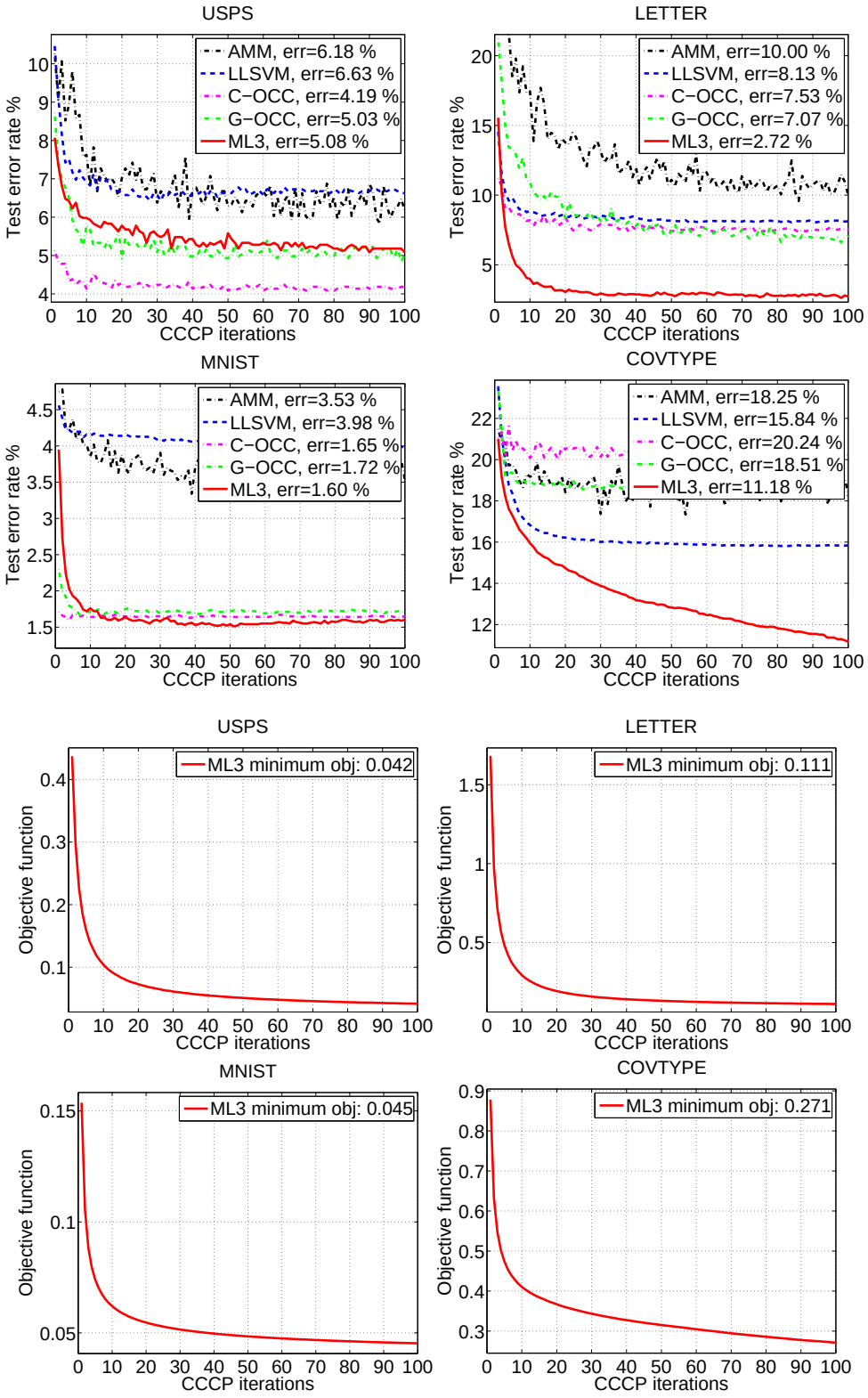

Figure 4.7 – Results varying the number of iterations on the USPS, LETTER, MNIST and COVTYPE datasets. Top: error rates on using $m = 100$. The curves related to OCC for LETTER and COVTYPE are obtained with $m = 16$ and $m = 54$, due to the limitations of the encoding. Bottom: value of the objective function of ML3 with $m = 100$.

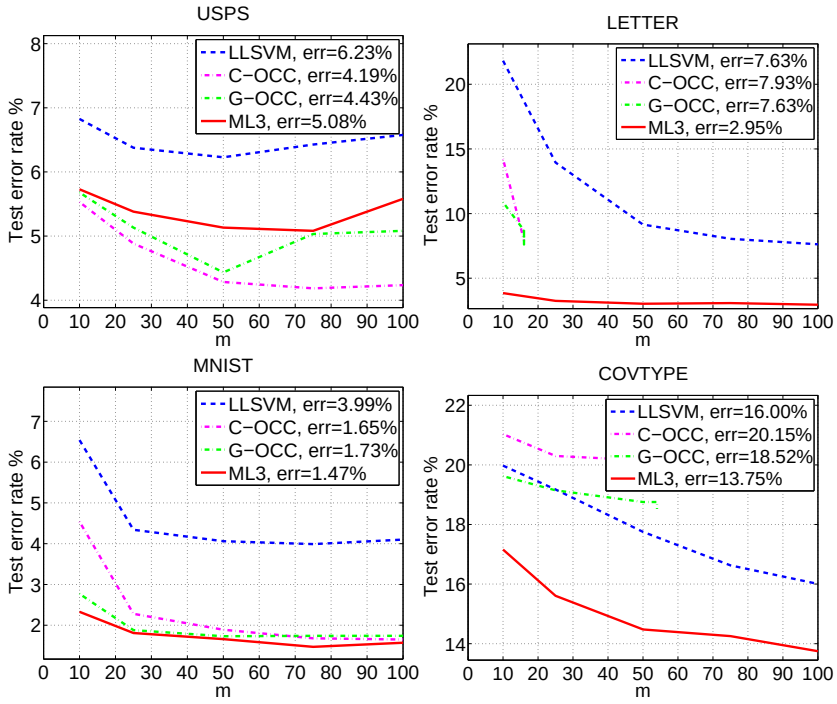

Figure 4.8 – Error rates varying the number of components m on the USPS, LETTER, MNIST and COVTYPE datasets.

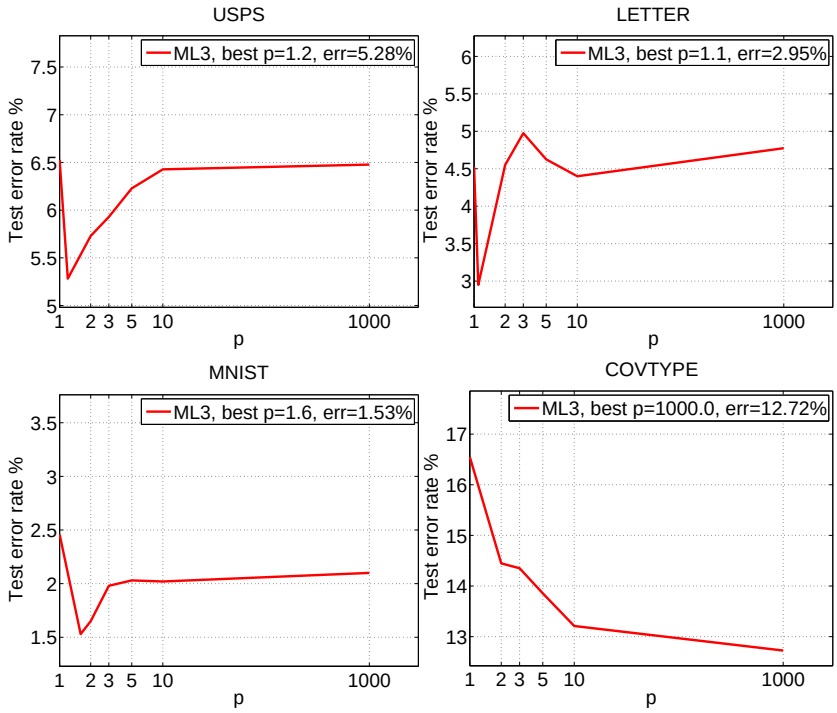

Figure 4.9 – Error rates varying the parameter p of the ML3 algorithm on the USPS, LETTER, MNIST and COVTYPE datasets.

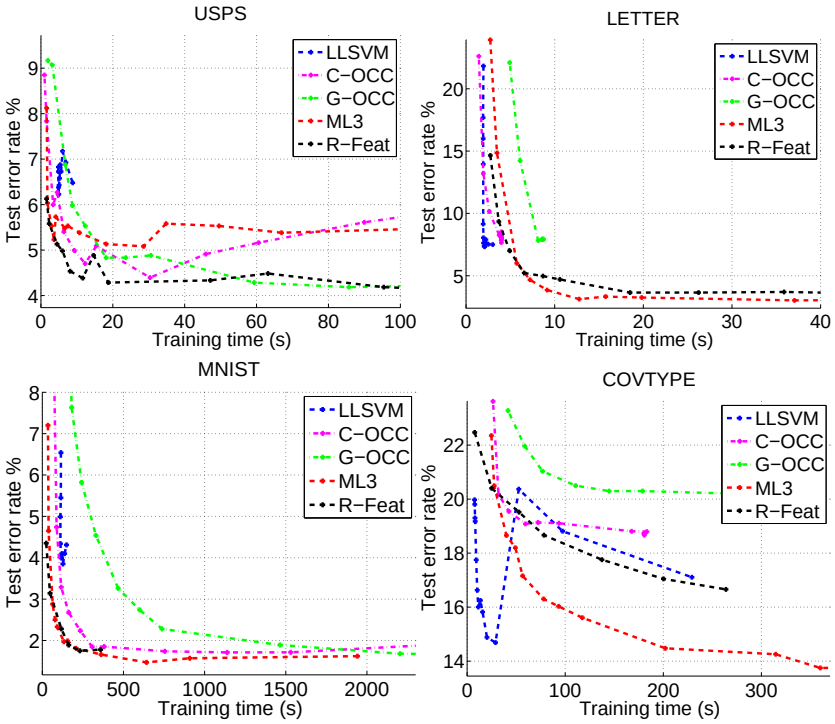

Figure 4.10 – Error rate versus training time on the USPS, LETTER, MNIST and COVTYPE datasets.

procedure and we measure the training time using five different runs. On two datasets out of four, ML3 has a better trade-off between performance and time w.r.t. the other algorithms.

On USPS dataset, all the methods seem to show similar behaviors in a small range of performance and time, with R-Feat obtaining the best trade-off. On the COVTYPE dataset, LLSVM seems to be the best performer on the left part of the plot (when a very limited amount of time is available). However, as for the other datasets, it also exhibits an early performance saturation, so that with larger amount of training time available, ML3 achieves the best trade-off. On the MNIST dataset, ML3 and R-Feat result to be the most effective algorithms. Please note that the performance of R-Feat is limited by the amount of memory available to the machine (due to the explicit feature expansion), while the ML3 algorithm can increase the complexity of the model without affecting the dimensionality of the features (and thus the memory footprint). ML3 is thus able to achieve a better absolute performance on this dataset. Finally, on LETTER dataset, LLSVM and C-OCC seem to achieve the most competitive performance on the left part of the plot (whenever a very little amount of time is available). Unfortunately, they also suffer from an early saturation of the performance (for C-OCC the maximum amount of components is $m = 16$ on this dataset), so that when as few as 6 seconds are available for training, ML3 guarantees the best trade-off.

4.7.3 Scene Recognition

As a final benchmark for ML3 , we perform evaluation on the three scene recognition datasets considered throughout this thesis, namely: the MIT-Indoor-67 [Quattoni and Torralba, 2009], the 15-Scenes [Lazebnik et al., 2006] and the UIUC-Sports [Li and Fei-Fei, 2007] datasets. For a complete description of these datasets and their benchmarking procedures please refer to Section 2.1.

As discussed in Section 1.3 and Section 2.1, these are difficult real-world classification problems, with high degrees of intra-class variability and inter-class similarity. This is particularly true for the MIT-Indoor-67 and the UIUC-Sports datasets. Indeed, as pointed out also in Chapter 3, in indoor scenes the location of meaningful regions and objects within each category changes drastically from image to image. Moreover, the close-up distance between the camera and the subject increases the severity of the view-point changes. Similarly, sport events may take place in different environments and involve a wide variety of subjects and poses. The MIT-Indoor-67 and the UIUC-Sports datasets are thus perfect test-beds for multi-component classification algorithms. For a complete description of these datasets and their benchmarking procedures please refer to Section 2.1.

For this last benchmark we make use of the features and kernels discussed in Chapter 3. Similarly to Chapter 3, all the SVM experiments (except for the multiclass linear SVM) are run using LIBSVM in a one-vs-all configuration, and using pre-computed kernel matrices. This enables us to quickly perform experiments and to compare with the exponential χ^2 kernel [Fowlkes et al., 2004] used in Chapter 3 (which is not otherwise supported by LIBSVM). On the other hand, it also prevents a fair comparison of the running times of the algorithms. In order to partially provide such a comparison, for the linear and the Gaussian kernel we re-run some of the experiments using, respectively, LIBLINEAR and LIBSVM, without pre-computing the kernel matrices. Similarly to before, the scaling parameter for the Gaussian and the exponential χ^2-kernel is tuned using the inverse average pairwise distances of the training samples. We run each experiment five times on five random training / testing splits and we report the average performance.

Please note that in order to make these results easily comparable with those reported in the other Chapters, the performance measure employed for these experiments is not the error rate (as for the other benchmarks in this Chapter). On the contrary, similarly to Chapter 3 and 5 we use the multiclass accuracy, computed as the mean of the diagonal of the confusion matrix.

We perform two sets of experiments, the first one comparing the performance of several multi-component algorithms on the MIT-Indoor-67 dataset, and the second one comparing the performance of ML3 to the ones obtained by SVM using linear, Gaussian and exponential χ^2 kernels on all the scene recognition datasets.

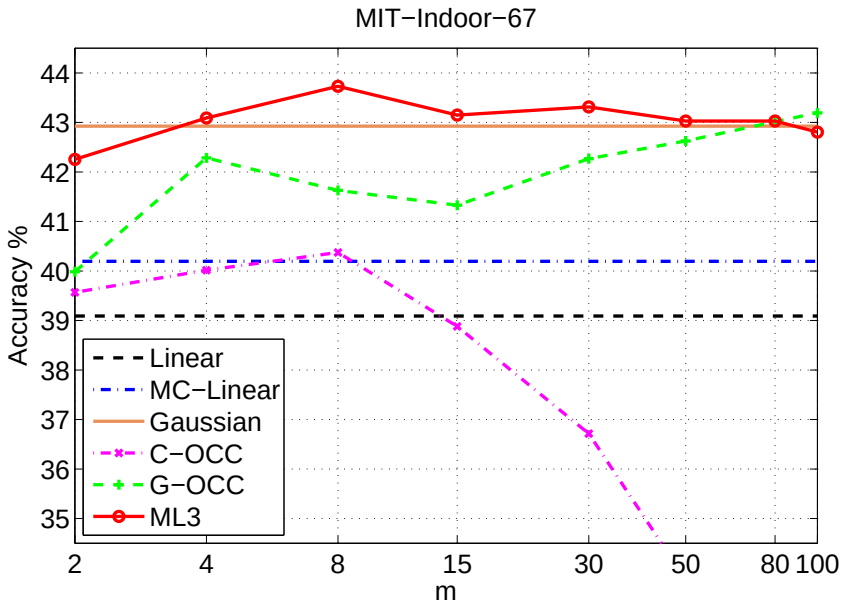

Figure 4.11 – Average test error rate on the ISR dataset, varying the number of components m.

Multi-component algorithms for Indoor Scene recognition

In the first experiment we focus on the most difficult scene recognition dataset, the MIT-Indoor-67 dataset, reporting the performance of the most competitive multi-component algorithms (namely ML3, C-OCC and G-OCC), together with the performance of the Gaussian kernel and of the linear SVM algorithms (multi-class and one-vs-all). We make use of multiresolution Horizontal features (see Section 3.3.2) and, as anticipated at the beginning of the Chapter, we replace average pooling with max-pooling (since latter has been shown to be a more suitable for linear classifiers [Yang et al., 2009]). This results in a compact, but highly discriminative 4096-dimensional descriptor, specifically designed to work with linear classifiers.

In Figure 4.11 we plot the test error rate of the multi-component algorithms w.r.t. the number of components used, comparing their performance to that of linear SVM (multiclass and one versus all) and of Gaussian SVM. As it is possible to see, on this dataset both ML3 and G-OCC are able to achieve and outperform the Gaussian SVM, with the former generally outperforming the latter. We note that already with two components ML3 performs largely better than a linear SVM, and that with as few as four components it is already able to match the performances of the Gaussian SVM. The best performance seems to be obtained with $m = 8$ components, while above this value the algorithm does not seem able to further improve the results. Finally, we also note that on this last dataset, the performances of C-OCC seem quite unsatisfactory. A reason for this could be found in the limited amount of samples (80) available to separately train each class-specific manifold, on the high-dimensional data ($d = 4096$).

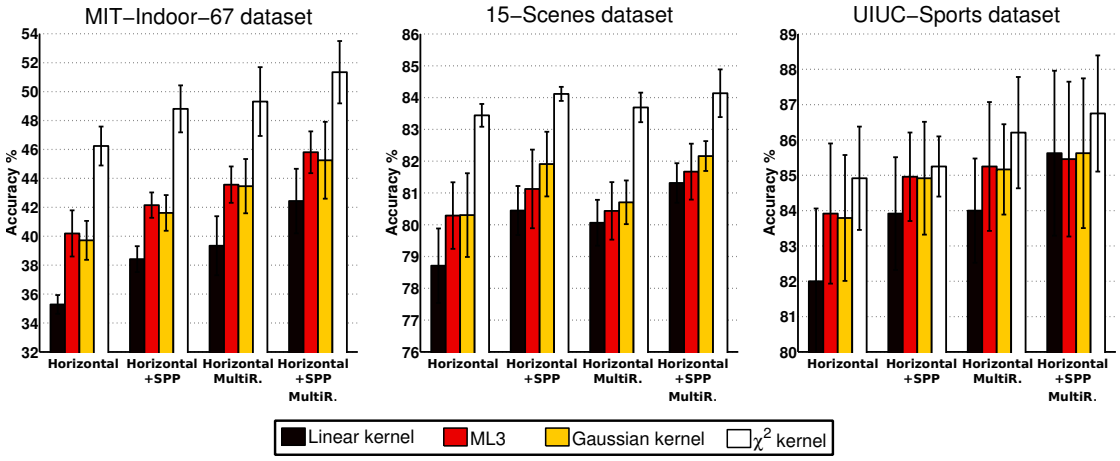

Figure 4.12 – Average accuracy on MIT-Indoor-67 (left), 15-Scenes (center) and UIUC-Sports (right), with $m = 8$ components.

Using ML3 with SPP representation

A second set of experiments is performed on all the scene recognition datasets, using the Horizontal and Horizontal + SPP image representation proposed in the previous Chapter (see Section 3.3), both in single and multiresolution (*MultiR.*). For these experiments we compare the performance of ML3 with the results obtained using SVM with linear, Gaussian and exponential χ^2 kernels. As before, for all the experiments except for the ones using exponential χ^2 kernel we replace average pooling with max-pooling, as the latter has been shown to perform better with linear algorithms [Yang et al., 2009].

In Figure 4.12 we report the performance of the four considered algorithms on all the three considered datasets. As before, each experiment is repeated five times and the regularization parameter is tuned using a 5-fold cross-validation procedure. Following the results reported in Figure 4.11, the number of components for the ML3 algorithm is set to $m = 8$ for all the datasets. In Table 4.5 we also report the training and testing times, together with the size of the learned model for the ML3 algorithm, the Linear SVM (as measured using LIBLINEAR) and the Gaussian kernel SVM trained and tested (without pre-computing the kernel matrix) on the multiresolution Horizontal + SPP image signatures. We do not report the same quantity for the exponential χ^2 kernel, as it is not supported by LIBSVM.

As it can be seen, on all the datasets and features (except for multiresolution Horizontal + SPP on the UIUC-Sports dataset) the best performing classification algorithm results to be SVM with exponential χ^2 kernel, the worst performing one results to be Linear SVM, with ML3 and Gaussian SVM in the middle. While there is a significant gap between the performances obtained by ML3 algorithm and the exponential χ^2 kernel SVM, the ML3 algorithm achieves scene recognition performance very close to the one obtained by the Gaussian SVM. Moreover, as it can be seen in Table 4.5, also for the scene recognition datasets, the training an testing times required by ML3 are orders of magnitude lower than the ones necessary to train and

Table 4.5 – Training times, testing times and size of the model for Linear SVM, ML3 and Gaussian kernel SVM, as measured using multiresolution Horizontal + SPP image signatures.

	MIT-Indoor-67			15-Scenes			UIUC-Sports		
Method	training	testing	size	training	testing	size	training	testing	size
Linear	221s	0.65s	4.19MB	7.60s	0.52s	0.94MB	2.58s	0.09s	0.5MB
ML3	977s	3.54s	33.5MB	37.8s	1.66s	7.5MB	8.12s	0.13s	4MB
Gaussian	9,714s	4,639s	4,774MB	229s	841s	339MB	40.6s	57.9s	166MB

test Gaussian SVM. Finally, the ML3 model is composed of $c * m$ hyperplanes, with m fixed in advance (in this case $m = 8$). On the contrary, the model of the Gaussian SVM includes a variable-length set of support-vectors, which for the considered datasets is typically in the order of hundreds per class. This results in a model, that for the MIT-Indoor-67 dataset occupies 4.77GB of memory, against the 33.5MB required by the ML3 model.

For completeness of results, in Table 4.6 we compare the performance obtained by the ML3 algorithm with the performance obtained by other multi-component approaches that have been applied to scene recognition problems (for a discussion about these methods please refer to Section 2.4.3). For each method we report the accuracy obtained on the three scene recognition datasets, the number of components used by each algorithm and the number of times the multi-component model has to be evaluated to classify each image.

We divide the table in two parts. On the top part we report the performance of the best-performing NBNN-based approaches [Boiman et al., 2008]. As discussed in Section 2.4.2, these algorithms can be viewed as multi-component methods making use of a very large set of components (normally one component for each local feature in each training image) that need to be evaluated on each patch of a query image. On the bottom we report the performance of multi-component approaches that are instead applied to the full image, or to large image patches (e.g. as obtained by using a spatial pyramid division).

We sort the approaches w.r.t. the number of evaluations of the multi-component model required to classify one image and w.r.t. the number of components in each model. For the NBNN methods the number of evaluations required is only an estimate, as obtained by considering the density of the sampling of the patches and the average size of the images in the datasets. For the method of Pandey and Lazebnik [2011], which adopts a specific sliding window approach with multiple parts involved, it was not possible to determine this number.

As it is possible to see, our approach provides competitive performances on all the datasets. Compared to other multi-component methods making use of a number of components with the same order of magnitude as for our algorithm, we obtain state of the art performances. W.r.t. NBNN methods, the strongest competitor seems to be the algorithm of Çakir et al. [2011], which outperforms our approach on the MIT-Indoor-67 dataset, and obtains a similar performance on 15-Scenes. Nonetheless, it is worth noting that: a) as reported by the authors, the accuracies for Çakir et al. [2011] reported in Table 4.6 require a manual tuning of their

Method	# eval.	m	MIT-Indoor-67	15-Scenes	UIUC-Sports
Vitaladevuni et al. [2013]	$> 10^3$	$> 10^5$	48.84	79.0	84.67
Çakir et al. [2011]	$> 10^3$	800	47.01	82.08	-
Pandey and Lazebnik [2011]	-	2	30.4	-	-
Parizi et al. [2012]	16	16	37.93	78.6	-
ML3, MultiR. Horiz. + SPP-Itti's	1	8	**45.81**	**81.67**	**85.46**

Table 4.6 – Performance comparison with previous studies applying multi-component approaches to scene recognition problems. For each approach we also report the number of components m and the number of times the multi-component model has to be evaluated to produce the final image classification.

model parameters; b) without this manual tuning, their best performance degrades to 45.22% for the MIT-Indoor-67 dataset and 81.04% for the 15-Scenes datasets (accuracies that are similar to the ones obtained by ML3). Moreover, all the NBNN algorithms make use of a number of components that is orders of magnitude higher than in our case, while requiring the multi-component model to be evaluated on each image patch. Instead, ML3 needs to be evaluated just once on the final image signature.

4.8 Discussion

In this Chapter we aimed at addressing the high intra-class variability and the high inter-class similarity of scene categories, by proposing a new classification algorithm. In order to cope with the computational efficiency requirements of many scene recognition applications, without resorting to simple linear classifiers, we focused on multi-component algorithms. This class of algorithms allows to automatically discover and represent sub-categorical structures (e.g. sub-categories corresponding to different view-points, or different compositions of the images), without the need of providing sub-categorical annotations. Multi-component algorithms are thus able to learn non-linear decision boundaries, by automatically assigning each sample to one or more components, representing a different sub-category. Following this approach we proposed a new multi class algorithm (ML3) based on a Latent and Locally Linear SVM formulation. Differently from previous works, ML3 has the advantage of neither requiring a two-stages formulation (i.e. manifold and classifier learning), nor a complex objective function, using additional gating functions. Moreover, differently from current Latent SVM implementations, ML3 makes use of a hyper-parameter p allowing to modulate how the different components contribute to the prediction of each sample. This in turn allows to finely tune the level of smoothness of the decision function. Using this formulation, we showed the sample-to-component assignments to be computable using an exact closed form solution. Moreover, by plugging the closed form solution back into the scoring function, we obtained a prediction rule where the sample-to-component coefficients do not need to be explicitly computed. We analyzed this scoring function from an encoding point of view and showed it to be meaningful for the approximation of smooth locally linear functions. In this context the parameter p was also shown to be useful to tune the trade-off between localization and

reconstruction error minimization. Finally, we also discussed the implicit feature-map used by ML3 during the prediction phase, and empirically showed how in this space the intra-class similarity of the samples is increased, while the inter-class similarity is reduced.

From an experimental point of view, the ML3 algorithm was first analyzed and validated by an in-depth evaluation on typical machine learning benchmarks, such as datasets from the UCI collection [Frank and Asuncion, 2010] and character recognition datasets [Hull, 1994; Frank and Asuncion, 2010; LeCun et al., 1998]. On these datasets the algorithm was shown to be only relatively sensitive to the choice of the number of components m, while the parameter p proved to play a more important role. In all the considered benchmarks, the best performance was never achieved using $p = 1$ (assigning each sample only to one component), while the majority of the experiments supported our default value $p = 1.5$. We also found a few problems in which the performance of the algorithm was improved by setting p to larger values. For optimal performance it is thus advisable to cross-validate this hyper-parameter.
Compared to state of the art multi-component classification algorithms, ML3 was shown to provide a very competitive trade-off between prediction performance and training time, being also able to achieve performances close to the ones provided by Gaussian SVM.

In the last part of this Chapter we evaluated ML3 on the three scene recognition datasets used throughout this thesis. We made use of the SPP representations introduced in Chapter 3, both at single and at multiple resolutions, and we compared ML3 against SVM with linear, Gaussian and exponential χ^2 kernel (as used in Chapter 3). On the majority of the considered scene recognition datasets and features, ML3 was shown to provide recognition performances on par with the ones achieved by Gaussian SVM. Moreover, the training and testing times, and the memory footprint of ML3 were shown to be orders of magnitude lower than the ones required by Gaussian SVM. On the other hand, neither the Gaussian SVM, nor the ML3 algorithm were able to provide the same level of performance of the χ^2 SVM. The ML3 algorithm could thus be used only to partially fill the gap between linear SVM and the best performing kernel SVM. This could be due also to the fact that for BoW representations, χ^2 is a more suitable distance than the Euclidean one. With respect to multi-component approaches specifically designed for image classification tasks, ML3 seemed also to be outperformed by (computationally expensive) NBNN approaches [Boiman et al., 2008], operating on a patch level. Still, when compared to previously proposed multi-component algorithms applied to the full images (and specifically designed for image classification problems), our approach provided significantly higher accuracies.

In the next Chapter we discuss how to make use of the ML3 algorithm to reduce the very high testing complexity of the patch-based multi-component NBNN classifier, while also increasing its predictive performance. Future research directions that could be worth investigating are the design of compact image signatures optimized for the ML3 classifier, and the design of incremental procedures to efficiently minimize the objective function of ML3. It could also be interesting to evaluate the effectiveness of ML3 for selecting image-specific saliency thresholds, maximizing the confidence of the classifier on each image.

5 Patch-based classification of Visual Scenes

Two crucial features of a good scene recognition algorithm are its ability to generalize and its robustness w.r.t. high degrees of intra-class variability. Such features are amongst the distinctive characteristics of the Naive Bayes Nearest Neighbor (NBNN) algorithm [Boiman et al., 2008], an image classification framework that since its introduction in 2008 has been gaining momentum in the visual recognition community. In this Chapter we show how with a straightforward modification of the original NBNN scoring function it is possible to use the Multiclass Latent Locally Linear SVM (ML3) algorithm introduced in Chapter 4 to discriminatively learn a set of prototypical local features for each class. The resulting classification algorithm, that we call Naive Bayes Non-linear Learning (NBNL) preserves the generality and robustness properties of the original approach, while greatly reducing its memory requirements during testing, and significantly improving its performance. To the best of our knowledge this is the first work to exploit the structure of the local features through the use of a multi-component discriminative learning method. Experiments over the three scene recognition datasets considered throughout this thesis [Quattoni and Torralba, 2009; Lazebnik et al., 2006; Li and Fei-Fei, 2007] show the effectiveness of the proposed algorithm, which outperforms several existing NBNN-based methods and is competitive with standard Bag-of-Words plus SVM approaches.

5.1 Introduction

The dominating trend over the last decade in visual recognition has been the use of Bag of Words representations (BoW) [Csurka et al., 2004], combined with state of the art machine learning classifiers, ranging from max-margin algorithms [Lazebnik et al., 2006] to Bayesian frameworks [Fei-Fei and Perona, 2005]. This general approach is crucially based on the assumptions that: 1) it is possible to determine the class of an image by computing image-to-image distances; 2) the representations based on vector quantization, or other forms of encoding are sufficient to describe the images. Since the seminal work of Boiman et al. [2008], these two assumptions have been challenged with the introduction of the Naive Bayes Nearest Neighbor (NBNN) algorithm [Boiman et al., 2008]. The NBNN classifier drops the vector quantization and the image-to-image distance computation in favor of an image-to-class approach. Hence,

classes are directly represented by unordered sets of local features extracted from the training images, and a query image is classified by directly comparing its local features with those contained in each class-specific set of local features. This results in a classification method that is competitive, performance-wise, with more established learning methods using simple BoW representations, while at the same time promising a high degree of robustness and generality when applied to categorization problems. This last feature of NBNN, and of NBNN-based methods is very appealing for scene recognition problems. The community of researchers working on NBNN-based algorithms has acknowledged the potential of these methods for this specific application, by using more and more often several of the existing public databases for scene classification, as benchmarks to evaluate their approaches [Behmo et al., 2010; Wang et al., 2011a; Çakir et al., 2011; Vitaladevuni et al., 2013].
The original NBNN algorithm does not perform any learning during training, as it simply stores all the available local features for all classes. While this makes the method attractively simple, it also leads to potential memory problems and scalability issues during testing. In this Chapter we propose a method for tackling these issues, while also improving the recognition performance of the algorithm. We build on the ML3 algorithm introduced in Chapter 4, and we show how with a moderate modification of the decision function used by NBNN it is possible to use this algorithm to learn an extremely compact set of prototypical local features for each class. This new representation results in a greatly reduced memory footprint and computation time during testing, while also significantly increasing the predictive performance w.r.t. the original NBNN algorithm. We call the resulting algorithm *Naive Bayes Non-linear Learning* (*NBNL*).
To assess our method, we perform experiments on the three scene recognition datasets used throughout this thesis (MIT-Indoor-67 [Quattoni and Torralba, 2009], 15-Scenes [Lazebnik et al., 2006] and UIUC-Sports [Li and Fei-Fei, 2007]), comparing it with previously proposed NBNN-based algorithms and with a BoW+SVM approach augmented by a form of Spatially Local Coding [McCann and Lowe, 2012a]. Experiments show that NBNL significantly outperforms NBNN on all the datasets, while also achieving competitive or better performance than the BoW+SVM baseline and previously proposed NBNN-based algorithms. Finally, by applying the Horizontal + SPP approach proposed in Chapter 3 to the NBNL algorithm proposed in this Chapter, the scene recognition performance can be further increased.

The rest of the Chapter is organized as follows: in Section 5.2 we relate the present work to the most relevant previous works discussed in Section 2.4.2, instantiating also this work in the scene recognition pipeline introduced in Chapter 2. In Section 5.3 we introduce our approach, by directly deriving it from the NBNN algorithm. In Section 5.4 we report the experimental results, while in Section 5.5 we provide a final discussion.

5.2 Related works

As in the previous Chapters we present the works related to our approach by instantiating the scene recognition pipeline introduced in Chapter 2 for the specific approach discussed here. In Figure 5.1 we report a visualization of the scene recognition pipeline used in this Chapter. As

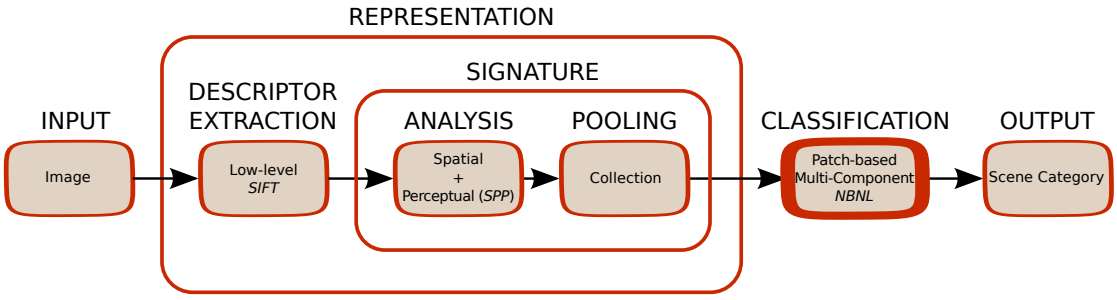

Figure 5.1 – The scene recognition pipeline proposed in this Chapter. The component of the pipeline related to the main contribution of this Chapter is highlighted by a thick border.

it is possible to see, the proposed pipeline is related to the ones used in the previous Chapters: we use the same visual primitives (namely, small densely sampled SIFT patches) and a similar spatial and perceptual analysis of the image. Moreover, similarly to Chapter 4, we make use of a multi-component classification algorithm. The idea of applying a spatial partitioning scheme to the NBNN algorithm was first discussed by Wang et al. [2011a], who proposed to restrict the Nearest Neighbor search performed by NBNN only to features extracted from the same area of each considered local feature. Similarly to Wang et al. [2011a], we perform experiments by using only the prototypes learned from features extracted from the same area of each considered local feature. However, differently from Wang et al. [2011a], we follow Chapter 3 and consider both spatial and saliency driven image partitions.

The main differences w.r.t. the approaches discussed in the previous Chapters lie in the representation of the extracted features, in the type of image signature used, and in the specific classifier used. In particular, similarly to most of the NBNN methods [Boiman et al., 2008] discussed in Section 2.4.2, we make use of low-level descriptors of the patches, without any additional mid-level encoding. Moreover, as for all the NBNN algorithms, each image is represented by a collection of SIFT features, while the multi-component model is applied to each single feature. For a full review of the literature of NBNN-based methods please refer to Section 2.4.2.

As outlined in 2.4.2, the main limitation of the NBNN algorithm lies in the high computational complexity and high memory requirements of the testing procedure. NBNN methods, indeed, require to repeatedly perform a Nearest Neighbor search for each local feature in the query image, w.r.t. a training set that may consist of hundreds of thousands of local features for each class. Some of the methods discussed in Section 2.4.2 can be used to alleviate this problem [Çakir et al., 2011; McCann and Lowe, 2012b; Vitaladevuni et al., 2013; Escalante et al., 2014]. For example, Escalante et al. [2014] propose to use Prototype Generation (*PG*) techniques to obtain a set of representative prototypes for each class. Some of the considered PG algorithms are shown to produce a reduced set of prototypes, without hurting (or, in few cases, even slightly improving) performance. Still, on a simple 4 classes object recognition problem, the best performing methods require several days of training and produce a number of prototypes in the order of thousands per class. By using a class-specific k-means procedure,

Çakir et al. [2011] are able to obtain a more compact set of prototypes. Differently from these works, our algorithm learns the prototypes in a supervised and discriminative fashion. Moreover, similarly to Timofte et al. [2012], for each local feature our prediction function uses more than just the single Nearest Neighbor in each class (both during training and testing). The combination of these two modifications allows us to obtain an extremely compact and discriminative set of prototypes for each class, significantly outperforming the NBNN algorithm on all the considered datasets.

This Chapter is based on the work published in Fornoni and Caputo [2014].

5.3 The NBNL approach

While many of the NBNN methods presented in Sections 2.4.2 and 5.2 result in a performance increase w.r.t. the original NBNN algorithm, often also with reduced time and space complexities, only the method of [Çakir et al., 2011] produces a relatively compact representation of the training data. In this Chapter we show how with a straightforward modification of the NBNN scoring function it is possible to make use of ML3 algorithm discussed in Chapter 4 to directly learn an extremely compact and discriminative set of prototypical local feature for each class. By effectively exploiting the structure of the training patches, the proposed method greatly reduces the memory necessary to represent the training set, while also significantly increasing the classification accuracy and the testing speed.

5.3.1 The NBNN algorithm

In the NBNN algorithm [Boiman et al., 2008], the class of an image is estimated by a Maximum A-Posteriori (MAP) approach. Let $\boldsymbol{X}_i = \begin{bmatrix} \boldsymbol{x}_{1i} & \boldsymbol{x}_{2i} & \dots & \boldsymbol{x}_{ri} \end{bmatrix} \in \mathbb{R}^{d \times r}$ be a query image containing a set of r local features $\boldsymbol{x}_{ji} \in \mathbb{R}^d$ and $\mathcal{Y} \triangleq \{1, \dots, c\}$ be a set of classes. If we assume that the class priors are uniform and that the local features are conditionally independent given the class (Naive-Bayes assumption), the MAP estimate of the class of image $\boldsymbol{X}_i$ can be written as

$$\hat{y}_i = \underset{y \in \mathcal{Y}}{\arg\min} \; - \sum_{j=1}^{r} \log p(\boldsymbol{x}_{ji} | y). \tag{5.1}$$

$p(\boldsymbol{x}_{ji}|y)$ can be estimated using a kernel density estimator

$$\hat{p}(\boldsymbol{x}_{ji}|y) = \frac{1}{n_y h^d} \sum_{l=1}^{n_y} k\left(\frac{\boldsymbol{x}_{ji} - \boldsymbol{x}_l}{h}\right), \tag{5.2}$$

where k is a kernel function [Epanechnikov, 1969], $\boldsymbol{x}_l$ is the l-th local feature from class y, n_y is the total number of training local features extracted from images of class y and h is the bandwidth parameter.

The RHS of equation (5.2) may be difficult to compute, as the number of local features n_y in

each class may be very large. Nonetheless, as shown by Boiman et al. [2008], it can be reliably approximated by computing the value of the kernel only for the single Nearest Neighbor $NN_y(\boldsymbol{x}_{ji})$ of $\boldsymbol{x}_{ji}$ in class y. Using this approximation and choosing the kernel function to be the Gaussian: $k(\boldsymbol{x}) = (2\pi)^{-\frac{d}{2}} \exp\left(-\|\boldsymbol{x}\|^2\right)$, the NBNN decision rule becomes

$$\hat{y}_i = \underset{y\in\mathscr{Y}}{\arg\min} - \sum_{j=1}^{r} \log \hat{p}(\boldsymbol{x}_{ji}|y) \tag{5.3a}$$

$$= \underset{y\in\mathscr{Y}}{\arg\min} - \sum_{j=1}^{r} \log\left(\frac{1}{n_y h^d} \sum_{l=1}^{n_y} k\left(\frac{\boldsymbol{x}_{ji} - \boldsymbol{x}_l}{h}\right)\right) \tag{5.3b}$$

$$\approx \underset{y\in\mathscr{Y}}{\arg\min} - \sum_{j=1}^{r} \log\left(k\left(\frac{\boldsymbol{x}_{ji} - NN_y(\boldsymbol{x}_{ji})}{h}\right)\right) \tag{5.3c}$$

$$= \underset{y\in\mathscr{Y}}{\arg\min} \sum_{j=1}^{r} \|\boldsymbol{x}_{ji} - NN_y(\boldsymbol{x}_{ji})\|^2. \tag{5.3d}$$

The resulting classification algorithm is very simple and can provide image recognition performances close to those obtained using BoW models [Boiman et al., 2008].
As anticipated before, one main disadvantage of this algorithm is that it requires to store all the local features of the training set, while its expected prediction complexity grows either linearly, or logarithmically (if the exact NN search is replaced with an approximated one [Arya and Fu, 2003]) w.r.t. the number of local features in the training set. In the next Section we discuss a modification of the NBNN decision rule, allowing to use the ML3 algorithm (discussed in Chapter 4) to learn a compact set of prototypical local features for each class.

5.3.2 The NBNL decision rule

Let n_y be the number of local features in the training set of class y. In order to reduce the memory requirements and the search space of the NBNN classifier it would be desirable to preselect a set of $m \ll n_y$ representative prototypes for each class y. Though a difficult task at first glance, the goal is achievable by making use of ML3 algorithm introduced in Chapter 4. Let us call $\boldsymbol{W}_y$ the matrix $\begin{bmatrix} \boldsymbol{w}_{1y} & \boldsymbol{w}_{2y} & \dots & \boldsymbol{w}_{my} \end{bmatrix}^\top \in \mathbb{R}^{m\times d}$ containing the set of prototypical local features from class y. Let us also assume that all the local features and prototypes are normalized to one (e.g. SIFT features are normalized by design). For a given testing image $\boldsymbol{X}_i$, the NBNN prediction rule can be decomposed as

$$\hat{y}_i = \underset{y\in\mathscr{Y}}{\arg\max}\, s(\boldsymbol{X}_i, y) \tag{5.4a}$$

$$s(\boldsymbol{X}_i, y) = \sum_{j=1}^{r} f(\boldsymbol{x}_{ji}, y), \tag{5.4b}$$

$$f(\boldsymbol{x}_{ji}, y) = -\|\boldsymbol{x}_{ji} - NN_{\boldsymbol{W}_y}(\boldsymbol{x}_{ji})\|^2, \tag{5.4c}$$

where, once again, $NN_{\boldsymbol{W}_y}(\boldsymbol{x}_{ji})$ indicates the nearest neighbor of $\boldsymbol{x}_{ji}$ in $\boldsymbol{W}_y$. Using this notation we can rewrite $f(\boldsymbol{x}_{ji}, y)$ as

$$\begin{aligned} f(\boldsymbol{x}_{ji}, y) &= -\min_{k=\{1,\dots,m\}} \|\boldsymbol{x}_{ji} - \boldsymbol{w}_{ky}\|^2 \\ &= -\min_{k=\{1,\dots,m\}} \|\boldsymbol{x}_{ji}\|^2 + \|\boldsymbol{w}_{ky}\|^2 - 2\boldsymbol{w}_{ky}^\top \boldsymbol{x}_{ji} \\ &= \max_{k=\{1,\dots,m\}} 2\left(\boldsymbol{w}_{ky}^\top \boldsymbol{x}_{ji} - 1\right), \end{aligned} \tag{5.5}$$

where for the last equality we have used the assumption that all the features and prototypes are normalized to 1 (with this assumption it is also possible to see that $0 \leq \|\boldsymbol{x}_{ji} - \boldsymbol{w}_{ky}\|^2 \leq 4$, $\forall i, j, k, y$). Using the normalization assumption and removing the constants, the NBNN prediction rule can thus be equivalently written as

$$\hat{y}_i = \arg\max_{y \in \mathcal{Y}} \sum_{j=1}^{r} \max_{k=\{1,\dots,m\}} \boldsymbol{w}_{ky}^\top \boldsymbol{x}_{ji}. \tag{5.6}$$

As suggested also by Timofte et al. [2012], and especially since we assume to be using a matrix $\boldsymbol{W}_y$ with a highly reduced set of prototypical features, it would be advisable to use more than just the single closest prototype, when computing the score for a given local feature. Taking inspiration from the scoring functions introduced in Chapter 4, we thus propose to search for a linear combination of all the prototypes in $\boldsymbol{W}_y$, maximizing the alignment of the combination with the considered local feature $\boldsymbol{x}_{ji}$. This idea can be formalized by the following objective function

$$f(\boldsymbol{x}_{ji}, y) = \max_{\boldsymbol{\beta} \geq 0, \|\boldsymbol{\beta}\|_p \leq 1} \boldsymbol{\beta}^\top \boldsymbol{W}_y \boldsymbol{x}_{ji}, \tag{5.7}$$

where $\boldsymbol{\beta}$ is an m-dimensional vector of coefficients that weights how the different prototypes in the matrix $\boldsymbol{W}_y$ are combined to compute $f(\boldsymbol{x}_{ji}, y)$. As discussed in Section 4.4.2, the first constraint in (5.7) avoids that the vector $\boldsymbol{\beta}$ inverts the similarities between $\boldsymbol{x}_{ji}$ and the prototypes $\boldsymbol{w}_{ky}$, while without the second constraint the maximization problem would be unbounded. As discussed in Section 4.6, the parameter p allows to control the locality of the combination (with $p = 1$ producing the most local combination) and the smoothness of the classifier (with $p = 2$ producing the smoothest decision boundaries). As it can be seen, for each sample, (5.7) finds a local linear combination of the class-prototypes maximizing the alignment of the combination with the sample.

An important property of (5.7) is that it has an analytical solution for $\boldsymbol{\beta}$ (see Section 4.4.1), making it possible to efficiently compute $f(\boldsymbol{x}_{ji}, y)$ for any given $\boldsymbol{x}_{ji}$ and class y. For example, it is easy to show that when $p = 1$, an optimal solution of (5.7) is of the form $\boldsymbol{\beta} = \begin{bmatrix} 0 & \dots & 0 & 1 & 0 & \dots & 0 \end{bmatrix}$, where the only 1 is in the k-th position providing the maximum positive value for $\boldsymbol{w}_{ky}^\top \boldsymbol{x}_{ji}$ (for a proof please refer to Section 4.4.1). Except for a constant factor, this is equivalent to solve (5.5) with the additional requirement that, instead of searching amongst all the m prototypes of class y, we restrict the search to the closest ones (such that

$\|\boldsymbol{x}_{ji} - \boldsymbol{w}_{ky}\|^2 \leq 2$). If we instead allow p to vary in $(1,2]$, multiple $\boldsymbol{w}_{ky}$ could take part in the linear combination. For example, as shown by Lemma 4 in Section 4.4.1, with $p = 2$ the weight assigned to each prototype would be directly proportional to its alignment to $\boldsymbol{x}_{ji}$. Plugging (5.7) into (5.4) the NBNL decision rule is finally defined as

$$\hat{y}_i = \arg\max_{y \in \mathscr{Y}} s(\boldsymbol{X}_i, y) \tag{5.8a}$$

$$s(\boldsymbol{X}_i, y) = \sum_{j=1}^{r} f(\boldsymbol{x}_{ji}, y) \tag{5.8b}$$

$$f(\boldsymbol{x}_{ji}, y) = \max_{\boldsymbol{\beta} \geq 0, \|\boldsymbol{\beta}\|_p \leq 1} \boldsymbol{\beta}^\top \boldsymbol{W}_y \boldsymbol{x}_{ji}. \tag{5.8c}$$

As explained above, the non-negativity constraints of this formulation introduce some locality in the computation of the nearest neighbor of $\boldsymbol{x}_{ji}$. It is to note, however, that in real scene recognition applications using NBNN there will likely be at least one SIFT feature in the pool $\boldsymbol{W}_y$ that projects positively on $\boldsymbol{x}_{ji}$. In those cases and when $p = 1$, the formulations in equations (5.4) and (5.8) would thus produce identical results.

5.3.3 Learning the NBNL prototypes

As anticipated before, in order to be able to efficiently represent the training set and efficiently predict the class of a query image, we need to learn the matrix $\boldsymbol{W}_y$ for each class. This can be achieved by making use of the ML3 algorithm discussed in Chapter 4, which is a multi component locally linear classifier, based on a latent SVM formulation [Yu and Joachims, 2009]. As discussed in Chapter 4, the ML3 algorithm allows to learn smooth non-linear classifiers as local linear combinations of linear ones. For a query instance, the linear components of each class are locally linearly combined according to their confidence on the sample. A main characteristic of this approach is that it allows to efficiently train and test powerful non-linear classifiers, without the computational complexity and memory requirements of kernels, or the computational burden and architectural complexity of multi-layer architectures.

Given a block-matrix $\boldsymbol{W}$ defined as $\boldsymbol{W} \triangleq \begin{bmatrix} \boldsymbol{W}_1^\top & \boldsymbol{W}_2^\top & \dots & \boldsymbol{W}_c^\top \end{bmatrix}^\top \in \mathbb{R}^{mc \times d}$ and containing one block for each class $y \in \{1, \dots, c\}$, the prediction of the ML3 algorithm is defined (see equation 4.18) as:

$$\hat{y}_i(\boldsymbol{W}) \triangleq \arg\max_{y \in \mathscr{Y}} s_{\boldsymbol{W}}(\boldsymbol{x}_i, y) \tag{5.9}$$

$$s_{\boldsymbol{W}}(\boldsymbol{x}_i, y) \triangleq \max_{\boldsymbol{\beta} \geq 0, \|\boldsymbol{\beta}\|_p \leq 1} \boldsymbol{\beta}^\top \boldsymbol{W}_y \boldsymbol{x}_i. \tag{5.10}$$

Suppose we are given a set of training examples $(\boldsymbol{x}_i, y_i) \in \mathscr{X} \times \mathscr{Y}$, where $i = \{1, \dots, n\}$, $\mathscr{X} \subseteq \mathbb{R}^d$ is the input space and $\mathscr{Y} = \{1, \dots, c\}$ is the output (and the decision) space. The objective

function of ML3 is defined as

$$\min_{\boldsymbol{W},\boldsymbol{\xi}} \frac{\lambda}{2}\|\boldsymbol{W}\|_F^2 + \frac{1}{n}\sum_{i=1}^{n}\xi_i \tag{5.11a}$$

$$\text{s.t. } 1 + \max_{y\in\mathcal{Y}\setminus\{y_i\}} s_{\boldsymbol{W}}(\boldsymbol{x}_i, y) - s_{\boldsymbol{W}}(\boldsymbol{x}_i, y_i) \leq \xi_i, \qquad i = 1,\dots,n \tag{5.11b}$$

$$\xi_i \geq 0, \qquad i = 1,\dots,n \tag{5.11c}$$

where $\|\boldsymbol{W}\|_F$ is the Frobenius norm of $\boldsymbol{W}$. For more technical details about this objective function and the optimization procedure, please refer to Section 4.4.2. As discussed in Chapter 4, the resulting algorithm provides a powerful non-linear classifier that can be both efficiently trained and tested.

As it can be seen, equation (5.10) has exactly the same form of (5.7), meaning that we can directly make use of the scores provided by the ML3 algorithm into the NBNN-like decision function (5.8). More importantly, we can also make use of the ML3 algorithm to learn the matrices $\boldsymbol{W}_y$.

Given a set of local features extracted from all the training images of each class, the task assigned to the ML3 algorithm is thus to learn how to predict the class of any single local feature, by discriminatively training the matrices $\boldsymbol{W}_y$. Though this is a very hard task, it does not need to be solved exactly, as during the prediction phase the Naive Bayes classifier on top of the ML3 algorithm can correct the mistakes made by the latter.

Since we still make use of the Naive Bayes assumption while have replaced the Nearest-Neighbor distance with the score provided by a non-linear learning algorithm, we call our approach *Naive Bayes Non-linear Learning* (*NBNL*). The proposed algorithm adopts the promising image-to-class distance paradigm and combines it with a discriminative training phase to produce a compact representation of the training data. This results in a remarkable reduction of the memory requirements during prediction and a significative improvement in the classification accuracy, as it is demonstrated in the next Section.

5.4 Experiments

In this Section we report the results obtained by NBNL, comparing it against BoW with χ^2 or intersection kernel and 512 visual words, NBNN [Boiman et al., 2008], NBNN applied to PCA-SIFT [Vitaladevuni et al., 2013], as well as the results reported in the NBNN literature. We also perform a comparison against a simple One-vs-All linear SVM trained on the local features. We do not attempt to use a kernel-SVM to classify the patches, as it would not scale to the millions of samples that we are dealing with.

We perform experiments on the three scene recognition datasets considered throughout this thesis, namely: the MIT-Indoor-67 [Quattoni and Torralba, 2009], the 15-Scenes [Lazebnik et al., 2006] and the UIUC-Sports [Li and Fei-Fei, 2007] datasets. For a complete description

of these datasets and their benchmarking procedures please refer to Section 2.1. For all our experiments (and for all the algorithms implemented in our benchmark) we use a common feature extraction procedure. We initially rescale all the images so that their smallest dimension is equal to 200px (keeping the original aspect ratio), in order to enforce scale consistency. We favor SIFT features over other ones (e.g. NIMBLE Kanan and Cottrell [2010], as suggested by Timofte et al. [2012]) to fairly compare with the wide majority of NBNN and BoW methods. We thus use VLFeat [Vedaldi and Fulkerson, 2008] to densely extract SIFT features every 8px, using four different patch sizes: 16, 24, 36 and 54 pixels. As in [Boiman et al., 2008; Rematas et al., 2012; Timofte et al., 2012; Tuytelaars et al., 2011; Vitaladevuni et al., 2013], to each local feature we concatenate the coordinates of its relative position in the image. We finally normalize each local feature to 1. Using these features in a standard BoW model results in an approach close to the recently introduced Spatially Local Coding [McCann and Lowe, 2012a], in which the patches in the dictionary include also an expected location. We thus name our Bag-of-Words baseline *SLC-BoW* to underline its difference w.r.t. the vanilla BoW model, lacking any spatial information.

We perform three sets of experiments. In the first one we make use of a reduced feature set and analyze the performance of the NBNL algorithm while varying the number of prototypes, the parameter p and the number of local features in the training set. We then perform a second set of experiments by using the full feature set on a fixed configuration of the NBNL algorithm. Finally, we perform experiments applying the perceptual and spatial pooling approaches introduced in Chapter 3 to the NBNL algorithm.

5.4.1 Single-scale experiments

In our first set of experiments we analyze the performance of our method when using single-resolution SIFT features with a patch size of 16px. With this configuration the total number of training local features is around 500,000 for the UIUC-Sports dataset and approximatively the double for the 15-Scenes dataset. In this scenario the amount of information provided by the features is relatively limited (compared to the full multi-scale setup) and a good classifier is fundamental to achieve reasonable performance. In Figure 5.2 (left) we use the 15-Scenes and the UIUC-Sports datasets to compare our approach with a simple Naive Bayes Linear Learning (NBLL) algorithm, in which we replace ML3 with a One-Vs-All linear SVM. Each experiment is repeated five times, on five different random splits, while the regularization parameter is tuned using 5-fold cross-validation. We vary the number of prototypes (components) learned by ML3 between $m = 5$ and $m = 200$, and we plot the average accuracy together with the standard deviation. As it can be noted, the non-linearity introduced by the ML3 classifier results in a remarkable improvement w.r.t. the linear classifier. On the UIUC-Sports dataset the absolute improvement is about +39%, and it is even higher on the 15-Scenes dataset. Intuitively, learning only a single prototype per class is not sufficient to accurately represent the complexity of the local features and the discrimination between different classes is not possible. On the other hand, by learning a set of m prototypes per class and predicting with a sample-

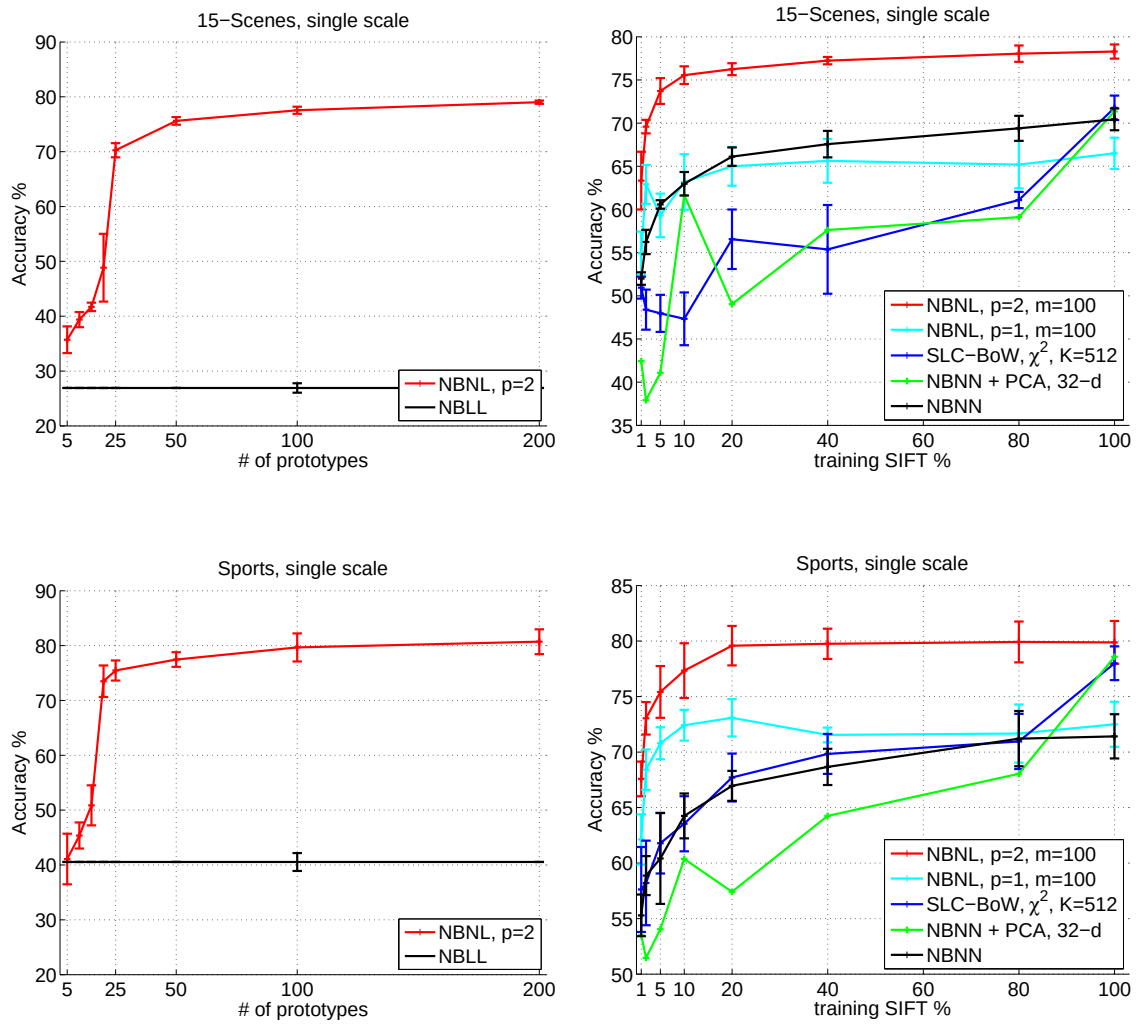

Figure 5.2 – Left: performance of the NBNL algorithm varying the number of prototypes, w.r.t. the NBLL baseline (using a One-Vs-All linear SVM). Right: performances of our method and several other baselines with an asymmetric sampling strategy (sub-sampling the patches only from the training images). All the results are obtained using single-scale SIFT features. Results on the top are obtained on the 15-Scenes dataset. Results on the bottom are obtained on the UIUC-Sports dataset.

specific linear combination of them, each NBNL class model can represent a wide range of local features, resulting in greatly improved performances. With as few as 25 prototypes our method already achieves competitive results on both datasets, while 100 prototypes are sufficient to obtain a performance level very close to the maximal one.

In Figure 5.2 (right) we evaluate the robustness of our approach against several other methods, when applying an asymmetric sampling strategy, as advocated by Timofte et al. [2012]. We randomly sub-sample the training local features by keeping only a given percentage of features per image and we run experiments with each setting. We plot the results of our method with $p \in \{1, 2\}$, together with the results of SLC-BoW, NBNN and NBNN + PCA. Following the observations presented in Figure 5.2 (left), we fix the number of NBNL prototypes to $m = 100$. As before, each experiment is repeated five times and the regularization parameter is tuned using cross-validation. The average accuracy is then plotted, together with the standard deviation. As it is possible to see, amongst the considered methods the proposed NBNL approach results to be the most robust w.r.t. sub-sampling the training patches. By using as little as 2% of the training samples NBNL can already reach the performance level of the NBNN algorithm using the full training data, while superior performance can be achieved using only 10% of the training data. Moreover, with just 20% of the training data, our method is already able to reach a level of performance close to the maximal one. When all the training features are preserved, the performance of NBNL with $p = 2$ is similar or better to that of the SLC-BoW and the NBNN + PCA baselines, significantly outperforming both NBNN and NBNL with $p = 1$. We also note that, while with $p = 1$ the NBNL algorithm performs similarly to the original NBNN, setting $p = 2$ (and thus allowing for multiple prototypes to take part in the prediction) significantly improves the results. Finally, using $p = 1.5$ we observed an advantage over $p = 2$ only whenever the number of prototypes per class is very low. We thus opt for $p = 2$, as it slightly speeds up the training procedure.

For visualization purposes, in Figure 5.3 we also plot the top-scoring patches selected by the original NBNN algorithm and the proposed NBNL approach on example images of the UIUC-Sports and the 15-Scenes datasets. As it can be noted, NBNL favors patches lying in the most discriminative areas, correcting some of the mistakes made by the vanilla NBNN algorithm. For example, water is a more discriminative cue than paddles, as they are easily confused with field delimitation rods and mallets used in croquet games. We also note that on the UIUC-Sports dataset our algorithm has learned a spatial bias towards the patches lying on the top of the scene (rich of contextual data).

5.4.2 Multi-scale experiments

For our second set of experiments we benchmark our algorithm on all the three scene recognition datasets, against all the considered baselines, using all the training features with the full multi-scale setup. Using this configuration the total number of training SIFT features amounts to about 1,950,000 for the UIUC-Sports dataset, 3,690,000 for 15-Scenes and more

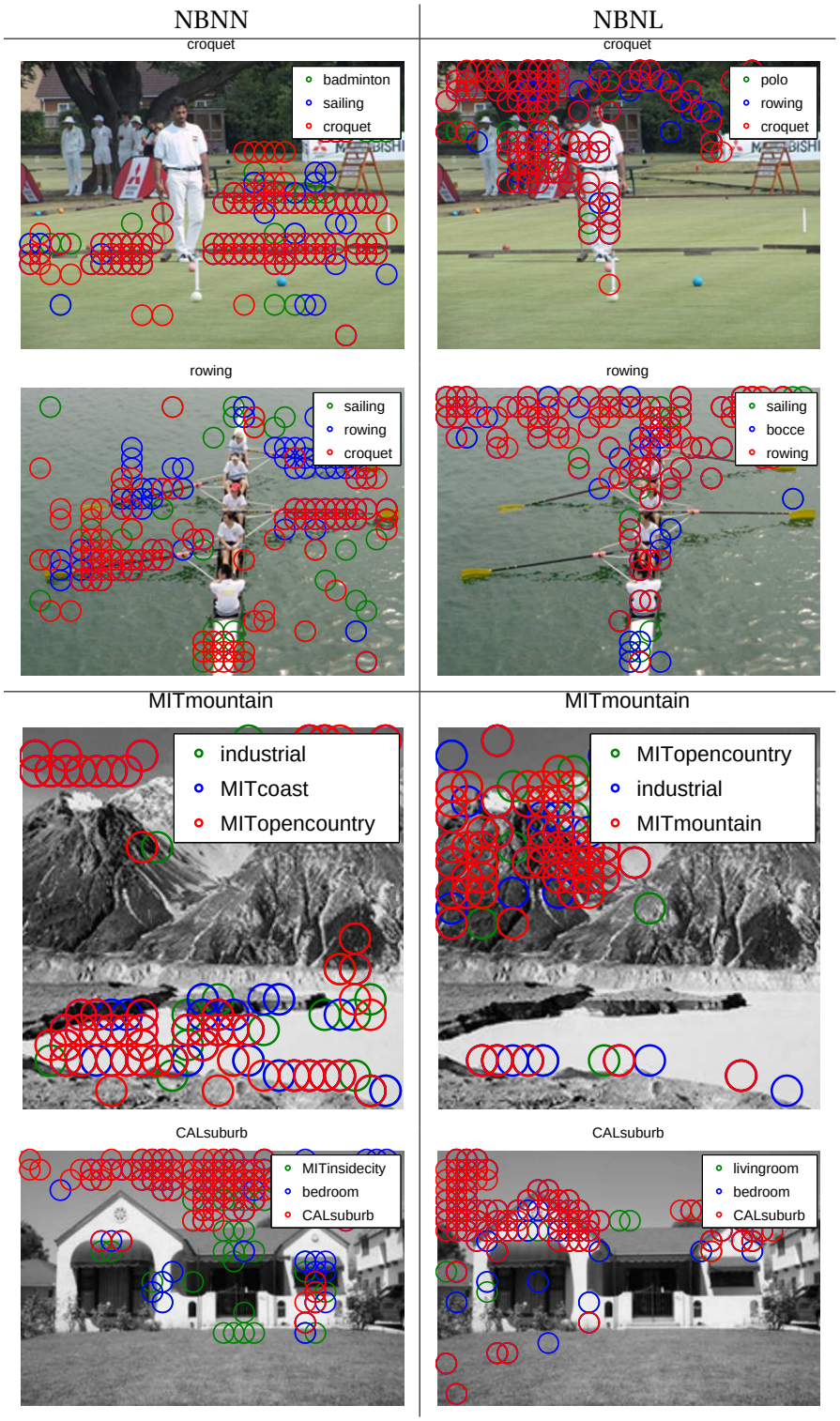

Figure 5.3 – Visualization of the classification results on images of the UIUC-Sports (top) and the 15-Scenes (bottom) datasets. Each image is titled with its ground-truth label, while in green, blue and red are visualized the top-scoring SIFT patches for the three top-scoring classes (from lowest to highest) of each image.

Table 5.1 – Results of NBNL using multi-scale SIFT features, compared to NBNN baselines on the same feature set, NBNN results reported in the literature (with citation) and two SLC-BoW algorithms on the same feature set (top).

Method	MIT-Indoor-67	15-Scenes	UIUC-Sports
SLC-BoW (Intersection, K=512)	37.54±2.24	79.99±0.55	84.58±2.61
SLC-BoW (χ^2, K=512)	40.92±0.89	81.31±0.39	86.54±2.27
NBNN [Wang et al., 2011a]	-	72.8±0.7	67.60±1.1
NBNN + NIMBLE [Timofte et al., 2012]	-	74.2±1.0	-
NBNN [Tuytelaars et al., 2011]	-	75±3	-
NBNN [Vitaladevuni et al., 2013]	-	-	81.48
NB-INN + NIMBLE [Timofte et al., 2012]	-	78.2±1.0	-
NNbMF [Çakir et al., 2011]	42.46	78.99	-
NBNN + PCA (32-d) [Vitaladevuni et al., 2013]	**48.84**	79.0	84.67
NBNN-kernel [Tuytelaars et al., 2011]	-	79±2	-
Pooled NBNN + NIMBLE [Rematas et al., 2012]	-	79.7±1.5	-
NB-INN + G-KDES + PCA [Timofte et al., 2012]	-	79.8±1	-
NBNN + LI2C [Wang et al., 2011a]	-	80.07±0.4	82.07±1.2
NBNN	38.67±1.58	77.25±0.74	80.08±1.94
NBNN + PCA (32-d)	45.76±2.33	80.53±0.56	85.50±1.73
NBNL	42.15±1.60	**82.30**±0.99	**85.54**±2.81

than 16,000,000 for MIT-Indoor-67. Training a multi-class classifier on such a large number of samples can be a challenge. For the MIT-Indoor-67 dataset we thus opt to train the NBNL algorithm in single-precision and using a One-vs-One approach, which decomposes the problem into a number of very small binary problems, allowing for massive parallelization. For the other two datasets the original multi-class training procedure is used. Following the results presented in section 5.4.1 all the NBNL experiments are performed using $p = 2$ and $m = 100$. Each experiment is repeated five times, while the regularization parameter is tuned using 5-fold cross-validation. In Table 5.1 we report the average accuracy and the standard deviation of the algorithms implemented in our benchmark, together with the results reported in the NBNN-related literature. For this benchmark we focus on results that do not make use of spatial pyramid, or other types of spatial coding that could be combined with the methods (NBNN, BoW and NBNL) used in our benchmark. As it has been shown in Chapter 3 and repeatedly reported by other authors [Wang et al., 2010b; Çakir et al., 2011; Wang et al., 2011a], any enhanced spatial analysis can further improve the performance of scene classification algorithms. An empirical confirmation of this fact for the NBNL algorithm (obtained by applying the SPP pooling approach described in Chapter 3) is reported in the next Section.

As it can be seen, even when using a rich multi-scale representation our approach outperforms all the other NBNN-based algorithms on two out of three datasets, while being also competitive with the SLC-BoW baselines. Despite its simplicity, the NBNN + PCA approach seems to be a very good performer on the MIT-Indoor-67 dataset, though the difference is less marked using our feature set. We note also that in Vitaladevuni et al. [2013] the performance of the original NBNN is not reported for the MIT-Indoor-67 dataset, making it difficult to properly

evaluate the impact of the raw features on their final results. It is important to underline that our approach also produces an extremely compact representation of the original training set. For example, for the UIUC-Sports dataset (with the multi-scale setup) we have measured a memory footprint of less than 830 kilobytes for our model, while the original training data requires around 1.9 gigabytes to be stored in double precision, or about 475 megabytes in a PCA compressed format. This amounts to a three orders of magnitude compression w.r.t. the original feature set and more than two orders of magnitude w.r.t. the PCA-compressed representation. The reason for this compression lies in the fact that our representation contains $8 \times 100 = 800$ prototypes in total (100 prototypes per class), instead of the almost two millions in the original set. This is achieved at the cost of a training procedure that for this dataset takes 3 hours on average (on a single thread of an Intel(R) Core(TM) i7-2600K with 16GB of RAM). Despite this relatively expensive training procedure, another advantage of our approach w.r.t. the original NBNN algorithm lies in a highly reduced testing time. For example, with the multi-scale setup the average time necessary to evaluate our algorithm on all the testing images of the Sports dataset is of 51 seconds, while the NBNN algorithm implemented using a fast approximated nearest neighbor approach [Muja and Lowe, 2009] requires more than 20 minutes on average. This corresponds to a reduction of more than one order of magnitude in the testing time as well. Similar results are obtained on the 15-Scenes dataset as well, where the testing time is reduced from more than 3 hours required by NBNN to less than 7 minutes for NBNL.

5.4.3 Experiments with Horizontal and Saliency-driven Perceptual Pooling

This last experimental Section is dedicated to the application of the Horizontal + SPP pooling strategy introduced in Chapter 3, to the NBNL method proposed in this Chapter. Specifically, following Chapter 3, we make use of a combination of the Horizontal and SPP-Itti's pooling approaches (for technical details about these methods please refer to Section 3.3).

In order to apply these pooling techniques to the NBNL algorithm, and similarly to Wang et al. [2011a], we use ML3 to obtain a specialized set of prototypical features for each considered region: upper, lower, salient and non-salient. This is simply achieved by training ML3 only with the local features extracted from each image region. For a given query image, the feature-to-class distances are then computed by comparing each local feature only with the prototypes specifically trained for the regions it belongs to (salient vs. non-salient and upper vs. lower). We repeat the process using both the SPP and the Horizontal partitioning scheme, and we integrate the two approaches by simply averaging the image-to-class distances obtained in this way.

In Figure 5.4 we report the results obtained using this technique on the UIUC-Sports dataset and on the 15-Scenes dataset, using both single scale and multi-scale setups. As it is possible to see, similarly to what discussed in Chapter 3 (and to what reported by Wang et al. [2011a]), by forcing the NBNL algorithm to evaluate the local features taking into account also their

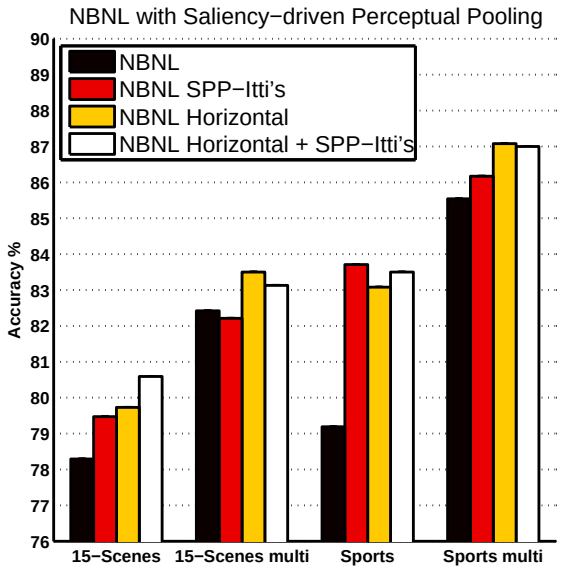

Figure 5.4 – Scene recognition performance obtained by applying the SPP approach described in Chapter 3 to the NBNL algorithm. Results marked with the keyword *multi* are obtained using a multi-scale setup. The other results are obtained using the single scale setup.

Table 5.2 – Comparison of the results of NBNL + Horizontal + SPP to other NBNN approaches using spatial information.

Method	15-Scenes	UIUC-Sports
NNbMF γ_m^* [Çakir et al., 2011]	82.08	-
NBNN + LI2C + Weight + SPP [Wang et al., 2011a]	**83.7**±0.49	84.3±1.52
NBNL multi-scale SPP-Itti's	82.21±0.55	86.17±2.49
NBNL multi-scale Horizontal + SPP-Itti's	83.13±0.62	**87.00**±2.10

quantized spatial location (upper vs. lower) and saliency (salient vs. non-salient) in the image leads to a further performance improvement. This is especially evident using the single-scale setup, where the performance gaps are larger and the best performance is always obtained by combining the spatial and the saliency driven pooling. On the other hand, and similarly to what found in Chapter 3, with the full multi-scale setup the performance gains of the SPP representation dwindle. As it can be noted on the 15-Scenes dataset, with this setup combining SPP with horizontal pooling does not seem to further improve the performance w.r.t. horizontal pooling itself.

In Table 5.2 we compare our multi-scale results to the ones reported by other authors using NBNN based methods and performing an additional spatial analysis. For these two datasets, these methods represent the most competitive approaches in the NBNN literature. As it is possible to see, our approach obtains comparable results on the 15-Scenes dataset, and better results on the UIUC-Sports dataset.

5.5 Discussion

In this Chapter we aimed at addressing the high intra-class variability of scene categories by using a suitable image representation and classification algorithm. To this end, we considered the NBNN algorithm, an image classification approach designed to address problems with high intra-class variability [Boiman et al., 2008]. Being a Nearest-Neighbor based algorithm, the main limitations of NBNN lie in the high computational complexity and memory requirements of the testing phase.

In this Chapter we proposed a method to improve both the performance and testing efficiency of the NBNN algorithm. Specifically, we showed that when the local features are normalized and the NN search is localized, the feature-to-class distances used by NBNN can be computed using the scoring function of the ML3 algorithm introduced in Chapter 4. We thus proposed to make use of ML3 to learn a set of prototypical local features for each class. In its most general formulation, the proposed algorithm computes the feature-to-class distances using sample and class specific linear combinations of the learned prototypes. The optimal combination coefficients are in turn computed using the closed form solution introduced in Chapter 4.

The proposed algorithm, named NBNL, was evaluated on the three scene recognition datasets used throughout this thesis. By effectively harnessing the training data in a discriminative framework, the NBNL algorithm is shown to provide two main advantages: 1) the memory footprint and computation time during prediction are reduced by more than one order of magnitude; 2) the recognition performance is significantly increased. In facts, on all the considered datasets the NBNL algorithm significantly outperforms the original NBNN algorithm, achieving performance o par with that of a BoW model using χ^2 kernel and a form of SLC spatial encoding [McCann and Lowe, 2012a]. In addition, by using the Horizontal + SPP pooling technique introduced in Chapter 3 to obtain multiple sets of prototypical patches specialized to different image regions (e.g. salient and non-salient), the performance of the NBNL algorithm is shown to be further increased.

The main limitation of the approach in its current form is that it presents a computationally intensive training procedure, due to the relatively slow convergence of the CCCP optimization used by the ML3 algorithm. Future works should focus on this issue. For example, Felzenszwalb et al. [2010] describe a data-mining algorithm for latent SVM, able to select hard examples and discard easy ones. It would be interesting to evaluate the effects of applying this method to the NBNL algorithm. Another interesting direction would be to explore ways to train ML3 in a fully stochastic incremental way. Finally, it might be also worth investigating the performance of other efficient multi-component methods, such as Adaptive Multi-hyperplane Machines [Wang et al., 2011b], or Locally Linear SVMs [Ladicky and Torr, 2011].

6 Conclusion

The goal of this thesis was the development of methods for automatically and efficiently annotating an image with the scene category that best describes it as a whole. The research problem was motivated by potential applications, such as automatic organization of digital collections of images and vision-based spatial reasoning for mobile robots. We followed the established image classification pipeline consisting of a representation step, followed by a classification step. In order to address the high intra-class variability and inter-class similarity characteristic of visual scene categories, while preserving classification efficiency required by the potential applications, we made the following design choices: 1) use of low-to-mid level image representations, not employing unreliable and computationally expensive object detectors; 2) multi-component categorical models, allowing to represent complex categories by means of simple and specialized sub-categorical components; 3) supervised discriminative learning algorithms, directly trained to minimize the number of classification errors. Following these design choices we made three main contributions, related to the representation and classification steps of the pipeline. The problems addressed and the contributions made in each part of this thesis are summarized in the next Section.

6.1 Achievements

Saliency-driven image representation

In Chapter 3 we focused on the representation step of the scene recognition pipeline. We tackled the problem of the high visual variability of scene categories, considering also the efficiency requirements of potential scene recognition applications. To this end, we proposed a compact image representation based on a saliency-driven pooling approach, named SPP. We made use of the established Itti's saliency [Itti et al., 1998] and proposed a saliency function directly operating on the low-level local features to be pooled. The considered saliency functions were used to segment the image into a salient and a non-salient region, and to separately pool the features from the two regions. The proposed pooling scheme was shown to generate pooling regions with different average levels of visual complexity (high for the salient regions and

low for the non-salient ones), thus capturing perceptually coherent structures independently of their positions in the scene. By combining SPP with a simple horizontal-bands pooling approach we obtained well-performing image signatures, up to one order of magnitude more compact than the ones obtained using standard spatial pyramid schemes. The proposed image representations were shown to be particularly effective on the most difficult scene recognition problems, while being comparable to spatial pyramid representations on the other problems.

Efficient multi-component scene recognition

In Chapter 4 we addressed the problem of designing an efficient classification algorithm capable of discriminating between classes with rich sub-categorical structures. To this end, we proposed to employ a set of sub-categorical linear components for each class and to use sample and class specific linear combinations of components to perform the prediction. We formulated the algorithm as a Locally Linear SVM problem [Ladicky and Torr, 2011] and casted it in a discriminative and latent SVM framework [Yu and Joachims, 2009]. Within this framework, the optimal sample-to-component assignments were shown to be computable using an efficient analytical solution, with a tunable level of sparsity. The resulting multi-class, multi-component algorithm was named Multiclass Latent Locally Linear SVM (ML3). We analyzed the scoring function used by ML3 from an encoding point of view, and discussed its behavior for the approximation of locally linear functions. We also discussed the implicit feature map used by this scoring function and empirically showed it to increase the intra-class similarity, while reducing the inter-class similarity. On typical machine learning benchmarks the algorithm was shown to provide a very competitive trade-off between prediction performance and training time, while also ensuring high efficiency of the prediction phase. On scene recognition problems, the proposed algorithm was evaluated on a modified version of the Horizontal + SPP representations introduced in Chapter 3. On these image representations, the ML3 algorithm was shown to provide scene recognition performance on par with that of a Gaussian kernel SVM [Cristianini and Shawe-Taylor, 2010], using only a fraction of the computational resources required by the latter.

Patch-based multi-component scene recognition

In Chapter 5 we focused on multi-component algorithms able to assign different parts of the image (e.g. image patches) to different sets of components. The classification algorithm proposed in this Chapter, named NBNL, was built upon the ML3 algorithm proposed in Chapter 4 and NBNN [Boiman et al., 2008], a patch-based Nearest-Neighbor classification algorithm designed to address problems with high levels of intra-class variability. We analyzed the link between the Nearest-Neighbor patch-to-class distance used by NBNN and the scoring function used by ML3. Exploiting this link, we replaced the Nearest-Neighbor based patch-to-class distance, with the patch-to-class distance obtained using the efficient scoring function of ML3. Differently from NBNN, which is limited to use only the single nearest patch in each class, the NBNL patch-to-class distance is computed w.r.t. a sample-specific linear combination of

prototypical patches. The patch prototypes were discriminatively learned by applying ML3 to the set of (SIFT) patches extracted from the training images. Differently from the approach proposed in Chapter 4, the NBNL algorithm proposed in Chapter 5 used different linear combinations of components (patch prototypes) for each of the patches extracted from a given query image. With respect to the original NBNN algorithm, the proposed algorithm was shown to provide two main advantages: 1) the memory footprint and computational complexity of the testing phase were reduced by several orders of magnitude; 2) the scene recognition performance was increased to the level provided by BoW + χ^2 kernel SVM approaches. Even higher performance levels were obtained by applying the Horizontal + SPP pooling technique introduced in Chapter 3 to the NBNL algorithm proposed in this Chapter.

6.2 Discussion, limitations and future work

The methods proposed in this thesis achieve competitive performance, with compact representations and relatively efficient classification procedures. Amongst the scene recognition methods proposed, the most effective resulted to be the one proposed in Chapter 3, making use of a χ^2 kernel SVM on the multiresolution Horizontal + SPP image signatures. For small datasets, similar levels of performance were achieved also by applying a combination of the SPP and the Horizontal pooling techniques to the NBNL algorithm proposed in Chapter 5. Thanks to the patch-level modeling, this algorithm obtained also the most compact models. The best trade-off between performance and efficiency was achieved by directly applying the ML3 algorithm to a modified version of the Horizontal + SPP image signatures, as proposed in Chapter 4.

In the following we discuss the most important limitations of the above discussed algorithms. For each of the considered shortcomings we also provide some potential research directions to tackle the problem and improve the considered approach.

Bottom-up saliency, with fixed segmentation threshold

The SPP technique presented in Chapter 3, and used throughout the thesis, is based on a segmentation of the image into the most and the least salient areas, using a bottom-up saliency operator and a mass threshold fixed to a constant for all the images. The threshold used in this thesis was justified by an extensive discussion and empirical evaluation. Still, it is important to remember that saliency is a relative property of each image and, consequently, an image-specific threshold should be used. Moreover, the discrimination between salient and non-salient areas should be done by taking into account also the scene recognition task to be addressed. These two goals could be jointly achieved by modeling the saliency threshold as an additional latent variable of the ML3 algorithm. In this way, the classification algorithm would directly be able to select an image and class specific segmentation threshold, maximizing the confidence of each class-model.

Batch optimization

The objective function of the ML3 algorithm is minimized using a Constrained Concave Convex Procedure (CCCP) [Yuille and Rangarajan, 2003; Smola et al., 2005; Sriperumbudur and Lanckriet, 2012]. Although each iteration of the algorithm is efficiently optimized using SGD, the full optimization procedure is carried on in a batch fashion, requiring multiple passes over the full training set. This prevents a direct application of the algorithm to very large scale problems, where more than a single pass over the data would be prohibitive. This limitation was already encountered in this thesis when we had to apply the ML3 algorithm to the 16+ millions patches extracted from the images of the 67 categories of the MIT-Indoor-67 dataset. A very important research direction would thus be to investigate how to optimize the objective function of ML3 in a fully stochastic fashion. A possible way to achieve this could be to make use of the recently introduced stochastic majorization-minimization algorithm [Mairal, 2013], which allows to stochastically minimize non-convex functions in the form of difference of convex functions, such as the one of ML3 . In order to accelerate convergence in the fully stochastic regime we could also consider replacing the non-differentiable multi-class hinge loss used by ML3 , with its smooth log-exponential counterpart [Amit et al., 2007].

Large and redundant sets of training patches

The NBNL algorithm presented in Chapter 5 directly operates on densely sampled SIFT features, extracted at multiple scales from each image. This results in a very large amount of training patches that needs to be processed. Moreover, given the local nature of the extracted patches, a large part of them may be redundant. The extremely high compression levels obtained by the NBNL algorithm are indeed strong indications in this direction. Beside improving the convergence speed of ML3 (as discussed above), another direction that could be considered is that of finding ways to explicitly mine hard samples. Felzenszwalb et al. [2010], for example, proposed a data-mining procedure for this purpose, adapted it to Latent SVMs and applied it to select difficult negative image patches for object detection tasks. We could apply a similar technique to select a set of difficult SIFT patches, during the training process of NBNL. Another alternative could be to avoid the usage of densely sampled SIFT features, in favor of features extracted from sparsely sampled patches. NIMBLE features [Kanan and Cottrell, 2010], for example, are extracted only from large salient areas of the image and in a fixed number (e.g. 100). Moreover, they have already been proved to be very effective for NBNN classification algorithms [Timofte et al., 2012].

A Mathematical proofs

Lemma (Solution for $\beta_{\boldsymbol{V}}(\boldsymbol{x})$ and $1 < p < \infty$) 1. *Let* $\boldsymbol{V} = \begin{bmatrix} \boldsymbol{v}_1 & \boldsymbol{v}_2 & \cdots & \boldsymbol{v}_m \end{bmatrix}^\top \in \mathbb{R}^{m \times d}$ *and* $\beta_{\boldsymbol{V}}(\boldsymbol{x})$ *be defined as in equation* (4.13)*. If* $1 < p < \infty$*, for every* $\boldsymbol{x} \in \mathbb{R}^d$ *s.t.* $\left\| (\boldsymbol{V}\boldsymbol{x})^+ \right\|_q > 0$*, with* $q = p/(p-1)$*, the* j*-th element of the optimal vector* $\beta_{\boldsymbol{V}}(\boldsymbol{x})$ *is given by*

$$\beta_{\boldsymbol{V}}(\boldsymbol{x})_j = \left(\frac{\left| \boldsymbol{v}_j^\top \boldsymbol{x} \right|_+}{\left\| (\boldsymbol{V}\boldsymbol{x})^+ \right\|_q} \right)^{q-1}.$$

Furthermore, $\beta_{\boldsymbol{V}}(\boldsymbol{x})^\top \boldsymbol{V}\boldsymbol{x} = \left\| (\boldsymbol{V}\boldsymbol{x})^+ \right\|_q$.

Proof. Decompose the objective function $\boldsymbol{\beta}^\top \boldsymbol{V}\boldsymbol{x}$ into two sums: one corresponding to the positive elements of the vector $\boldsymbol{V}\boldsymbol{x}$, and the other corresponding to the remaining elements. As $\boldsymbol{\beta}$ is constrained to be non-negative, it follows that all the $(\beta)_i$ associated with non-positive $\boldsymbol{v}_i^\top \boldsymbol{x}$ need to be zero. For the remaining β_i, the problem is equivalent to find an $\boldsymbol{\alpha}$ on the p-unit ball such that $\boldsymbol{\alpha}^\top \boldsymbol{d}$ is maximized for a vector $\boldsymbol{d}$, whose elements are the positive entries of the vector $\boldsymbol{V}\boldsymbol{x}$. The solution of this problem is the point on the surface of the p-unit ball such that its tangent plane has a normal vector parallel to $\boldsymbol{d}$. For the case $p > 1$, define

$$F(\boldsymbol{\alpha}) = \left(\sum_i \alpha_i^p \right)^{\frac{1}{p}} - 1 = 0, \tag{A.1}$$

as the equation of the points on the surface of the p-unit ball. The j-th coordinate of the normal to the tangent plane of this surface is defined as $\nabla_{\alpha_j} F(\boldsymbol{\alpha}) = \left(\sum_i \alpha_i^p \right)^{\frac{1-p}{p}} \alpha_j^{p-1}$. In order for $\nabla_{\boldsymbol{\alpha}} F(\boldsymbol{\alpha})$ to be parallel to $\boldsymbol{d}$ we have to enforce $\nabla_{\alpha_j} F(\boldsymbol{\alpha}) = \theta d_j$ with $\theta > 0$, which gives:

$$\alpha_j = \left(\frac{\theta d_j}{\left(\sum_i \alpha_i^p \right)^{\frac{1-p}{p}}} \right)^{\frac{1}{p-1}}. \tag{A.2}$$

By plugging (A.2) into (A.1) we obtain $\theta = \frac{\left(\sum_i \alpha_i^p\right)^{\frac{1-p}{p}}}{\|\boldsymbol{d}\|_q}$, which in turn provides the solution $\alpha_j = \left(\frac{d_j}{\|\boldsymbol{d}\|_q}\right)^{q-1}$. Since $\|\boldsymbol{d}\|_q = \left\|(\boldsymbol{V}\boldsymbol{x})^+\right\|_q$, the final solution can thus be written as

$$\beta_V(\boldsymbol{x})_j = \left(\frac{\left|\boldsymbol{v}_j^\top \boldsymbol{x}\right|_+}{\left\|(\boldsymbol{V}\boldsymbol{x})^+\right\|_q}\right)^{q-1}. \tag{A.3}$$

By plugging this solution back into the objective function, we can also write

$$\begin{aligned}
\beta_V(\boldsymbol{x})^\top \boldsymbol{V}\boldsymbol{x} &= \sum_{j=1}^{m}\left(\left(\frac{\left|\boldsymbol{v}_j^\top \boldsymbol{x}\right|_+}{\left\|(\boldsymbol{V}\boldsymbol{x})^+\right\|_q}\right)^{q-1}\boldsymbol{v}_j^\top \boldsymbol{x}\right) \\
&= \frac{1}{\left\|(\boldsymbol{V}\boldsymbol{x})^+\right\|_q^{q-1}}\sum_{j=1}^{m}\left|\boldsymbol{v}_j^\top \boldsymbol{x}\right|_+^q \\
&= \frac{\left\|(\boldsymbol{V}\boldsymbol{x})^+\right\|_q^q}{\left\|(\boldsymbol{V}\boldsymbol{x})^+\right\|_q^{q-1}} = \left\|(\boldsymbol{V}\boldsymbol{x})^+\right\|_q
\end{aligned}$$

□

For the case $p = 1$, the tangent plane is fixed, hence the solution reduces to extreme values of the p-unit ball, and can also be obtained by taking the limit for $p \to 1$ of the solution in Lemma 4.

Proposition 2. *Let $\mathcal{X}_\rho$ be defined as $\mathcal{X}_\rho \equiv \{\boldsymbol{x} \in \mathbb{R}^d : \|\boldsymbol{x}\|_2 \leq \rho\}$, let $\omega : \mathcal{X}_\rho \to \mathbb{R}^d$ be a vector-valued function with d dimensions $\{\omega(\cdot)_i\}_{i=1}^d$ and let $f : \mathcal{X}_\rho \mapsto \mathbb{R}$ be a locally linear function of the form $f(\boldsymbol{x}) = \omega(\boldsymbol{x})^\top \boldsymbol{x}$. If $\omega(\cdot)$ is α-Lipschitz smooth on $\mathcal{X}_\rho$ w.r.t. the ℓ_2-norm and there exist a scalar $\tau \geq 0$ s.t. $\|\omega(\boldsymbol{x})\|_2 \leq \tau$ for every $\boldsymbol{x} \in \mathcal{X}_\rho$, then $f(\cdot)$ is $(\alpha\rho + \tau)$-Lipschitz smooth on $\mathcal{X}_\rho$.*

Proof. Let $\boldsymbol{x}$ and $\boldsymbol{x}'$ be two vectors in $\mathcal{X}_\rho$. We can write:

$$\begin{aligned}
\left|f(\boldsymbol{x}) - f(\boldsymbol{x}')\right| &= \left|\omega(\boldsymbol{x})^\top \boldsymbol{x} - \omega(\boldsymbol{x}')^\top \boldsymbol{x}'\right| \\
&= \left|\left(\sum_{i=1}^{d}\omega(\boldsymbol{x})_i x_i - \sum_{i=1}^{d}\omega(\boldsymbol{x}')_i x_i\right) - \left(\sum_{i=1}^{d}\omega(\boldsymbol{x}')_i x'_i - \sum_{i=1}^{d}\omega(\boldsymbol{x}')_i x_i\right)\right| \\
&\leq \left|\omega(\boldsymbol{x})^\top \boldsymbol{x} - \omega\left(\boldsymbol{x}'\right)^\top \boldsymbol{x}\right| + \left|\sum_{i=1}^{d}\omega(\boldsymbol{x}')_i\left(x'_i - x_i\right)\right| \\
&\leq \left|\left(\omega(\boldsymbol{x}) - \omega\left(\boldsymbol{x}'\right)\right)^\top \boldsymbol{x}\right| + \sum_{i=1}^{d}\left|\omega(\boldsymbol{x}')_i\right|\left|x'_i - x_i\right| \\
&\leq \left\|\omega(\boldsymbol{x}) - \omega\left(\boldsymbol{x}'\right)\right\|\|\boldsymbol{x}\| + \left\|\omega(\boldsymbol{x}')\right\|_2\left\|\boldsymbol{x}' - \boldsymbol{x}\right\|_2 \\
&\leq \alpha\rho\left\|\boldsymbol{x} - \boldsymbol{x}'\right\|_2 + \tau\left\|\boldsymbol{x}' - \boldsymbol{x}\right\|_2 \\
&= \left(\alpha\rho + \tau\right)\left\|\boldsymbol{x} - \boldsymbol{x}'\right\|_2,
\end{aligned}$$

where the third inequality is an application of the Cauchy-Schwarz inequality and for the last inequality we have used the assumptions that $\omega(\cdot)$ is α-Lipschitz and $\|\omega(\boldsymbol{x})\|_2 \leq \tau$, while $\|\boldsymbol{x}\|_2 \leq \rho$. □

Proposition 2 tells us that if the function $\omega(\boldsymbol{x})$ is Lipschitz smooth and has bounded norm, then also the function $f(\boldsymbol{x}) = \omega(\boldsymbol{x})^\top \boldsymbol{x}$ is Lipschitz smooth. As an example, by using Proposition 2 it is easy to show that with $\omega(\boldsymbol{x}) = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}$, the locally linear function $f(\boldsymbol{x}) = \omega(\boldsymbol{x})^\top \boldsymbol{x}$ is $\left(2\rho\|\boldsymbol{A}\|_F + \|\boldsymbol{b}\|_2\right)$-Lipshitz smooth in $\mathscr{X}_\rho$. Please note that for α-Lipschitz smooth functions on $\mathscr{X}_\rho$ the condition $\|\omega(\boldsymbol{x})\|_2 \leq \tau$ is a loose one. Indeed, if there exists a vector $\boldsymbol{z} \in \mathscr{X}_\rho$ s.t. $\omega(\boldsymbol{z}) = \boldsymbol{0}$, then: $\|\omega(\boldsymbol{x})\|_2 = \|\omega(\boldsymbol{x}) - \omega(\boldsymbol{z})\|_2 \leq \alpha\|\boldsymbol{x} - \boldsymbol{z}\|_2 \leq \alpha(\|\boldsymbol{x}\|_2 + \|\boldsymbol{z}\|_2) \leq 2\alpha\rho$. This last condition, in turn, can be granted for a properly transposed version of $\omega(\boldsymbol{x})$ (e.g. by considering the function $\omega(\boldsymbol{x}) - \min_{\boldsymbol{y}\in\mathscr{X}_\rho}\omega(\boldsymbol{y})$).

Lemma (Linearization) 3. *Let $\mathscr{X}_\rho \equiv \{\boldsymbol{x} \in \mathbb{R}^d : \|\boldsymbol{x}\|_2 \leq \rho\}$ and $\omega : \mathscr{X}_\rho \to \mathbb{R}^d$ be a vector-valued function with d dimensions $\{\omega(\cdot)_i\}_{i=1}^d$. Let $f : \mathscr{X}_\rho \mapsto \mathbb{R}$ be a locally linear function of the form $f(\boldsymbol{x}) = \omega(\boldsymbol{x})^\top \boldsymbol{x}$, and let $\boldsymbol{V} = \begin{bmatrix}\boldsymbol{v}_1 & \boldsymbol{v}_2 & \cdots & \boldsymbol{v}_m\end{bmatrix}^\top \in \mathbb{R}^{m\times d}$ be an arbitrary $m \times d$ matrix. If $\omega(\cdot)$ is α-Lipschitz smooth on $\mathscr{X}_\rho$ w.r.t. the ℓ_2-norm, then for all $\boldsymbol{x} \in \mathscr{X}_\rho$ and $\boldsymbol{\beta} \in \mathbb{R}^m$ s.t. $\boldsymbol{\beta} \geq 0$ and $\|\boldsymbol{\beta}\|_1 = 1$, the following inequality holds, for some $\gamma \geq 0$ that depends on the specific form of $\omega(\cdot)$:*

$$\left| f(\boldsymbol{x}) - \sum_{i=1}^m \beta_i \omega(\boldsymbol{v}_i)^\top \boldsymbol{x} \right| \leq \rho\gamma \left\|\boldsymbol{x} - \boldsymbol{V}^\top \boldsymbol{\beta}\right\|_2 + \rho\left(\gamma + \alpha\right) \sum_{i=1}^m \beta_i \|\boldsymbol{x} - \boldsymbol{v}_i\|_2 . \tag{A.4}$$

Proof. Let us express $\omega(\cdot)$ as $\omega(\cdot) = \omega_1(\cdot) + \omega_2(\cdot)$, where $\omega_1 : \mathscr{X}_\rho \mapsto \mathbb{R}^d$ is a linear map and $\omega_2 : \mathscr{X}_\rho \mapsto \mathbb{R}^d$ is a remainder s.t. $\omega(\cdot)$ satisfies the hypothesis of this lemma (namely, that $\omega(\cdot)$ is α-Lipschitz smooth on $\mathscr{X}_\rho$).
Please note that it is always possible to express $\omega(\cdot)$ in this form, as the constant function $\omega_1(\cdot) = \boldsymbol{0}$ is a valid linear map. Please note also that, since $\omega_1(\boldsymbol{x})$ is linear, $\omega_1(\boldsymbol{x})^\top \boldsymbol{x}$ is a quadratic function of $\boldsymbol{x}$. Therefore, by assuming $\omega(\cdot)$ to have this form we are assuming that $f(\boldsymbol{x})$ may be composed of a smooth quadratic part, and a reminder which can be anything, as long as $\omega_1(\cdot) + \omega_2(\cdot)$ is α-Lipschitz smooth.

First we show that since $\omega_1(\cdot) + \omega_2(\cdot)$ is assumed to be α-Lipschitz smooth and ω_1 linear, also on ω_2 is Lipschitz smooth. Specifically, since $\omega_1(\boldsymbol{x})$ is a linear map, $\omega_1(\boldsymbol{x})$ is γ-Lipschitz for some $\gamma \geq 0$. Consequently, using $\omega_2(\boldsymbol{x}) = \omega(\boldsymbol{x}) - \omega_1(\boldsymbol{x})$, for any $\boldsymbol{x}$ and $\boldsymbol{x}'$ in $\mathscr{X}_\rho$ we have:

$$\begin{aligned}
\left\|\omega_2(\boldsymbol{x}) - \omega_2(\boldsymbol{x}')\right\| &= \left\|\left(\omega(\boldsymbol{x}) - \omega(\boldsymbol{x}')\right) + \left(\omega_1(\boldsymbol{x}') - \omega_1(\boldsymbol{x})\right)\right\| \\
&\leq \left\|\omega(\boldsymbol{x}) - \omega(\boldsymbol{x}')\right\| + \left\|\omega_1(\boldsymbol{x}') - \omega_1(\boldsymbol{x})\right\| \\
&\leq \alpha\left\|\boldsymbol{x} - \boldsymbol{x}'\right\| + \gamma\left\|\boldsymbol{x}' - \boldsymbol{x}\right\| \\
&= \left(\alpha + \gamma\right)\left\|\boldsymbol{x} - \boldsymbol{x}'\right\| .
\end{aligned}$$

Please note that in case $\omega_1(\cdot) = \boldsymbol{0}$, we have $\gamma = 0$, so that $\omega(\cdot) = \omega_2(\cdot)$ is once again α-Lipschitz.

Using this representation of $\omega(\cdot)$, we can write:

$$\begin{aligned}
\left| f(\boldsymbol{x}) - \sum_i \beta_i \omega(\boldsymbol{v}_i)^\top \boldsymbol{x} \right| &= \left| \omega_1(\boldsymbol{x})^\top \boldsymbol{x} - \sum_i \beta_i \omega_1(\boldsymbol{v}_i)^\top \boldsymbol{x} + \omega_2(\boldsymbol{x})^\top \boldsymbol{x} - \sum_i \beta_i \omega_2(\boldsymbol{v}_i)^\top \boldsymbol{x} \right| \\
&\leq \left| \omega_1(\boldsymbol{x})^\top \boldsymbol{x} - \sum_i \beta_i \omega_1(\boldsymbol{v}_i)^\top \boldsymbol{x} \right| + \left| \omega_2(\boldsymbol{x})^\top \boldsymbol{x} - \sum_i \beta_i \omega_2(\boldsymbol{v}_i)^\top \boldsymbol{x} \right| \\
&= \left| \left(\omega_1(\boldsymbol{x}) - \sum_i \beta_i \omega_1(\boldsymbol{v}_i) \right)^\top \boldsymbol{x} \right| + \left| \left(\sum_i \beta_i \omega_2(\boldsymbol{x}) - \sum_i \beta_i \omega_2(\boldsymbol{v}_i) \right)^\top \boldsymbol{x} \right| \\
&\leq \left| \left(\omega_1(\boldsymbol{x}) - \omega_1\left(\sum_i \beta_i \boldsymbol{v}_i \right) \right)^\top \boldsymbol{x} \right| + \sum_i \beta_i \left| (\omega_2(\boldsymbol{x}) - \omega_2(\boldsymbol{v}_i))^\top \boldsymbol{x} \right| \\
&\leq \left\| \omega_1(\boldsymbol{x})_2 - \omega_1\left(\sum_i \beta_i \boldsymbol{v}_i \right) \right\|_2 \|\boldsymbol{x}\|_2 + \sum_i \beta_i \|\omega_2(\boldsymbol{x}) - \omega_2(\boldsymbol{v}_i)\|_2 \|\boldsymbol{x}\|_2 \\
&\leq \rho\gamma \left\| \boldsymbol{x} - \sum_i \beta_i \boldsymbol{v}_i \right\|_2 + \rho(\gamma + \alpha) \sum_i \beta_i \|\boldsymbol{x} - \boldsymbol{v}_i\|_2 \\
&= \rho\gamma \left\| \boldsymbol{x} - \boldsymbol{V}^\top \boldsymbol{\beta} \right\|_2 + \rho(\gamma + \alpha) \sum_i \beta_i \|\boldsymbol{x} - \boldsymbol{v}_i\|_2 ,
\end{aligned}$$

where we have used the fact that, by the linearity of ω_1 we have $\sum_i \beta_i \omega_1(\boldsymbol{v}_i) = \omega_1\left(\sum_i \beta_i \boldsymbol{v}_i\right)$, and in the last two inequalities we applied the Cauchy-Schwarz inequality and the facts that ω_1 is γ-Lipschitz and ω_2 is $(\gamma + \alpha)$-Lipschitz. □

Proposition 4. *Let* $\boldsymbol{V} = \begin{bmatrix} \boldsymbol{v}_1 & \boldsymbol{v}_2 & \cdots & \boldsymbol{v}_m \end{bmatrix}^\top \in \mathbb{R}^{m \times d}$ *be an arbitrary matrix in* $\mathbb{R}^{m \times d}$, $\boldsymbol{x}$ *be an arbitrary vector in* $\mathbb{R}^d$ *and* $\boldsymbol{\beta} \in \mathbb{R}^m$ *be a vector such that* $\boldsymbol{\beta} \geq 0$. *If* $\|\boldsymbol{\beta}\|_p \leq 1$, *with* $p > 0$, *the following inequality holds:*

$$\sum_{i=1}^m \beta_i \|\boldsymbol{x} - \boldsymbol{v}_i\|_2 \leq \left(\|\boldsymbol{\beta}\|_1 \left(\|\boldsymbol{\beta}\|_1 \|\boldsymbol{x}\|_2^2 + \|\boldsymbol{V}\|_F^2 - 2\boldsymbol{\beta}^\top \boldsymbol{V}\boldsymbol{x} \right) \right)^{\frac{1}{2}}$$

Moreover, if $p \leq 2$, *the following inequality holds as well:*

$$\left\| \boldsymbol{x} - \boldsymbol{V}^\top \boldsymbol{\beta} \right\|_2 \leq \left(\|\boldsymbol{x}\|_2^2 + \|\boldsymbol{V}\|_F^2 - 2\boldsymbol{\beta}^\top \boldsymbol{V}\boldsymbol{x} \right)^{\frac{1}{2}}$$

Proof. For the first inequality we proceed as follow. For any vector $\boldsymbol{\beta} \geq 0$ with an unknown but fixed ℓ_1-norm $\|\boldsymbol{\beta}\|_1$ we can apply Jensen's inequality to get

$$\begin{aligned}
\left(\sum_{i=1}^m \frac{\beta_i}{\|\boldsymbol{\beta}\|_1} \|\boldsymbol{x} - \boldsymbol{v}_i\|_2 \right)^2 &\leq \sum_{i=1}^m \frac{\beta_i}{\|\boldsymbol{\beta}\|_1} \|\boldsymbol{x} - \boldsymbol{v}_i\|_2^2 \\
\left(\sum_{i=1}^m \beta_i \|\boldsymbol{x} - \boldsymbol{v}_i\|_2 \right)^2 &\leq \|\boldsymbol{\beta}\|_1 \sum_{i=1}^m \beta_i \|\boldsymbol{x} - \boldsymbol{v}_i\|_2^2
\end{aligned}$$

Moreover, since $\boldsymbol{\beta} \geq 0$ and $\left\|\boldsymbol{\beta}\right\|_p \leq 1$, we also have $\sum_{i=1}^m \beta_i \left\|\boldsymbol{v}_i\right\|^2 \leq \sum_{i=1}^m \left\|\boldsymbol{v}_i\right\|^2 = \left\|\boldsymbol{V}\right\|_F^2$, so that

$$\begin{aligned}\sum_{i=1}^m \beta_i \left\|\boldsymbol{x} - \boldsymbol{v}_i\right\|_2^2 &= \left\|\boldsymbol{\beta}\right\|_1 \left\|\boldsymbol{x}\right\|_2^2 + \sum_{i=1}^m \beta_i \left\|\boldsymbol{v}_i\right\|^2 - 2\sum_{i=1}^m \beta_i \boldsymbol{v}_i^\top \boldsymbol{x} \\ &\leq \left\|\boldsymbol{\beta}\right\|_1 \left\|\boldsymbol{x}\right\|_2^2 + \left\|\boldsymbol{V}\right\|_F^2 - 2\boldsymbol{\beta}^\top \boldsymbol{V}\boldsymbol{x},\end{aligned}$$

which gives the desired result.

The second inequality is due to the fact that by Cauchy-Schwarz and the equivalence of norms:

$$\left\|\boldsymbol{V}^\top \boldsymbol{\beta}\right\|_2^2 = \sum_{j=1}^d \left|\sum_{i=1}^m \beta_i v_{ij}\right|^2 \leq \sum_{j=1}^d \sum_{i=1}^m \beta_i^2 \sum_{k=1}^m v_{kj}^2 = \left\|\boldsymbol{\beta}\right\|_2^2 \left\|\boldsymbol{V}\right\|_F^2 \leq \left\|\boldsymbol{\beta}\right\|_p^2 \left\|\boldsymbol{V}\right\|_F^2 \leq \left\|\boldsymbol{V}\right\|_F^2.$$

It is therefore immediate to write

$$\left\|\boldsymbol{x} - \boldsymbol{V}^\top \boldsymbol{\beta}\right\|_2^2 = \left\|\boldsymbol{x}\right\|_2^2 + \left\|\boldsymbol{V}^\top \boldsymbol{\beta}\right\|_2^2 - 2\boldsymbol{\beta}^\top \boldsymbol{V}\boldsymbol{x} \leq \left\|\boldsymbol{x}\right\|_2^2 + \left\|\boldsymbol{V}\right\|_F^2 - 2\boldsymbol{\beta}^\top \boldsymbol{V}\boldsymbol{x}.$$

□

B Visualizations

We report here some visualizations of saliency maps on the MIT67-indoor scene recognition dataset.

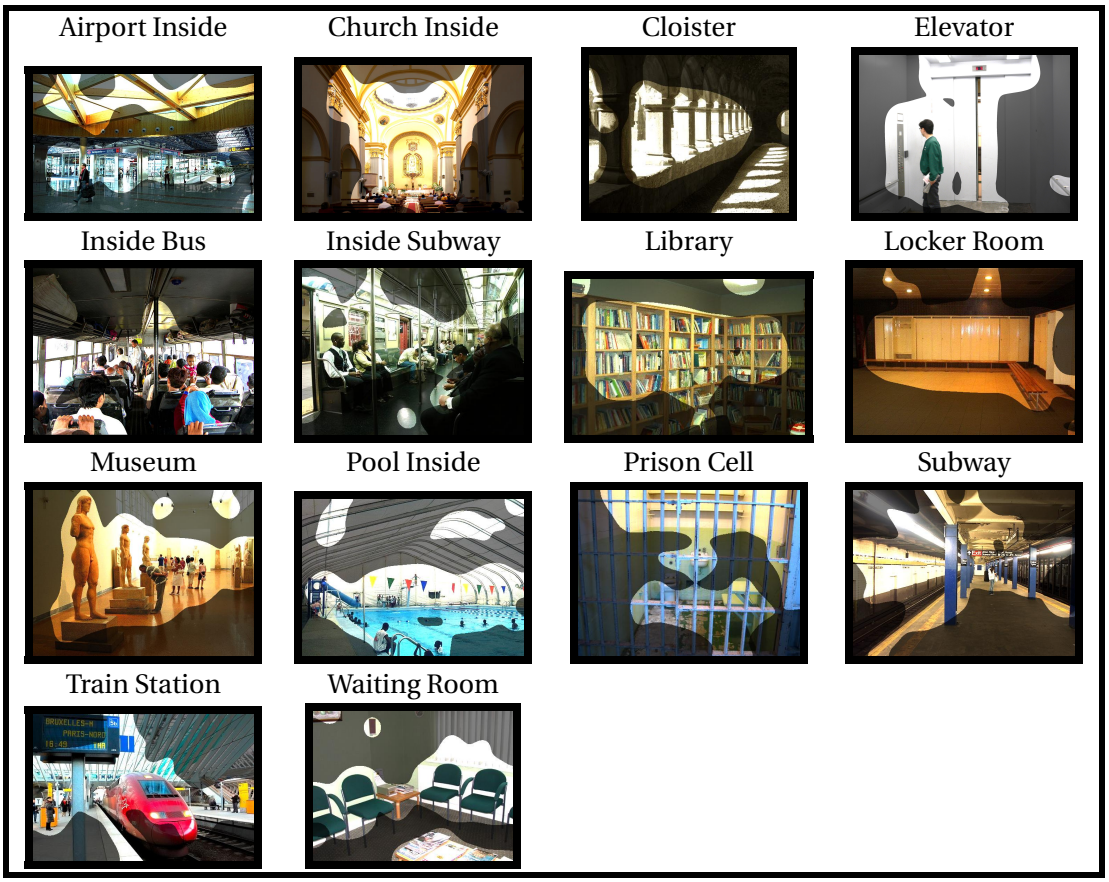

Figure B.1 – Visualizations of the segmentation masks obtained using Itti Saliency on images from the 14 categories of the “Public Spaces” macrogroup.

Bibliography

Fabio Aiolli and Alessandro Sperduti. Multiclass classification with multi-prototype support vector machines. Journal of Machine Learning Research, 6:817–850, 2005.

Yonatan Amit, Michael Fink, Nathan Srebro, and Shimon Ullman. Uncovering shared structures in multiclass classification. In Proc. of International Conference on Machine Learning, ICML, pages 17–24, 2007.

Stuart Andrews, Ioannis Tsochantaridis, and Thomas Hofmann. Support vector machines for multiple-instance learning. In Proc. of Neural Information Processing Systems, NIPS, pages 561–568, 2002.

Sunil Arya and Ho-Yam Addy Fu. Expected-case complexity of approximate nearest neighbor searching. SIAM Journal on Computing, 32(3):793–815, 2003.

Régis Behmo, Paul Marcombes, Arnak S. Dalalyan, and Véronique Prinet. Towards optimal naive bayes nearest neighbor. In Proc. of European Conference on Computer Vision, ECCV, pages 171–184, 2010.

Yoshua Bengio, Olivier Delalleau, and Nicolas Le Roux. The curse of highly variable functions for local kernel machines. In Proc. of Neural Information Processing Systems, NIPS, 2005.

Dimitri Panteli Bertsekas. Nonlinear Programming. Athena Scientific, 2nd edition, 1999. ISBN 1886529000.

Chirstopher M. Bishop. Pattern recognition and machine learning, volume 4. Springer New York, 2006.

Jock Blackard and Denis J. Dean. Comparative accuracies of artificial neural networks and discriminant analysis in predicting forest cover types from cartographic variables. Computers and Electronics in Agriculture, 24(3):131–151, December 1999.

Oren Boiman, Eli Shechtman, and Michal Irani. In defense of nearest-neighbor based image classification. In Proc. of International Conference on Computer Vision and Pattern Recognition, CVPR, 2008.

Antoine Bordes, Léon Bottou, and Patrick Gallinari. Sgd-qn: Careful quasi-newton stochastic gradient descent. Journal of Machine Learning Research, 10:1737–1754, 2009.

Anna Bosch, Andrew Zisserman, and Xavier Muñoz. Representing shape with a spatial pyramid kernel. In Proc. of Conference on Image and Video Retrieval, CIVR, pages 401–408, 2007.

Léon Bottou and Vladimir Vapnik. Local learning algorithms. Neural Computation, 4(6): 888–900, 1992.

Y-Lan Boureau, Jean Ponce, and Yann LeCun. A theoretical analysis of feature pooling in visual recognition. In Proc. of International Conference on Machine Learning, ICML, pages 111–118, 2010.

Y-Lan Boureau, Nicolas Le Roux, Francis Bach, Jean Ponce, and Yann LeCun. Ask the locals: Multi-way local pooling for image recognition. In Proc. International Conference on Computer Vision, ICCV, pages 2651–2658, 2011.

Stephen Boyd and Lieven Vandenberghe. Convex Optimization. Cambridge University Press, New York, NY, USA, 2004.

Neil D. B. Bruce and John K. Tsotsos. Saliency based on information maximization. In Proc. of Neural Information Processing Systems, NIPS, 2005.

Yang Cao, Changhu Wang, Zhiwei Li, Liqing Zhang, and Lei Zhang. Spatial-bag-of-features. In Proc. of International Conference on Computer Vision and Pattern Recognition, CVPR, pages 3352–3359, 2010.

Barbara Caputo and Luo Jie. A performance evaluation of exact and approximate match kernels for object recognition. Electronic Letters on Computer Vision and Image Analysis, 8(3):15–26, 2009.

Fatih Çakir, Ugur Güdükbay, and Özgür Ulusoy. Nearest-neighbor based metric functions for indoor scene recognition. Computer Vision and Image Understanding, 115(11):1483–1492, 2011.

Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. ACM Transactions on Intelligent Systems and Technology, 2:27:1–27:27, 2011. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

Ken Chatfield, Victor S. Lempitsky, Andrea Vedaldi, and Andrew Zisserman. The devil is in the details: an evaluation of recent feature encoding methods. In Proc. of British Machine Vision Conference, BMVC, pages 1–12, 2011.

Haibin Cheng, Pang-Ning Tan, and Rong Jin. Efficient algorithm for localized support vector machine. IEEE Trans. Knowl. Data Eng., 22(4):537–549, 2010.

Andrew Cotter, Joseph Keshet, and Nathan Srebro. Explicit approximations of the gaussian kernel. ACM Computing Research Repository, CoRR, abs/1109.4603, 2011.

Thomas M. Cover and Peter E. Hart. Nearest neighbor pattern classification. IEEE Transactions on Information Theory, 13(1):21–27, 1967.

Koby Crammer and Yoram Singer. On the algorithmic implementation of multiclass kernel-based vector machines. Journal of Machine Learning Research, 2:265–292, 2001.

Nello Cristianini and John Shawe-Taylor. An Introduction to Support Vector Machines and Other Kernel-based Learning Methods. Cambridge University Press, 2010. ISBN 978-0-521-78019-3.

Gabriella Csurka, Christopher Dance, Lixin Fan, Jutta Willamowski, and Cédric Bray. Visual categorization with bags of keypoints. In Proc. of workshop on statistical learning in computer vision, ECCV, volume 1, page 22, 2004.

Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In Proc. of International Conference on Computer Vision and Pattern Recognition, CVPR, pages 886–893, 2005.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In Proc. of International Conference on Computer Vision and Pattern Recognition, CVPR, pages 248–255, 2009.

Santosh Kumar Divvala, Alexei A. Efros, and Martial Hebert. How important are "deformable parts" in the deformable parts model? In Proc. of European Conference on Computer Vision, ECCV Workshops, pages 31–40, 2012.

Carl Doersch, Abhinav Gupta, and Alexei A. Efros. Mid-level visual element discovery as discriminative mode seeking. In Proc. of Neural Information Processing Systems, NIPS, pages 494–502, 2013.

Piotr Dollár, Boris Babenko, Serge J. Belongie, Pietro Perona, and Zhuowen Tu. Multiple component learning for object detection. In Proc. of European Conference on Computer Vision, ECCV, pages 211–224, 2008.

Charles Dubout and François Fleuret. Exact acceleration of linear object detectors. In Proc. of European Conference on Computer Vision, ECCV, pages 301–311, 2012. doi: 10.1007/978-3-642-33712-3_22. URL http://dx.doi.org/10.1007/978-3-642-33712-3_22.

Krista Anne Ehinger. Visual Features for Scene Recognition and Reorientation. PhD thesis, Massachusetts Institute of Technology, Department of Brain and Cognitive Sciences, 2013.

Vassiliy A. Epanechnikov. Non-parametric estimation of a multivariate probability density. Theory of Probability & Its Applications, 14(1):153–158, 1969.

Hugo Jair Escalante, Mauricio Sotomayor, Manuel Montes y Gómez, and Adrián Pastor López-Monroy. Object recognition with näive bayes-nn via prototype generation. In Proc. of Mexican Conference on Pattern Recognition, MCPR, pages 162–171, 2014.

Mark Everingham, Luc J. Van Gool, Christopher K. I. Williams, John M. Winn, and Andrew Zisserman. The PASCAL Visual Object Classes Challenge 2006 (VOC2006) Results. http://www.pascal-network.org/challenges/VOC/voc2006/results.pdf, 2006.

Mark Everingham, Luc J. Van Gool, Christopher K. I. Williams, John M. Winn, and Andrew Zisserman. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results. http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html, 2007.

Mark Everingham, Luc J. Van Gool, Christopher K. I. Williams, John M. Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. International Journal of Computer Vision, 88(2):303–338, 2010.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. Liblinear: A library for large linear classification. Journal of Machine Learning Research, 9:1871–1874, 2008.

Li Fei-Fei and Pietro Perona. A bayesian hierarchical model for learning natural scene categories. In Proc. of International Conference on Computer Vision and Pattern Recognition, CVPR, volume 2, pages 524–531. IEEE, 2005.

Li Fei-Fei, Robert Fergus, and Pietro Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. Computer Vision and Image Understanding, 106(1):59–70, 2007.

Christiane Fellbaum. WordNet. Wiley Online Library, 1998.

Pedro F. Felzenszwalb, David A. McAllester, and Deva Ramanan. A discriminatively trained, multiscale, deformable part model. In Proc. of International Conference on Computer Vision and Pattern Recognition, CVPR, 2008.

Pedro F. Felzenszwalb, Ross B. Girshick, David A. McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. IEEE Trans. Pattern Anal. Mach. Intell., 32(9):1627–1645, 2010.

Jiashi Feng, Bingbing Ni, Qi Tian, and Shuicheng Yan. Geometric ℓ_p-norm feature pooling for image classification. In Proc. of International Conference on Computer Vision and Pattern Recognition, CVPR, pages 2697–2704, 2011.

Marco Fornoni and Barbara Caputo. Indoor scene recognition using task and saliency-driven feature pooling. In Proc. of British Machine Vision Conference, BMVC, pages 1–12, 2012.

Marco Fornoni and Barbara Caputo. Scene recognition with naive bayes non-linear learning. In Proc. of the 22nd International Conference on Pattern Recognition (ICPR). IEEE, August 2014.

Marco Fornoni, Jesus Martínez-Gómez, and Barbara Caputo. A multi cue discriminative approach to semantic place classification. In Proc. of CLEF (Notebook Papers/LABs/Workshops), 2010.

Marco Fornoni, Barbara Caputo, and Francesco Orabona. Multiclass latent locally linear support vector machines. In Cheng Soon Ong and Tu-Bao Ho, editors, JMLR W&CP, Volume 29: ACML, pages 229–244, 2013.

Charless Fowlkes, Serge Belongie, Fan R. K. Chung, and Jitendra Malik. Spectral grouping using the nyström method. IEEE Trans. Pattern Anal. Mach. Intell., 26(2):214–225, 2004.

Andrew J. Frank and Arthur Asuncion. UCI machine learning repository, 2010. URL http://archive.ics.uci.edu/ml.

Brendan J. Frey and Delbert Dueck. Clustering by passing messages between data points. Science, 315:2007, 2007.

Zhouyu Fu, Antonio Robles-Kelly, and Jun Zhou. Mixing linear svms for nonlinear classification. IEEE Transactions on Neural Networks, 21(12):1963–1975, 2010.

Dashan Gao and Nuno Vasconcelos. Integrated learning of saliency, complex features, and object detectors from cluttered scenes. In Proc. of International Conference on Computer Vision and Pattern Recognition, CVPR, pages 282–287, 2005.

Salvador García, Joaquín Derrac, José Ramón Cano, and Francisco Herrera. Prototype selection for nearest neighbor classification: Taxonomy and empirical study. IEEE Trans. Pattern Anal. Mach. Intell., 34(3):417–435, 2012.

Peter V. Gehler and Sebastian Nowozin. On feature combination for multiclass object classification. In Proc. of International Conference on Computer Vision, ICCV, pages 221–228, 2009.

Mehmet Gönen and Ethem Alpaydin. Localized multiple kernel learning. In Proc. of International Conference on Machine Learning, ICML, pages 352–359. ACM, 2008.

Gregory Griffin, Alex Holub, and Pietro Perona. The Caltech 256. Technical report, California Institute of Technology, 2006.

Chunhui Gu, Pablo Andrés Arbeláez, Yuanqing Lin, Kai Yu, and Jitendra Malik. Multi-component models for object detection. In Proc. of European Conference on Computer Vision, ECCV, pages 445–458, 2012.

Efstathios Hadjidemetriou, Michael D. Grossberg, and Shree K. Nayar. Multiresolution histograms and their use for recognition. IEEE Trans. Pattern Anal. Mach. Intell., 26(7):831–847, 2004.

Tatsuya Harada, Yoshitaka Ushiku, Yuya Yamashita, and Yasuo Kuniyoshi. Discriminative spatial pyramid. In Proc. of International Conference on Computer Vision and Pattern Recognition, CVPR, pages 1617–1624, 2011.

Jonathan Harel. A saliency implementation in matlab @ONLINE, 2006. URL http://www.klab.caltech.edu/~harel/share/gbvs.php.

Jonathan Harel, Christof Koch, and Pietro Perona. Graph-based visual saliency. In Proc. of Neural Information Processing Systems, NIPS, pages 545–552, 2006.

Minh Hoai and Andrew Zisserman. Discriminative sub-categorization. In Proc. of International Conference on Computer Vision and Pattern Recognition, CVPR, pages 1666–1673, 2013.

Xiaodi Hou, Jonathan Harel, and Christof Koch. Image signature: Highlighting sparse salient regions. IEEE Trans. Pattern Anal. Mach. Intell., 34(1):194–201, 2012.

Jonathan J. Hull. A database for handwritten text recognition research. IEEE Trans. Pattern Anal. Mach. Intell., 16(5), 1994.

Aapo Hyvärinen and Erkki Oja. Independent component analysis: algorithms and applications. Neural Networks, 13(4-5):411–430, 2000.

Laurent Itti, Christof Koch, and Ernst Niebur. A model of saliency-based visual attention for rapid scene analysis. IEEE Trans. Pattern Anal. Mach. Intell., 20(11):1254–1259, 1998.

Tommi Jaakkola and David Haussler. Exploiting generative models in discriminative classifiers. In Proc. of Neural Information Processing Systems, NIPS, pages 487–493, 1998.

Robert A. Jacobs, Michael I. Jordan, Steven J. Nowlan, and Geoffrey E. Hinton. Adaptive mixtures of local experts. Neural Computation, 3(1):79–87, March 1991.

Anil K. Jain, Robert P. W. Duin, and Jianchang Mao. Statistical pattern recognition: A review. IEEE Trans. Pattern Anal. Mach. Intell., 22(1):4–37, 2000.

Herve Jegou, Matthijs Douze, Cordelia Schmid, and Patrick Pérez. Aggregating local descriptors into a compact image representation. In Proc. of International Conference on Computer Vision and Pattern Recognition, CVPR, pages 3304–3311, 2010.

Yangqing Jia, Chang Huang, and Trevor Darrell. Beyond spatial pyramids: Receptive field learning for pooled image features. In Proc. of International Conference on Computer Vision and Pattern Recognition, CVPR, pages 3370–3377, 2012.

Yuning Jiang, Junsong Yuan, and Gang Yu. Randomized spatial partition for scene recognition. In Proc. of European Conference on Computer Vision, ECCV, pages 730–743, 2012.

Thorsten Joachims. Text categorization with suport vector machines: Learning with many relevant features. In Proc. of European Conference on Machine Learning, ECML, pages 137–142, London, UK, UK, 1998. Springer-Verlag. ISBN 3-540-64417-2. URL http://dl.acm.org/citation.cfm?id=645326.649721.

Thorsten Joachims. Training linear svms in linear time. In Proc. of Conference on Knowledge Discovery and Data Mining, KDD, pages 217–226, 2006.

Mayank Juneja, Andrea Vedaldi, C. V. Jawahar, and Andrew Zisserman. Blocks that shout: Distinctive parts for scene classification. In Proc. of International Conference on Computer Vision and Pattern Recognition, CVPR, pages 923–930, 2013.

Christopher Kanan and Garrison Cottrell. Robust classification of objects, faces, and flowers using natural image statistics. In Proc. of International Conference on Computer Vision and Pattern Recognition, CVPR, pages 2472–2479. IEEE, 2010.

Vojislav Kecman and J. Paul Brooks. Locally linear support vector machines and other local models. In Proc. of International Joint Conference on Neural Networks, IJCNN, pages 1–6, 2010.

Teuvo Kohonen, Jussi Hynninen, Jari Kangas, Jorma Laaksonen, and Kari Torkkola. Lvq_pak – the learning vector quantization network program package. Technical report, Technical report, Laboratory of Computer and Information Science Rakentajanaukio 2 C, 1991-1992, 1996.

Piotr Koniusz and Krystian Mikolajczyk. Spatial coordinate coding to reduce histogram representations, dominant angle and colour pyramid match. In Proc. of International Conference on Image Processing, ICIP, pages 661–664, 2011.

Josip Krapac, Jakob J. Verbeek, and Frédéric Jurie. Modeling spatial layout with fisher vectors for image categorization. In Proc. International Conference on Computer Vision, ICCV, pages 1487–1494, 2011.

Ludmila I. Kuncheva. Combining pattern classifiers: methods and algorithms. John Wiley & Sons, 2004.

Roland Kwitt, Nuno Vasconcelos, and Nikhil Rasiwasia. Scene recognition on the semantic manifold. In Proc. of European Conference on Computer Vision, ECCV, pages 359–372, 2012.

Lubor Ladicky and Philip H. S. Torr. Locally linear support vector machines. In Proc. of International Conference on Machine Learning, ICML, pages 985–992, 2011.

Tian Lan, Michalis Raptis, Leonid Sigal, and Greg Mori. From subcategories to visual composites: A multi-level framework for object detection. In Proc. International Conference on Computer Vision, ICCV, pages 369–376, 2013.

Marc Teva Law, Nicolas Thome, and Matthieu Cord. Hybrid pooling fusion in the bow pipeline. In Proc. of European Conference on Computer Vision, ECCV Workshops, pages 355–364, 2012.

Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In Proc. of International Conference on Computer Vision and Pattern Recognition, CVPR, pages 2169–2178, 2006.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. Proc. of the IEEE, 86(11):2278–2324, November 1998.

Bastian Leibe and Bernt Schiele. Analyzing appearance and contour based methods for object categorization. In Proc. of International Conference on Computer Vision and Pattern Recognition, CVPR, pages 409–415, 2003.

Li-Jia Li and Li Fei-Fei. What, where and who? classifying events by scene and object recognition. In Proc. of International Conference on Computer Vision, ICCV, pages 1–8, 2007.

Li-Jia Li, Hao Su, Eric P. Xing, and Li Fei-Fei. Object bank: A high-level image representation for scene classification & semantic feature sparsification. In Proc. of Neural Information Processing Systems, NIPS, pages 1378–1386, 2010.

Quannan Li, Jiajun Wu, and Zhuowen Tu. Harvesting mid-level visual concepts from large-scale internet images. In Proc. of International Conference on Computer Vision and Pattern Recognition, CVPR, pages 851–858, 2013.

Oskar Linde and Tony Lindeberg. Object recognition using composed receptive field histograms of higher dimensionality. In Proc. of the 17th International Conference on Pattern Recognition, ICPR, pages 1–6, 2004.

David G. Lowe. Distinctive image features from scale-invariant keypoints. International journal of computer vision, 60(2):91–110, 2004.

Jie Luo, Andrzej Pronobis, Barbara Caputo, and Patric Jensfelt. The KTH-IDOL2 Database. Technical Report CVAP304, KTH Royal Institute of Technology, CVAP/CAS, Stockholm, Sweden, October 2006. URL http://www.pronobis.pro/publications/luo2006idol2.

James MacQueen. Some methods for classification and analysis of multivariate observations. In Proc. of the fifth Berkeley symposium on mathematical statistics and probability, volume 1, page 14. California, USA, 1967.

Julien Mairal. Stochastic majorization-minimization algorithms for large-scale optimization. In Proc. of Neural Information Processing Systems, NIPS, pages 2283–2291, 2013.

Julien Mairal, Francis Bach, Jean Ponce, Guillermo Sapiro, and Andrew Zisserman. Discriminative learned dictionaries for local image analysis. In Proc. of International Conference on Computer Vision and Pattern Recognition, CVPR, 2008.

Julien Mairal, Francis Bach, Jean Ponce, and Guillermo Sapiro. Online dictionary learning for sparse coding. In Proc. of International Conference on Machine Learning, ICML, page 87, 2009.

Tomasz Malisiewicz, Abhinav Gupta, and Alexei A. Efros. Ensemble of exemplar-svms for object detection and beyond. In Proc. International Conference on Computer Vision, ICCV, pages 89–96, 2011.

David Marr. Vision: A Computational Investigation into the Human Representation and Processing of Visual Information. Henry Holt and Co., Inc., New York, NY, USA, 1982. ISBN 0716715678.

Marcin Marszalek and Cordelia Schmid. Spatial weighting for bag-of-features. In Proc. of International Conference on Computer Vision and Pattern Recognition, CVPR, pages 2118–2125, 2006.

Marcin Marszałek, Cordelia Schmid, Hedi Harzallah, and Joost Van De Weijer. Learning object representations for visual object class recognition. In Proc. of Visual Recognition Challange workshop, in conjunction with ICCV, 2007.

Sancho McCann and David G. Lowe. Spatially local coding for object recognition. In Proc. of Asian Conference on Computer Vision, ACCV, pages 204–217, 2012a.

Sancho McCann and David G. Lowe. Local naive bayes nearest neighbor for image classification. In Proc. of International Conference on Computer Vision and Pattern Recognition, CVPR, pages 3650–3656. IEEE, 2012b.

Frank Moosmann, Diane Larlus, and Frederik Jurie. Learning saliency maps for object categorization. In Proc. of ECCV Workshop on the Representation and Use of Prior Knowledge in Vision, 2006.

Marius Muja and David G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In Proc. of International Conference on Computer Vision Theory and Application, VISSAPP), pages 331–340. INSTICC Press, 2009.

Milind R. Naphade, John R. Smith, Jelena Tesic, Shih-Fu Chang, Winston H. Hsu, Lyndon S. Kennedy, Alexander G. Hauptmann, and Jon Curtis. Large-scale concept ontology for multimedia. IEEE MultiMedia, 13(3):86–91, 2006.

Aude Oliva and Antonio Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. International journal of computer vision, 42(3):145–175, 2001.

Bruno A. Olshausen and David J. Fieldt. Sparse coding with an overcomplete basis set: a strategy employed by v1. Vision Research, 37:3311–3325, 1997.

Megha Pandey and Svetlana Lazebnik. Scene recognition and weakly supervised object localization with deformable part-based models. In Proc. of International Conference on Computer Vision, ICCV, pages 1307–1314, 2011.

Devi Parikh, C. Lawrence Zitnick, and Tsuhan Chen. Determining patch saliency using low-level context. In Proc. of European Conference on Computer Vision, ECCV, pages 446–459, 2008.

Sobhan Naderi Parizi, John G. Oberlin, and Pedro F. Felzenszwalb. Reconfigurable models for scene recognition. In Proc. of International Conference on Computer Vision and Pattern Recognition, CVPR, pages 2775–2782, 2012.

Florent Perronnin and Christopher R. Dance. Fisher kernels on visual vocabularies for image categorization. In Proc. of International Conference on Computer Vision and Pattern Recognition, CVPR, 2007.

Florent Perronnin, Jorge Sánchez, and Thomas Mensink. Improving the fisher kernel for large-scale image classification. In Proc. of European Conference on Computer Vision, ECCV, pages 143–156, 2010.

Andrzej Pronobis and Barbara Caputo. The KTH-INDECS Database. Technical Report CVAP297, KTH Royal Institute of Technology, CVAP/CAS, Stockholm, Sweden, September 2005. URL http://www.pronobis.pro/publications/pronobis2005indecs.

Andrzej Pronobis and Barbara Caputo. Cold: The cosy localization database. I. J. Robotic Res., 28(5):588–594, 2009.

Andrzej Pronobis, Marco Fornoni, Henrik I. Christensen, and Barbara Caputo. The robot vision track at imageclef 2010. In Proc. of CLEF (Notebook Papers/LABs/Workshops), 2010.

Guojun Qi, Qi Tian, and Thomas S. Huang. Locality-sensitive support vector machine by exploring local correlation and global regularization. In Proc. of International Conference on Computer Vision and Pattern Recognition, CVPR, pages 841–848, 2011.

Ariadna Quattoni and Antonio Torralba. Recognizing indoor scenes. In Proc. of International Conference on Computer Vision and Pattern Recognition, CVPR, pages 413–420, 2009.

Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In Proc. of Neural Information Processing Systems, NIPS, 2007.

Konstantinos Rapantzikos, Nicolas Tsapatsoulis, Yannis S. Avrithis, and Stefanos D. Kollias. Spatiotemporal saliency for video classification. Sig. Proc.: Image Comm., 24(7):557–571, 2009.

Konstantinos Rematas, Mario Fritz, and Tinne Tuytelaars. The pooled nbnn kernel: beyond image-to-class and image-to-image. In Proc. of Asian Conference on Computer Vision, ACCV, pages 176–189. Springer, 2012.

Olga Russakovsky, Yuanqing Lin, Kai Yu, and Li Fei-Fei. Object-centric spatial pooling for image classification. In Proc. of European Conference on Computer Vision, ECCV, pages 1–15, 2012.

Bryan C. Russell, Antonio Torralba, Kevin P. Murphy, and William T. Freeman. Labelme: A database and web-based tool for image annotation. International Journal of Computer Vision, 77(1-3):157–173, 2008.

Fereshteh Sadeghi and Marshall F. Tappen. Latent pyramidal regions for recognizing scenes. In Proc. of European Conference on Computer Vision, ECCV, pages 228–241, 2012.

Jorge Sánchez, Florent Perronnin, and TeóFilo De Campos. Modeling the spatial layout of images beyond spatial pyramids. Pattern Recognition Letters, 33(16):2216–2223, 2012.

Cordelia Schmid, Roger Mohr, and Christian Bauckhage. Evaluation of interest point detectors. International Journal of Computer Vision, 37(2):151–172, 2000. ISSN 0920-5691. doi: 10.1023/A:1008199403446.

Nicola Segata and Enrico Blanzieri. Fast and scalable local kernel machines. Journal of Machine Learning Research, 11:1883–1926, 2010.

Navid Serrano, Andreas E. Savakis, and Jiebo Luo. Improved scene classification using efficient low-level features and semantic cues. Pattern Recognition, 37(9):1773–1784, 2004.

Thomas Serre, Lior Wolf, and Tomaso Poggio. Object recognition with features inspired by visual cortex. In Proc. of International Conference on Computer Vision and Pattern Recognition, CVPR, pages 994–1000, 2005.

Shai Shalev-Shwartz, Yoram Singer, and Nathan Srebro. Pegasos: Primal estimated sub-gradient solver for svm. In Proc. of International Conference on Machine Learning, ICML, pages 807–814, 2007.

Shai Shalev-Shwartz, Yoram Singer, Nathan Srebro, and Andrew Cotter. Pegasos: Primal estimated sub-gradient solver for svm. Mathematical Programming, 127(1):3–30, 2011.

Gaurav Sharma and Frédéric Jurie. Learning discriminative spatial representation for image classification. In Proc. of British Machine Vision Conference, BMVC, pages 1–11, 2011.

Gaurav Sharma, Frédéric Jurie, and Cordelia Schmid. Discriminative spatial saliency for image classification. In Proc. of International Conference on Computer Vision and Pattern Recognition, CVPR, pages 3506–3513, 2012.

John Shawe-Taylor and Nello Cristianini. Kernel Methods for Pattern Analysis. Cambridge University Press, 2004. ISBN 978-0-521-81397-6.

Saurabh Singh, Abhinav Gupta, and Alexei A. Efros. Unsupervised discovery of mid-level discriminative patches. In Proc. of European Conference on Computer Vision, ECCV, pages 73–86, 2012.

Alex Smola, S. V. N. Vishwanathan, and Thomas Hoffman. Kernel methods for missing variables. In Proc. of the Tenth International Workshop on Artificial Intelligence and Statistics, 2005.

Bharath K. Sriperumbudur and Gert R. G. Lanckriet. A proof of convergence of the concave-convex procedure using zangwill's theory. Neural Computation, 24(6):1391–1407, 2012.

Jian Sun and Jean Ponce. Learning discriminative part detectors for image classification and cosegmentation. In Proc. International Conference on Computer Vision, ICCV, pages 3400–3407, 2013.

Martin Szummer and Rosalind W. Picard. Indoor-outdoor image classification. In Proc. of International Workshop of Content-Based Access of Image and Video Database, CAIVD, pages 42–51, 1998.

Radu Timofte, Tinne Tuytelaars, and Luc Van Gool. Naive bayes image classification: beyond nearest neighbors. In Proc. of Asian Conference on Computer Vision, ACCV, pages 689–703. Springer, 2012.

Antonio Torralba, Robert Fergus, and William T. Freeman. 80 million tiny images: A large data set for nonparametric object and scene recognition. IEEE Trans. Pattern Anal. Mach. Intell., 30(11):1958–1970, 2008.

Lorenzo Torresani, Martin Szummer, and Andrew W. Fitzgibbon. Efficient object category recognition using classemes. In Proc. of European Conference on Computer Vision, ECCV, pages 776–789, 2010.

Isaac Triguero, Joaquín Derrac, Salvador García, and Francisco Herrera. A taxonomy and experimental study on prototype generation for nearest neighbor classification. IEEE Transactions on Systems, Man, and Cybernetics, Part C, 42(1):86–100, 2012.

Tinne Tuytelaars, Mario Fritz, Kate Saenko, and Trevor Darrell. The NBNN kernel. In Proc. of Internation Conference on Computer Vision, ICCV, pages 1824–1831, 2011.

Vladimir Vapnik. Principles of risk minimization for learning theory. In Proc. of Neural Information Processing Systems, NIPS, pages 831–838, 1991.

Andrea Vedaldi and Br Fulkerson. VLFeat: An open and portable library of computer vision algorithms. http://www.vlfeat.org/, 2008.

Andrea Vedaldi and Andrew Zisserman. Efficient additive kernels via explicit feature maps. IEEE Trans. Pattern Anal. Mach. Intell., 34(3):480–492, 2012.

Shiv N. Vitaladevuni, Pradeep Natarajan, Shuang Wu, Xiaodan Zhuang, Rohit Prasad, and Premkumar Natarajan. Scene image categorization and video event detection using naive bayes nearest neighbor. In Proc. of Winter Conference on Application of Computer Vision, WACV, pages 140–147. IEEE, 2013.

Julia Vogel and Bernt Schiele. A semantic typicality measure for natural scene categorization. In Proc. of German Association for Pattern Recognition (DAGM) Symposium, pages 195–203, 2004.

Kevin N. Walker, Timothy F. Cootes, and Christopher J. Taylor. Locating salient object features. In Proc. of British Machine Vision Conference, BMVC, pages 1–10, 1998.

Jinjun Wang, Jianchao Yang, Kai Yu, Fengjun Lv, Thomas S. Huang, and Yihong Gong. Locality-constrained linear coding for image classification. In Proc. of International Conference on Computer Vision and Pattern Recognition, CVPR, pages 3360–3367, 2010a.

Xinggang Wang, Baoyuan Wang, Xiang Bai, Wenyu Liu, and Zhuowen Tu. Max-margin multiple-instance dictionary learning. In Proc. of International Conference on Machine Learning, ICML, pages 846–854, 2013.

Yang Wang and Greg Mori. Max-margin hidden conditional random fields for human action recognition. In Proc. of International Conference on Computer Vision and Pattern Recognition, CVPR, pages 872–879, 2009.

Zhengxiang Wang, Yiqun Hu, and Liang-Tien Chia. Image-to-class distance metric learning for image classification. In Proc. of European Conference on Computer Vision, ECCV, pages 706–719. Springer, 2010b.

Zhengxiang Wang, Yiqun Hu, and Liang-Tien Chia. Improved learning of i2c distance and accelerating the neighborhood search for image classification. Pattern Recognition, 44(10): 2384–2394, 2011a.

Zhuang Wang, Nemanja Djuric, Koby Crammer, and Slobodan Vucetic. Trading representability for scalability: adaptive multi-hyperplane machine for nonlinear classification. In Proc. of Conference on Knowledge Discovery and Data Mining, KDD, pages 24–32, 2011b.

Cristopher Williams and Matthias Seeger. Using the Nyström method to speed up kernel machines. In Proc. of Neural Information Processing Systems, NIPS, pages 682–688, 2000.

Robert Andrew Wilson and Frank C Keil. The MIT encyclopedia of the cognitive sciences. MIT press, 2001.

Jeremy M. Wolfe. Visual memory: What do you know about what you saw? Current Biology, 8 (9):R303–R304, 1998. ISSN 0960-9822.

Jianxin Wu and James M. Rehg. CENTRIST: A visual descriptor for scene categorization. IEEE Trans. Pattern Anal. Mach. Intell., 33(8):1489–1501, 2011.

Jianxin Wu, Henrik I. Christensen, and James M. Rehg. Visual place categorization: Problem, dataset, and algorithm. In Proc. of International Conference on Intelligent Robots and Systems, IROS, pages 4763–4770, 2009.

Jianxiong Xiao, James Hays, Krista A. Ehinger, Aude Oliva, and Antonio Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In Proc. of International Conference on Computer Vision and Pattern Recognition, CVPR, pages 3485–3492, 2010.

Lingxi Xie, Jingdong Wang, Baining Guo, Bo Zhang, , and Qi Tian. Orientational Pyramid Matching for Recognizing Indoor Scenes. Proc. of International Conference on Computer Vision and Pattern Recognition, CVPR, 2014.

Jianchao Yang, Kai Yu, Yihong Gong, and Thomas S. Huang. Linear spatial pyramid matching using sparse coding for image classification. In Proc. of International Conference on Computer Vision and Pattern Recognition, CVPR, pages 1794–1801, 2009.

Liu Yang, Rong Jin, Rahul Sukthankar, and Frédéric Jurie. Unifying discriminative visual codebook generation with classifier training for object category recognition. In Proc. of International Conference on Computer Vision and Pattern Recognition, CVPR, 2008.

Tao Yang and Vojislav Kecman. Adaptive local hyperplane classification. Neurocomputing, 71 (13–15):3001–3004, 2008.

Yi Yang and Shawn Newsam. Bag-of-visual-words and spatial extensions for land-use classification. In Proc. of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems, GIS, pages 270–279, 2010.

Bangpeng Yao, Aditya Khosla, and Li Fei-Fei. Combining randomization and discrimination for fine-grained image categorization. In Proc. of International Conference on Computer Vision and Pattern Recognition, CVPR, pages 1577–1584, 2011.

Youngohc Yoon, George Swales Jr., and Thomas M. Margavio. A comparison of discriminant analysis versus artificial neural networks. Journal of the Operational Research Society, pages 51–60, 1993.

Chun-Nam John Yu and Thorsten Joachims. Learning structural svms with latent variables. In Proc. of International Conference on Machine Learning, ICML, page 147, 2009.

Kai Yu and Tong Zhang. Improved local coordinate coding using local tangents. In Proc. of International Conference on Machine Learning, ICML, pages 1215–1222, 2010.

Kai Yu, Tong Zhang, and Yihong Gong. Nonlinear learning using local coordinate coding. In Proc. of Neural Information Processing Systems, NIPS, pages 2223–2231, 2009.

Alan L. Yuille and Anand Rangarajan. The concave-convex procedure. Neural Computation, 15(4):915–936, 2003.

Hao Zhang, Alexander C. Berg, Michael Maire, and Jitendra Malik. SVM-KNN: Discriminative nearest neighbor classification for visual category recognition. In Proc. of International Conference on Computer Vision and Pattern Recognition, CVPR, pages 2126–2136, 2006.

Lei Zhang, Meng Yang, Xiangchu Feng, Yi Ma, and David Zhang. Collaborative representation based classification for face recognition. ACM Computing Research Repository, CoRR, abs/1204.2358, 2012.

Ziming Zhang, Lubor Ladicky, Philip H. S. Torr, and Amir Saffari. Learning anchor planes for classification. In Proc. of Neural Information Processing Systems, NIPS, pages 1611–1619, 2011.

Yingbin Zheng, Yu-Gang Jiang, and Xiangyang Xue. Learning hybrid part filters for scene recognition. In Proc. of European Conference on Computer Vision, ECCV, pages 172–185, 2012.

Li Zhou, Dewen Hu, Zongtan Zhou, and Zhaowen Zhuang. Natural scene recognition using weighted histograms of gradient orientation descriptor. Frontiers of Electrical and Electronic Engineering in China, 6(2):318–327, 2011.

Xi Zhou, Kai Yu, Tong Zhang, and Thomas S. Huang. Image classification using super-vector coding of local image descriptors. In Proc. of European Conference on Computer Vision, ECCV, pages 141–154, 2010.

Xiangxin Zhu, Carl Vondrick, Deva Ramanan, and Charless Fowlkes. Do we need more training data or better models for object detection? In Proc. of British Machine Vision Conference, BMVC, pages 1–11, 2012.

Marco Fornoni

Curriculum Vitae

Idiap Research Institute
Rue Marconi 19
CH-1920 Martigny
Switzerland
marco.fornoni@idiap.ch
http://fornoni.github.io/

Research Interests

My broad interests are in the field of pattern recognition, machine learning and computer vision. My current research is related to automatic classification of images, with a focus on scene recognition, saliency-driven representations and efficient multi-component classification algorithms.

Education

Ph.D. in Electrical Engineering

02/2010 – 09/2014

Automatic visual scene recognition, computer vision, pattern recognition, machine learning

Ecole Polytechnique Fédérale de Lausanne, Lausanne – Switzerland
Idiap Research Institute, Martigny - Switzerland

M.Sc. in Computer Science: 110/110 Magna cum Laude

03/2006 - 10/2009

Machine learning and AI, statistics, theory of computation, information theory, advanced algorithms

Thesis: Multi-view Learning for modeling audio-visual patterns
Supervisor: Prof. Nicolò Cesa-Bianchi
Co-supervisor: Dr. Francesco Orabona

Università degli Studi di Milano, Milan - Italy

10/2001 – 02/2006

B.Sc. in Computer Science: 108/110

Programming, software engineering, databases, operating systems, digital signal processing

Thesis: Reingegnerizzazione ed implementazione del modulo di aggregazione dati di contesto dell'architettura CARE
Supervisor: Prof. Claudio Bettini
Co-supervisor: Dr. Daniele Riboni

Università degli Studi di Milano, Milan - Italy

Work experience

Research Assistant - Idiap Research Institute, Martigny – Switzerland

01/2010 – 06/2014

Design and implementation of multi-component algorithms and saliency-driven representations for automatic visual scene recognition

Technologies: Matlab, C++, BASH, Sun Grid Engine

Research Intern - Toyota Technological Institute, Chicago - USA

07/2013 - 08/2013

Design, implementation and benchmarking of the Multiclass Latent Locally Linear Support Vector Machine classifier

Technologies: Matlab, C++

Research Intern - Idiap Research Institute, Martigny – Switzerland

05/2009 - 09/2009

Analysis and modification of the matrix perceptron algorithm. Using a simple orthogonalization technique the original matricial algorithm was turned into an efficient online 2,p-norm multiple kernel learning algorithm.

Technologies: Matlab

Research Student - DaKWE / EveryWare Lab, University of Milan, Milan – Italy

07/2005 - 01/2006

Thesis: Re-engineering and reimplementation of the contextual data aggregation module and large parts of the CARE architecture. The data aggregation time was reduced by more than one order of magnitude. The communication time between modules was reduced by 75%.

Technologies: Java J2EE, JBoss, JNDI, JDBC, JMS, EJB, Servlet, JavaBean, Web Services, Sockets

---

Publications

Scene Recognition with Naive Bayes Non-linear Learning
Marco Fornoni, Barbara Caputo
In Proc. of the 22nd International Conference on Pattern Recognition , ICPR, 2014 (oral)

Multiclass Latent Locally Linear Support Vector Machines
Marco Fornoni, Barbara Caputo and Francesco Orabona
In JMLR W&CP: Asian Conference on Machine Learning, ACML, 2013 (long oral)

Indoor Scene Recognition using Task and Saliency-driven Feature Pooling
Marco Fornoni and Barbara Caputo
In Proc. of British Machine Vision Conference, BMVC, 2012

OM-2: An Online Multi-class Multi-kernel Learning Algorithm
Jie Luo, Francesco Orabona, Marco Fornoni, Barbara Caputo and Nicolo Cesa-Bianchi
In Proc. of IEEE Online Learning for Computer Vision Workshop, CVPRW, 2010

The Robot Vision Track at ImageCLEF 2010
Andrzej Pronobis, Marco Fornoni, Henrik I. Christensen and Barbara Caputo
In CLEF 2010 LABs and Workshops, Notebook Papers, 2010

A Multi Cue Discriminative Approach to Semantic Place Classification
Marco Fornoni, Jesus Martinez-Gomez and Barbara Caputo
In CLEF 2010 LABs and Workshops, Notebook Papers, 2010

Open Source Software

ML3: implementation of the Multiclass Latent Locally Linear Support Vector Machine algorithm
Software release copyrighted by Idiap Research Institute and available under GPL license
https://www.idiap.ch/scientific-research/resources/ml3

Activities

Teaching Assistant: Cognitive Vision for Cognitive Systems, EPFL PhD course, Fall 2012

Reviewer / Co-reviewer: IEEE Signal Processing Letters, ICRA, ACCV, ECCV , ICCV, BMVC, NIPS, CVPR

Organizer and Participant: ImageCLEF - Robot Vision Challenge 2010

Participant: INRIA Visual Recognition and Machine Learning Summer School 2010

Professional skills

Machine Learning: multiclass Support Vector Machines, latent SVM, locally linear SVM, stochastic gradient descent, multiple kernel learning, online learning, dictionary learning

Computer Vision: scene recognition, object recognition, saliency maps, image descriptors, NBNN

Mathematics & AI: Matlab, Mathematica, Theano

Development & Scripting: Java J2SE / J2EE, C++, BASH, Sun Grid Engine

OS & Tools: Linux, Windows, Virtualbox, Eclipse, SVN, Git

Scientific writing: LaTeX

Languages: Italian (mother tongue), English (fluent C1), French (intermediate B1/B2)

Background and course work in machine learning, computer vision, optimization and statistics