Dissertation

Learning from Internet Images for Object Categorization

Inayatullah Khan

Graz, 30^{th} July 2012

Thesis supervisors

Univ.-Prof. Dr. Horst Bischof

Univ.-Prof. Dr. Peter Auer

Institute for Computer Graphics and Vision

Graz University of Technology

To My father Abdul Wadud, mother Nasreen Wadud,
wife Irfana Khan, and son Abdul Rafay Khan

Statutory Declaration

I declared that I have authored this thesis independently, that I have not used other than the declared sources / resources, and that I have explicitly marked all materials which has been quoted either literally or by context from the used sources.

Place	Date	Signature

Eidesstattliche Erklárung

Ich erkláre an Eides statt, das ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Qullen / Hilfsmittle nicht benutzt, und die den benutzten Quellent wórtlich und inhaltlich entnommene Stellen als solche kenntlich gemacht habe.

Ort	Datum	Unterschrift

Acknowledgments

First of all thanks to Almighty Allah for blessing me with auspicious considerations in my life. After that, I pay my whole gratitude to my supervisor, Prof. Dr. Horst Bischof, for the encouragement, the patient guidance, the support, the advice, and simply everything that he has provided me through out my time as his student. I found my self lucky to have a supervisor who cared so much about my work and who responded to my questions and queries so promptly. Horst gave me the chance to perform research in field of computer vision and the freedom to develop my own ideas.

This thesis finds me indebted to all my present and former colleagues and scientists in ICG, who helped in many ways for completion of my thesis. Special thanks to my teacher, friend and the Boss Dr. Peter M. Roth who helped me a lot in changing my approach to scientific problems and also for coordination in improving my communication skills. His remarkable skills of guiding and motivating a PhD student are appreciated in real manners. He is a great teacher and I would not able to finish my thesis without his guidance. I must thank Dr. Amir Saffari for his guidance, collaborative work and sharing of interesting ideas. Thanks to Dr. Christean Leistner for his valuable discussion and guidance in my research. I am highly obliged and thankful to all my colleagues in the ICG's vision group especially Thomas Mauthner, Martin Godec, Martin Hirzer, Sabine Sternig, Martin Kötinger, Paul Wohlhart, Yunjin Chen, Martin Lenz and last but not least Dr. Michael Donsor.

I want to express my gratitude to Andreas Wurm who provided me the Gracelands with enough computational powers for doing all necessary calculations for my research. Thanks to Renate Hönel, Eva-Maria Christina Fuchs, and Karin Veronika Maier for providing help in office administrative works.

I owe thanks to all my family members: my mother, father, sisters, brother, my parents in law and my wife Irfana Khan, without whose prayers it would have been impossible for me to complete my work. Completing this work would have been all the more difficult were it not for the encouragement provided by my father, Prof.

Abdul Wadud, and my mother, Nusreen Wadud. Special thanks to my life partner who took care of me and our three sons whole heartedly during my whole study period.

I pay thanks to Muhammad Tahir Mushtaq for his company and for giving me an opportunity to work on his research project. I would like to thank Stefan Leskovac, for his company and for helping me in correcting my english writing. Thanks to Nikola Stefanov Prusiyski, Borbála Anna Stuber, Gergely Kovács, Joanna Pereswiet Soltan and Nikoli Na for helping me making things better in free time. The time spent with them will always florish in my memories.

I must also thanks to all my Pakistani friends in Graz and Linz, specially Dr. Mudassar Abbas, Muhammad Jadoon Khan, Farrukh Shahzad, Khurram Shahzad, Rizwan Alam, Dr. Alam zeb, Safdar Zaman, Dr. Tahseen Aslam, Farhan Hyder Sahito, Shujait Ali, Dr. Sarfraz Ahmad and Dr. Muhammad Shafiq Siraj for helping me through their guidance and for confabulation with me in leisure time. I pay my accolades to all my friends, colleagues and people whose names are not mentioned for providing help to improve my knowledge in the field of Computer vision and in life overall.

I am highly obliged for the financial support of Higher Education Commission of Pakistan, which leads to the completion of the task. Thanks to OeAD for their services during my stay. I must thanks to my friends, Fazl-e-Amin from Ireland and Muhammad Jadoon Khan for the financial supports for my stay in Graz for the last couple of months.

Abstract

The Internet is a huge source of images and can thus play a vital role in reducing the laborious manual efforts usually required in learning object categorization models. In this thesis we investigate limitations of existing methods for learning object categorization models directly from the Internet: (a) the ambiguity in true-class labels of the retrieved images, (b) the ambiguity in location of an object, (c) the insufficient representation of image.

We concentrate on support vector machine (SVM), and its extension to semi-supervised manifold learning called Laplacian SVM (LapSVM). Thus, we propose several methods to overcome the problems in training a desired category-level image classifier directly with images collected from the Internet.

Firstly, we present a supervised algorithm which is based on multiple kernel learning. This method combines the textual information describing the objects contained in the retrieved images with their visual features in learning a classification model for image categorization. Experimental evaluations demonstrate that the proposed method can be used to improve the image ranking quality of an image search engine which utilizes only textual information for retrieval of images. Such image ranking mechanism will allow us to collect visually consistent images for any category of objects.

Secondly, we examine a framework of learning from ambiguous examples known as multiple instance learning (MIL). Each example is a bag, consisting any number of instances. We develop an algorithm, called Co-miSVM, which extends LapSVM to MIL domain for learning from multiple-instance examples. This extension enables us to exploit the underlying manifold semantics shared by different visual features and simultaneously copes with the ambiguous examples in training an instance classifier. We apply this method for learning of an object classification model directly

with Internet searched images. Our experimental evaluations demonstrate the competitiveness of the proposed method with respect to state-of-the-art methods.

Thirdly, we propose another method called Huberized LapSVM. Instead of a standard squared Hinge loss function, this method solves the optimization problem of the LapSVM in the primal with a Huber Hinge loss function. The Huber Hinge loss measure the misclassification during training of a classifier and penalizes mildly the noisy labeled training data compared to a squared hinge loss. With this method we can train category-level image classifiers directly with images retrieved by an image search engine. Experimental evaluations on a number of datasets demonstrate the increase in classification performance when training data is contaminated with labeled noise.

Finally, we propose a multi-stage system for training complex part based object detectors with the Internet searched images. The system is necessary to overcome the problem of location ambiguity of an object contained in each of the retrieved images. The system works in three stages and requires only the name of the object class name as input, in the first stage we learn a category-level image classification model for image categorization. Then in the second stage we localize the desired objects contained in those images; finally, in the third stage we trained a supervised part based detector. In the first two stages to deal with ambiguity in the true class label and location of objects, we apply an extended MIL method for the actual learning tasks. With experimental evaluations on test dataset show that we are able to train a reasonable detector without using any manually annotated training examples.

Kurzfassung

Das Internet kann als nahezu unerschöpfliche Quelle von Bildern angesehen werden, durch die der manuelle Aufwand beim Lernen von Modellen zur Objekt-Kategorisierung drastisch reduziert werden kann. In dieser Arbeit wird versucht die Nachteile existierender Verfahren zu umgehen. Dazu gehören unter anderem: (a) die nicht eindeutigen Labels der Bilder, (b) ungenaue Objektpositionen und unzureichende Bildbeschreibungen. Obwohl es unterschiedliche Ansätze gibt, beschränken wir uns in dieser Arbeit auf Support Vector Machines (SVM) und deren semi-supervised Erweiterung Laplacian Support Vector Machines (LapSVM), wo wir unterschiedliche Ansätze präsentieren und bestehende Nachteile umgehen zu können.

Als erstes untersuchen wir eine Methode, die visuelle Information (aus Bildern) mit dazugehöriger Textinformation kombiniert. Die Ergebnisse zeigen, dass auf diese Art und Weise deutlich bessere Ergebnisse erzielt werden können als wenn nur eine der beiden Beschreibungen verwendet wird. Als zweites adressieren wir das Problem von ungenau gelabelten Daten via Multiple Instance Learning (MIL). Dazu führen wir Co-miSVM als neuen Algorithmus ein, der die den Daten zugrunde liegende Manifoldstrukur abbilden kann. Die Methode wird dann im Speziellen um ein Klassifikationsmodell direkt aus Internetbildern zu lernen, wobei im Vergleich zum Stand der Technik sehr gute Resultate erzielt werden können. Für die dritte Methode, Huberized LabSVM, ersetzen wir die standardmäßig verwendete Hinge-Loss Funktion durch eine robustere Formulierung ersetzt, die dann, in diesem Fall, effektiv in der primalen Formulierung gelöst wird. Wiederum können auf diese Art und Weise bessere Klassifikationsergebnisse erzielt werden; selbst wenn von Internetbildern gelernt wurde. Als vierten Beitrag stellen wir ein vor um multi-part Detek-

toren aus Internetbildern zu lernen. Im ersten Schritt wird ein Bild kategorisiert, im zweiten das Objekt lokalisiert und im dritten ein Detektor trainiert. Die Ergebnisse zeigen, dass selbst ohne menschlichen Aufwand in der Praxis einsetzbare Detektoren trainiert werden können.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Object categorization or *object category recognition* is a core problem in various high level computer vision applications. It can be described as the knowledge that a visual object (e.g., any car or any tea cup) present in an arbitrary (digital) image belongs to a previously learned category of objects (e.g., cars or tea cups). It is important to mention that an arbitrary image (depicting a scene) may contain a number of objects belonging to the same or different categories. For classification the objective is to decide the presence or absence of an object of a learned category in an image while for detection the goal is to find the location of the object in addition to its classification. A classifier or detector is learned with a set of training images containing various objects to automatically perform these tasks in future images.

To ensure a good accuracy, a classifier or an object detector must be trained with a large amount of example images; to capture the high variations in visual characteristics of objects within the same category, e.g., shape, appearance, and scale of objects. However in practice, collecting large amounts of representative images and then manually assigning the corresponding labels is a laborious or expensive and tedious work. For classification, an annotator has to provide training sets of images containing instances of objects (e.g., any car and any face) that belong to a certain object category; for detection, additionally the bounding box around an instance has to be specified. The manual labeling and annotation efforts limit the scalability

Figure 1.1: Images collected via Google's image search using the keyword "car fronts". These images can be directly used as positively labeled images for training the "car-front" classifier while negatively labeled images can be obtained from unrelated queries.

of visual learning methods to a few well-defined object categories. This rises the need to explore various sources such as the Internet that enable us automating the collection and labeling of images required for learning object categories.

There is a huge and exponentially growing amount of images available on the Internet. For example, in August 2011 Flickr, one of the major image sharing websites, has reached a milestone of 6 billion images. These images which may depict a scene containing various visual objects, are accompanied with text which describes their contents. Often search engines such as Google, Yahoo and Bing as well photo sharing websites such as Flickr and Zoomr return images using the text information, because of the efficient text-based retrieval techniques developed for a web scale searching. For example, a Google's image search engine returned thousands of images in less than a second using only the keyword "car".

The continuously improving image search facility, provided by most of the search engines, is capable of not only reducing the human labeling efforts but also automating the collection of representative training images. Typing an object category name as a keyword such as "car" or a key phrases such as "car front", "car rear" or "car side" the Google search engine returns images which are mostly visually relevant and the objects are nicely centered in the images. Figure 1.1 shows some of the top ranked "car front" images retrieved by Google image search engine. Similarly, more images can be collected using other image search services via the Internet. This

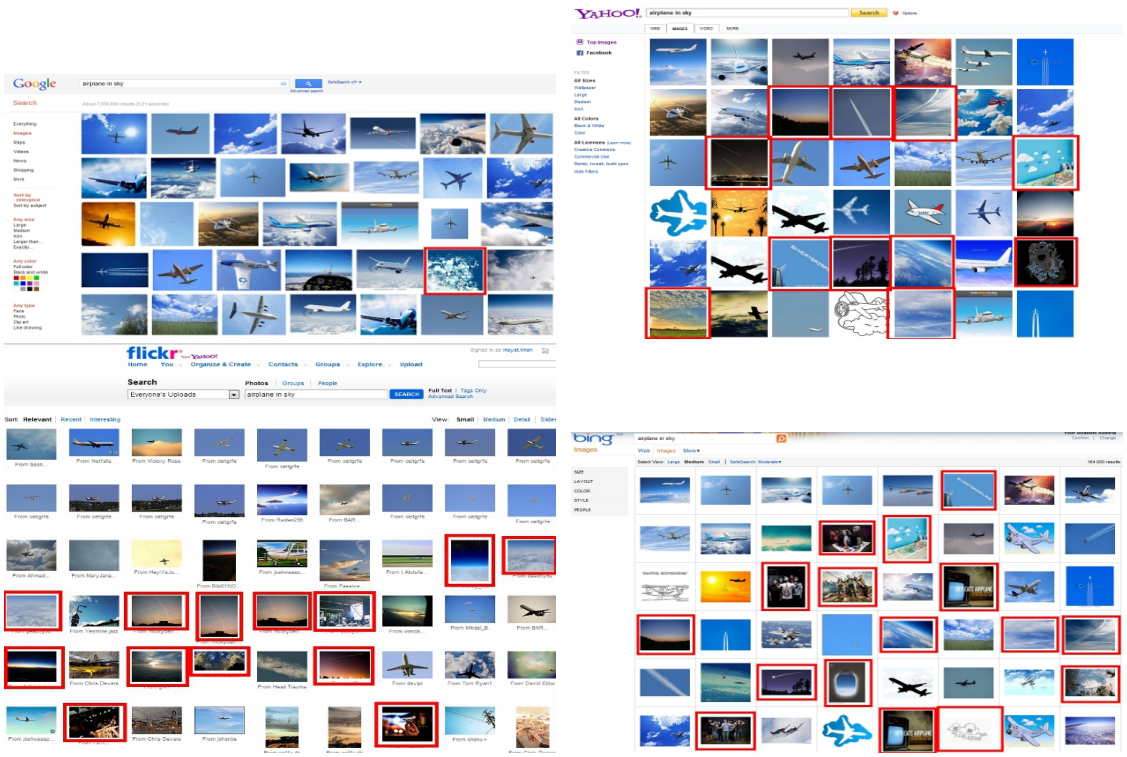


Figure 1.2: Some example of images returned by the four image search engines for an keyword "airplanes"; Top: Google and Yahoo, Bottom: Flickr and Bing. Unrelated or noisy images (not containing "cars") are marked with a red colored box.

way, for any known number of object category names or phrases we can collect a large amount of representative training images. Such images can be used in learning models for the task of object categorization.

In this thesis, the objective is to propose models that enable us to automate visual learning of any object category models directly from the Internet image collection avoiding or reducing the manual labeling efforts. These models can then be used for object category recognition and detection. However, automatic learning from the Internet searched images poses several challenges which we are going to address collectively rather than individually.

First, the true-class-label for each of the returned images could be ambiguous; every image may not contain object(s) which belongs to the desired category. A keyword-based image search engine return images based on the relevance matching between the given keyword and the surrounding text or tags on the web pages containing them, i.e., without using any image contents or visual information. Hence,

Figure 1.3: The top ranked images returned by Google image search for a keyword: "crane". The retrieved images are ambiguous as they belong to different categories, "crane a bird" and "crane a construction equipment".

there is a possibility to collect images which may not be visually consistent for every object category, i.e., the retrieved images may not contain the desired objects. The weaker textual cues are helpful but can not guaranty the presence or absence of a target object in every returned image, e.g., due to polysemous nature of text, i.e., a word or phrase having multiple meanings. This results in ambiguous collections of images which may contain unrelated and noisy or outliered images, even in the top ranked images. In Figure 1.2, we can see noisy images marked by a red colored box which are returned by four different search engines Google, Yahoo, Flickr, and Bing for a keyword "airplanes". Figure 1.3 shows another example explaining polysemous nature of text.

Second, there exist wide varieties in visual characteristics of objects within the same category found in the returned images. Figure 1.4 shows examples for "cars" where we can see the variations in visual characteristics such as shape, appearance, scale and pose. Representation is an important step for learning of an object model. Different types of features such as global (e.g., PHOG [1], BOW, and PHOW [1]) as well local features (e.g., SIFT [2], MSER, and LBP) have been developed that are specialized in capturing certain visual characteristics of an image. For example, shape is modeled using PHOG [1] descriptors, the descriptor is a histogram of oriented or unoriented gradients computed on the output of canny edge detector. Appearance information is modeled using SIFT [2], color or gray level, descriptors

Figure 1.4: Top ranked images returned by Google's image search using the keyword "car". These images can be directly used as positively labeled images for training the "cars" classifier while negatively labeled images can be obtained from unrelated queries. A huge variety in car's visual characteristics such as shape, appearance, scale and number of objects is notable.

which are either computed on a regular grid on the image or on the key-points detected in an image. A good survey on feature extraction and visual object representation can be found in [3]. However, it is difficult to find a single representation that captures most of those visual variations present within class found in the Internet images. Invariance is an important improvement for object category recognitions and object category level image classification, but it is clear that none of the feature descriptor will have the same discriminative power for all object categories. Using different types of features the diverse visual properties found in Internet searched images can be characterized more precisely for discriminating object categories. The challenge is how to combine various types of features during the learning process, since it is difficult to select a universal representation while concatenation may results in curse of dimensionality.

Third, there is an ambiguity in location of the objects in Internet images for arbitrary object category. The direct link between the object the "keyword" and the corresponding location(s) of visual object(s) in an image of the collection is not clear and hard to define automatically. For example, some of the images returned by the Google search engine for the object category "mountain bikes" are shown in Figure 1.5. We can observe that a *"mountain bike"* may appear at any position in an image. Hence, it is challenging to learn a generic object detector model directly

Figure 1.5: Images returned by the Google search engine for the object category "mountain bikes". It is difficult to train an object detector directly using these images where the location of an object in unknown.

from such images which need training images with at least bounding boxes around the objects of interest marking its location.

In this thesis we mainly focus on Support Vector Machines (SVM), a supervised learning approach originally developed by Vapnik and co-workers, for object categorization and detection. It has the ability (soft-margin or hinge loss and regularization) to cope with labeled noise in the training data up to certain extent and additionally it allows us to take benefits of different types of features, which can be combined at the kernel level (e.g., [4, 5]). A combined kernel is defined as a weighted linear combination of individual kernels where the weights are learned with Multiple kernel learning (MKL). MKL [6, 7] aims to optimize the kernel weights while training a classifier. However, for training classifiers with Internet images the supervision is provided by an image search engine which is an imperfect or weaker supervision, since for a given class name we can expect a variable amount of label noise in the collected images.

We can cast the training of classifiers directly with Internet searched images to a Multiple Instance Learning (MIL) problem which deals with the uncertainty in image labels [8, 9]. In contrast to a single instance supervised learning, in MIL the training images are provided to the learner as pairs of bags of instances (images) with labels for the bags. Instance labels remain unknown and must be inferred during learning. A positive labeled bag indicates that at least one instance of that

bag belongs to a positive class and the remaining instances labels are unknown or uncertain. All images in a negatively labeled bag are ensured to be negative and there is no uncertainty for their labels. Now, collecting bags, e.g., bags of images, and specifying the class label for a bag need less manual efforts and is more relaxed then providing class label for each image. For training a category based image classifier such labeled bags of images can be collected easily by using a category name of an object as a keyword via multiple search engines.

1.2 Thesis Contribution

First, we investigate and propose a method to improve the image ranking quality of an image search engine. We use a supervised MKL [10] which provides an elegant way to combine textual and visual information at kernel levels for training a class-level image classifier. Experiemental evaluation demonstrate that the combination of modalities can boost the image ranking of an image search engine that uses only textual information for retrieving images. A useful out-come of this work is a multi-modal dataset called the TVGraz, "Text and Vision Graz". Datasets are essential elements in visual object category recognition research for the performance evaluation of various classification algorithms [11]. Most of training datasets are composed of images without textual modalities. However, TVGraz dataset is a one of the limited number of its kind, containing both images with associated textual description. This dataset is publicly available to research community for benchmarking algorithms developed for categorizing or ranking images by combining textual and visual features. Image ranking by MKL-based classification is an important mechanism to collect visually consistent images for any visual object category. However, being a supervised methods it requires a noiseless or unambiguously true-class-labeled images as training examples.

Next, we propose an approach to directly learn discriminative models for object category recognition from images only. Our approach explicitly acknowledges and simultaneously accounts for their ambiguities (relatedness of the image to a target object category) as well their large variations in visual characteristics. Our main contribution is a novel multiple instance learning algorithm which we call

Co-miSVM (Co-regularized multiple instance SVM). Our method is based on the principle of manifold regularization [12, 13] and extends the supervised multiple instance SVM [14] to semi-supervised MIL domain. This extension enables us to utilize various visual features and to cope with the label noise in training a category-level image classifier. For a given object category name as keyword, we collect groups of images called bags via multiple image search engines. Instead of grouping all of the images or instances returned by a search engine in one huge bag. We encapsulate the top rank few images in a positive bag assuming that at least one of them may contain an object of the desired category (in addition to some irrelevant examples) and treat the rest as unlabeled images or unlabeled singleton bags, while the negative images (not containing the desired category) are obtained executing unrelated queries. The obtained category models can be used for tasks such as re-ranking image search results as well for other object category based image classification.

Since there is an ambiguity in location of the objects in Internet images for arbitrary object category. These images cannot be used directly for training an object detector without manual annotation efforts. To deal with this problem, we propose a multi-stage system that ultimately enables us to train an object detector with images directly obtained from the Internet. The system requires only the name of the target category as input. In the classification stage, we learn an object model that specifies the presence or absence a target object. Then in a localization step, we crop image patches describing the object which are finally used to learn a sophisticated object detector. In the first two stages, we use a multiple instance SVM for training a classifier and a detector. We extend the supervised MIL with a semi-supervised kernel that enable us to learn a better model.

Laplacian Support Vector Machine (LapSVM) [12, 13] has shown state-of-the-art performance in single instance semi-supervised classification problems. To overcome some issues of the original dual formulation, recently Mellacci et al. [15] have extended it and proposed an efficient method for training LapSVM in the primal space, LapSVMp. This reduces the training time when dealing with large number of data and allows for a fast computation with the same classification accuracy as the original one. However, they have used a squared hinge loss for the labeled examples, which may penalize noisy labeled examples too much; especially when dealing with

problems such as learning models from Internet images for object categorization. The accuracy of such an approach may decrease in such situations. We extend LapSVMp to Huberized LapSVMp by using a continuously differentiable Huber hinge loss function. The Huber hinge loss gives a milder penalty than the squared hinge loss. This makes the proposed solution fit in situations when the available labeled data is noisy or it is hard to obtain a clean label training examples.

1.3 Outline

Chapter 2, briefly describes the work related to learning models from Internet images for object categorization and the we present the basic formulations of SVM for single instance learning and multiple instance learning. This is necessary to understand the contribution and the experimental analysis conducted in subsequent chapters. Chapter 3 presents the formulation of the MKL method to combine textual and visual data for category-level image classification or image categorization. These classifiers can then be used to re-rank the images returned by a search engine to improve their visually consistency. The availability of datasets plays an essential role in evaluating a research work. However, there are few datasets available which contains both images and textual information. We present the TVGraz dataset. In Chapter 4, we propose a semi-supervised multiple instance SVMs algorithm, i.e., Co-miSVM. The proposed algorithm allows to utilize different types of visual features and at the same cope with the label noise present in the images collected via multiple keyword-based image searches. We present a direct approach for learning object category recognition models from Internet image searches. In Chapter 5, we present an approach to train supervised part based object detectors from Internet search images through a multi-stage framework which requires only object category names as input.In Chapter 6, we present the detailed formulation for the training of Huberized LapSVM in the primal and shows the benefits of the proposed extension by a number of experiments. Finally, we conclude the thesis in Chapter 7.

Chapter 2

Related Work and Background

The objective of this chapter is twofold. First, we give a review of previous approaches which were adopted by a number of authors to learn object models with images automatically collected from the Internet. Secondly, we discuss the basic concepts and formulations of Support Vector Machines (SVMs) and present some notations that will be used through out this thesis. This will form a basis for understanding our proposed extensions to support vector classifier training algorithms in Chapters 4, 5 and 6 and the corresponding experimental analysis.

Section 2.1 presents an overview for the previous efforts in learning object models with Internet images. Section 2.2, presents the mathematical formulations for training support vector classifiers under a single instance learning paradigm. Section 2.3.1, introduces the training of support vector classifiers under Multiple Instance Learning (MIL) paradigm. In contrast to single instance learning, MIL is a weaker or relaxed form of supervised learning where a learner gets a training set in the form of bag-label pairs where a bag is a collection of instances and the label of a bag is observed for each bag. Here, we give formulations for multiple instance SVM (mi-SVM) proposed in [14] and sparse MIL(sMIL) proposed in [16].

2.1 Related Work

The prospect of learning object models from Internet image search results has started to receive considerable research attention from mid of the last decade. In the fol-

lowing we briefly review of the work done by others who have adopted a variety of techniques for learning object models from noisy search results.

Fergus et al. [17] investigate the problem of learning object categories from Google data. They develop a variant of a clustering method called probabilistic Latent Semantic Analysis (pLSA) [18] to successfully learn object category models from the noisy web data. The interesting component of this work is their method of selecting training data. They pick the top ranked few images returned by a search engine as a training, since the top result tends to be less noisy. To collect even more potential images they submit their query in multiple languages. Later in [19] they further improve their clustering method by including spatial information. Another outcome of their research is the "Fergus dataset" [17]. Such databases are essential elements of object category recognition research for testing the performance of various classification and detection algorithms [11].

Fritz and Schiele [20] present another clustering method for discovery of visual object categories in image collection. Their method is a variant of topic based model called Latent Dirichlet Allocation (LDA), such methods are unsupervised however they usually need selection of dominant proportion of learned topic or cluster in every image. Their approach is best for learning level compact representation of visual categories without labeling of training instances. This representation can then be used to learn object categories in an unsupervised manner. This method is not directly applied for learning object categories from the noisy Internet images. However, their method performance has been evaluated on Fergus [17] dataset providing us an opportunity for performance comparison.

Berg and Forsyth [21] look at the combination of text and image features for learning animals categories. They do pre-clustering using LDA and their supervision is to pick the correct cluster. The images in cluster with their associated text are then used to build classifiers to determine whether an image depicts an animal. They employ 3 types of image features shape based geometric blur features, color features and texture features. They also produce a animals database called "Berg dataset" in this thesis.

Schroff et al. [22] use single instance supervised SVM algorithm to learn object models for specified object category. Due to variable amount of noise in the collected

images they use a multi-staged setup for leaning object category classifier. In the first stage they train an RBF kernel based SVM binary classifier with hand labeled images to discriminate between abstract images such as plots, maps, charts, drawing as well sketches and non abstract images. The classifier is then applied to downloaded images in order to filter such junk images. In the next stage no visual information is used and the images are re-ranked using a Bayes posterior estimator trained on text surrounding the images (the remaining images after removing the junk). The top-ranked images are then finally used as noisy training data and an SVM classifier is learn to improve the ranking further. In the final stage, an image is represented by concatenation of normalized histogram of BOW and HOG with χ^2 distance based RBF kernel.

Guillaumin et al. [23] use text and visual image features for learning object category models from images downloaded from Flickr. First, they train a supervised single instance SVM classifier. MKL is used to combined textual and 15 different image representations (L1 distance for the color histograms, L2 distance for GIST and χ^2 for PHOW). The classifiers are then used to score the unlabeled images. Finally, they train SVM classifiers with visual features using the high scored images as training set. These classifiers are then used to classify unseen images.

Vijayanarasimhan and Graumen [9], present a more direct and automatic method for learning object models. Instead of removing outliers or noisy or unrelated images, they collected bags of images using multiple keyword based image search engines. They directly trained classifiers using those bags of images with a supervised multiple instance SVMs. They extends sMIL, with an iterative refinement scheme that updates the weight of each instance in a bag of images. Such an iterative scheme down weights the outliers or noisy images while training an SVM classifier.

A number of authors have proposed graph-based learning algorithms such as [24], [25], and [26] as well the recent one [27] which are especially developed to re-rank Internet searched images. The basic idea is to exploit the data graph that reflects the intrinsic manifold structure [28] hidden in the images that tends to capture the underlying semantics. In these single instance learning based methods, the higher ranked few images are assumed as weakly labeled images and then conduct

the re-ranking on the whole set of images returned by an keyword-based image search engine. The authors of [27], argued that the underlying manifold structure is beneficial to improve the quality of a search engine only if the outliers are removed. Consequently, they formulate a spectral filter to remove the outliers before using a manifold ranker [28].

Notation

In this thesis we will use the following notation. Vectors are represented in bold notations, e.g., $\mathbf{x} \in \mathbb{R}^d$ and their scalar components in *italic script*, e.g., $x_1, x_2, \cdots, x_d$ where d is the dimension of the vector. Matrices are represented in **bold** fonts, e.g., $\mathbf{Q} \in \mathbb{R}^{m \times n}$ is a matrix of m rows and n columns and the i'th column of this matrix is indicated by $\mathbf{Q}_i \in \mathbb{R}^{m \times 1}$.

2.2 Single Instance Learning

2.2.1 Support Vector Machines

In this section, we briefly sketch the supervised SVM algorithms for solving binary classification problems (c.f. [29–32]). We start our discussion for the simple case of linear and will gradually move to non-linear SVMs.

The earliest recognition system were linear classifiers [33]. Thus, given a set of training examples in the form $\mathcal{L} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{l}$ where each instance $\mathbf{x}_i \in \mathbb{R}^d$, d being the dimension of the input feature space X, belongs to a class labeled by $y_i \in \{+1, -1\}$. The objective of linear SVM is to search for a hyperplane that separates the training examples such that all the points with the same labels remains on the same side of the hyperplane. This is equivalent to finding $\mathbf{w}$ and b so that

$$y_i(\mathbf{w} \cdot \mathbf{x}_i + b) > 0, \tag{2.1}$$

for $i = 1, \cdots, l$. If the given training data is linearly separable there will exist an infinite number of hyperplanes satisfying (2.1). In such a case, it is always possible to rescale $\mathbf{w}$ and b so that the closest point to the hyperplane has the distance of

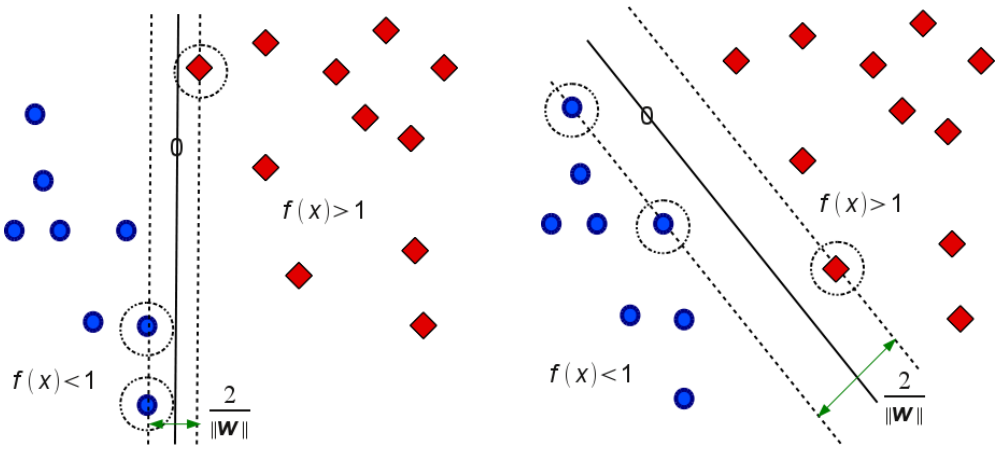


Figure 2.1: Two hyperplanes for linear separation of positive points (red) from negatives points (blue). Left: A separating hyperplane with a small margin. Right: An optimal separating hyperplane with a maximum margin. The points on the margins are called support vectors.

$\frac{1}{\|\mathbf{w}\|}$. Then (2.1) becomes

$$y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1. \tag{2.2}$$

The optimal separating hyperplane is the one for which the distance to the closest point is maximal. A linear SVM, in the primal form, finds an optimal hyperplane by minimizing $\|\mathbf{w}\|^2$ under the constraints (2.2) (c.f. [30] chap. 7). A straight forward calculus defines the margin equivalent to the quantity $\frac{2}{\|\mathbf{w}\|}$ (c.f [30] chap. 7). Figure 2.1 shows examples of two separating hyperplanes, one with a small margin and the other with a widest margin. Among the possible number of separating hyperplanes, the optimal separating hyperplane is the one with maximal margin. The margin can be seen as the measure of generalization ability to previously unseen data, i.e., the larger the margin, the better generalization is expected to be [29, 32]. Minimizing $\|\mathbf{w}\|^2$ under the linear constraints (2.2) is usually done with Lagrangian technique (c.f. [30, 34]), since $\|\mathbf{w}\|^2 \geq 0$ is convex. This approach leads to solving a simpler quadratic dual problem in the positive Lagrangian multipliers $\boldsymbol{\beta} =$

$[\beta_i, \cdots, \beta_l]^T$ which are found by maximizing

$$\begin{aligned} \boldsymbol{\beta}^* = \max_{\boldsymbol{\beta} \in \mathbb{R}^l} & \sum_{i=1}^{l} \beta_i - \frac{1}{2} \sum_{i,j}^{l} y_i \beta_i y_j \beta_j \mathbf{x}_i \cdot \mathbf{x}_j \\ \text{subject to: } & \sum_{i=1}^{l} \beta_i y_i = 0, \\ & \boldsymbol{\beta}_i \geq 0 \quad i = 1, \cdots, l \end{aligned} \tag{2.3}$$

The Lagrangian (Wolf) dual problem (2.3) is easier to solve because of its simpler equality constraints. Various quadratic program (QP) solvers are available for finding an efficient and scalable solution of (2.3), i.e., $\boldsymbol{\beta}^* \in \mathbb{R}^l$ (c.f. [31] chap. [1]). The primal variables or the optimal separating hyperplane $(\mathbf{w}^*, b^*)$ is then computed from $\boldsymbol{\beta}^*$ as

$$\mathbf{w}^* = \sum_{i}^{l} \beta_i^* y_i \mathbf{x}_i. \tag{2.4}$$

Here, it is worth to remind that the support vectors are the training points for which $\beta_i^* > 0$ and satisfying (2.2) with equality. Considering (2.4), we can write the decision function as

$$f(\mathbf{x}) = \text{sgn}\left(\sum_{i=1}^{l} \beta_i^* y_i \mathbf{x}_i \cdot \mathbf{x} \right). \tag{2.5}$$

When the training data is not linearly separable then slack variables $\boldsymbol{\xi} \in \mathbb{R}^l$ are introduced with $\xi_i \geq 0$ [35] such that

$$y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1 - \xi_i, \tag{2.6}$$

to cope with examples violating (2.2). The objective of the slacks is to allow misclassified examples which have their corresponding $\xi_i > 1$. For this reason, $\sum_i \xi_i$ is an upper bound on the training errors. The resulting SVM constrained problem, also called soft margin SVM, is then defined as

$$\min_{\mathbf{w}, \boldsymbol{\xi}} \ \frac{1}{2} \mathbf{w} \cdot \mathbf{w} + C \sum_{i=1}^{l} \xi_i \tag{2.7}$$

subject to constraints (2.6) and $\xi_i > 1$. The first term is minimized to control the learning capacity as in the separable case, discussed earlier, while the second term or the penalty term controls the number of misclassified points [32]. The free parameter C is chosen by hand and controls the trade-off between large margin and small margin violations [36]. For a very large value of C all training examples are correctly classified which causes an overfitting to the training data resulting in a small training error but in a large test error. A smaller value of C produces better results on noisy labeled training examples [30, 35], since this will avoid overfitting to training data already contaminated by label noise. The dual formulation for the problem (2.7) is defined as

$$\begin{aligned} \boldsymbol{\beta}^* = \max_{\boldsymbol{\beta}\in\mathbb{R}^l} \sum_{i=1}^{l} \beta_i - \frac{1}{2}\sum_{i,j}^{l} y_i\beta_i y_j\beta_j \mathbf{x}_i \cdot \mathbf{x}_j \\ \text{subject to: } \sum_{i=1}^{l} \beta_i y_i = 0, \\ 0 \leq \boldsymbol{\beta}_i \leq C \quad i = 1, \cdots, l \end{aligned} \tag{2.8}$$

which is strikingly similar to (2.3). The only change is the upper bound C for the duals co-efficients $\boldsymbol{\beta}$ [36].

In case of a non-linear SVM, the input data is mapped from the original space X into a high dimensional feature space $\mathcal{H}$ through a nonlinear mapping Φ which is chosen a priori [32]. The optimal separating hyperplane is then constructed in this feature space, i.e., a Hilbert space. If we replace $\mathbf{x}$ by its mapping $\Phi(\mathbf{x})$ then (2.8) becomes

$$\begin{aligned} \boldsymbol{\beta}^* = \max_{\boldsymbol{\beta}\in\mathbb{R}^l} \sum_{i=1}^{l} \beta_i - \frac{1}{2}\sum_{i,j}^{l} y_i\beta_i y_j\beta_j \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j) \\ \text{subject to: } \sum_{i=1}^{l} \beta_i y_i = 0, \\ 0 \leq \boldsymbol{\beta}_i \leq C \quad i = 1. \cdots, l \end{aligned} \tag{2.9}$$

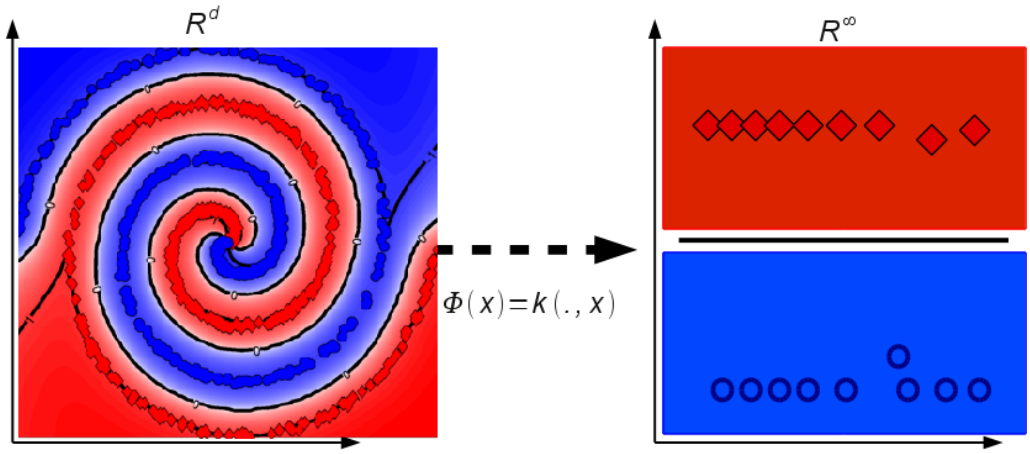


Figure 2.2: The idea of mapping functions: SVM maps a training example into a higher dimensional space via Φ and construct a separating hyperplane with maximum margin. This leads to the basic idea of non-linear decision boundary in the input feature space (shown on the left). The use of the kernel function, $k : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$, makes it possible to compute the optimal hyperplane without explicitly carrying out mapping into a very high dimensional space (shown on the right).

Now, if we define a kernel function as

$$k(\mathbf{x_i}, \mathbf{x_j}) = \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j). \tag{2.10}$$

We need only k in the training algorithm and the mapping $\Phi(\mathbf{x})$ is never explicitly used, since a kernel matrix $\mathbf{K}$ is in hand. In this thesis, a 'kernel' always means a positive definite kernel. The definition of a positive definite kernel function k and a positive definite kernel matrix $\mathbf{K}$ differs as in the former case, we are free to choose the points on which the kernel is evaluated; positive definiteness implies positivity on the diagonal and symmetry of a matrix. Kernels can represent complicated decision boundaries, i.e., nonlinear boundaries directly in the input feature space, that may accommodate any training data whether linearly separable or non-separable. Figure 2.2 shows an example of such a kernel function, this is made possible because of the associated reproducing kernel Hilbert space.

A reproducing kernel Hilbert space $\mathcal{H}$ is a space of functions $X \rightarrow \mathbb{R}$, i.e., a vector space of functions endowed with an inner product such that in the associated norm $\|.\|$, $\mathcal{H}$ forms a complete metric space. The reproducing kernel Hilbert spaces (RKHS) [37] theory (e.g., Mercer theory) precisely states which kernel function cor-

responds to a dot product or an inner product and which linear spaces are implicitly induced by these non linear kernel functions. Once a kernel satisfying the Mercer's theorem [29], has been chosen the SVM training algorithm solves the dual problem (2.9) by replacing the dot product with a kernel as

$$\begin{aligned} \boldsymbol{\beta}^* = \max_{\boldsymbol{\beta} \in \mathbb{R}^l} & \sum_{i=1}^{l} \beta_i - \frac{1}{2} \sum_{i,j}^{l} y_i \beta_i y_j \beta_j k(\mathbf{x}_i, \mathbf{x}_j) \\ \text{subject to: } & \sum_{i=1}^{l} \beta_i y_i = 0, \\ & 0 \leq \boldsymbol{\beta}_i \leq C \quad i = 1, \cdots, l \end{aligned} \tag{2.11}$$

and the decision function becomes

$$f(\mathbf{x}) = \operatorname{sgn}\left(\sum_{i=1}^{l} \beta_i^* y_i k(\mathbf{x}_i, \mathbf{x}) \right). \tag{2.12}$$

With the introduction of kernel functions a standard non linear SVM now requires only the parameter C and a appropriate kernel for the input feature vectors. Several types of kernel exist. For example, a well known Gaussian RBF kernel defined as

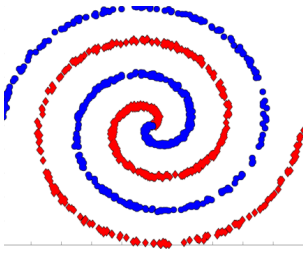
$$k(\mathbf{x_i}, \mathbf{x_j}) = \exp^{\frac{1}{2\tau^2} \|\mathbf{x}_i - \mathbf{x}_j\|_2^2}, \tag{2.13}$$

where τ is the width of the kernel τ usually selected by hand. This kernel can implement any continuous decision boundary. A generalized form of this kernel can be defined as

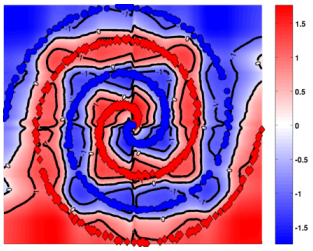
$$k(\mathbf{x_i}, \mathbf{x_j}) = \exp^{\frac{1}{\tau} D(\mathbf{x}_i, \mathbf{x}_j)}, \tag{2.14}$$

where $D(\mathbf{x}_i, \mathbf{x}_j)$ is any distance function in the input feature space. e.g., L_1 distance defined as

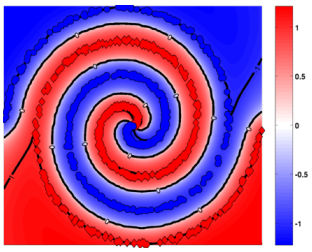
$$D_{L_1}(\mathbf{x}, \mathbf{z}) = \sum_{k=1}^{d} |x_k - z_k|. \tag{2.15}$$



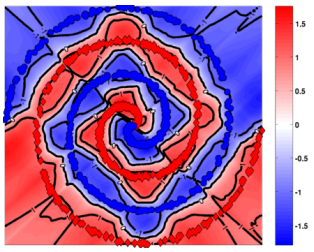
(a) Prob: Find a decision function separating red diamonds (class +1) from blue circles (class -1).



(b) Solution: SVM with L_1 distance based RBF kernel



(c) Solution: SVM with L_2 distance based RBF kernel



(d) Solution: SVM with L_∞ distance based RBF kernel

Figure 2.3: (a) A simple two-dimensional (2D) two spirals supervised classification problem, the spiral with red diamonds belongs to one class and the other spiral with blue circles belong to the other. Solved by 3 supervised support vector algorithms with (b)L_1 (c) L_2 and (d) L_∞ distances based RBF kernels respectively. The L_2 distance based RBF kernel seems to be more appropriate.

The χ^2 distance is defined as

$$D_{\chi^2}(\mathbf{x}, \mathbf{z}) = \frac{1}{2} \sum_{k=1}^{d} \frac{(x_k - z_k)^2}{x_k + z_k}, \tag{2.16}$$

which is more appropriate for histogram comparisons. Figure 2.3 illustrates the use of different distances based RBF kernels for solving a supervised classification problem with a non-linear SVM. Selection of an appropriate kernel for the given input features is important, since an inappropriate kernel may lead to poor performance.

We are not going to discuss here the other benefits of kernels, i.e., a MKL (Chapter 3 will present its detailed formulation), which turns the standard SVM to support vector kernel learning machine inheriting its benefits such as noise tolerance due to soft margin, and extending its applicabilities to learn by combining different

features-based representations [4, 5].

Here, we are going to focus on finding the solution of a non-linear SVM optimization problem by solving it directly in the primal, i.e., without reducing the primal problem (2.7) to a Lagrangian dual form (2.11). The primal and dual are two ways to solve the same optimization problem. In general, there are two main reasons to solve the problem in the dual form. First, the duality provides a simple way to deal with the constraints. Secondly, the dual problem can be written in terms of dot products, thereby making it possible to use kernel functions. The motivation behind the primal is that it directly minimizes the primal variables we are interested in, its implementation does not require any commercial solver or particular complex libraries, and in some cases it may be faster to converge [38]. However, it has been shown that both lead to the same computational complexity [38]. The difference in performance comes when computing an approximate solution, the primal optimization is superior because it is more focused on minimizing what we are interested: the primal objective function [38]. The training of supervised SVM in the primal have been extensively studied and analyzed by a number of authors such as [39] or [38]. In contrast, Mellacci et al. [15] are the only ones who recently explored training LapSVM in the primal. The following discussion will help to understand the next section and ultimately the contribution of Chapter 6.

Following the approach in [38], i.e., considering a non-linear SVM with a kernel function k and the associated RKHS $\mathcal{H}_k$, we can re-write the primal problem (2.7) as

$$\min_{f \in \mathcal{H}_k} \ \gamma_A \|f\|_2^2 + \sum_{i=1}^{l} \ell(y_i, f(\mathbf{x}_i)). \tag{2.17}$$

Which is similar problem but with a little a change in the regularization parameter $\gamma_A = \frac{1}{2C}$ that controls the smoothness the decision surface f measured by the L2 kernel norm (ambient norm) $\|.\|_2^2$. We also replace the slack variables with a differentiable squared Hinge loss function ℓ, defined as $\ell(y,t) = \max(0, 1-yt)^2$, for imposing penalties on violations on the wrong side of the hyperplane. Now using the reproducing property $f(\mathbf{x}_i) = f \cdot k(\mathbf{x}_i, .)$, we can differentiate (2.17) with respect

to f and at the optimal solution f^* the gradient vanishes, yielding

$$\min_{f \in \mathcal{H}_k} \; 2\gamma_A f^* + \sum_{i=1}^{l} \frac{\partial}{\partial t} \ell(y_i, f^*(\mathbf{x}_i)) k(\mathbf{x}_i, .) = 0, \tag{2.18}$$

where $\frac{\partial \ell}{\partial t}$ is the partial derivative of the loss function $\ell(y, t)$ with respect to the second argument. This implies that the optimal function f^* can be written as a linear combination of kernel functions evaluated at the training samples. This result is also known as the Representer theorem [40]. Thus we seek for a final solution, in terms of l co-efficients $\boldsymbol{\alpha}$, of the form

$$f(\mathbf{x}) = \sum_{i=1}^{l} \alpha_i k(\mathbf{x}_i, \mathbf{x}). \tag{2.19}$$

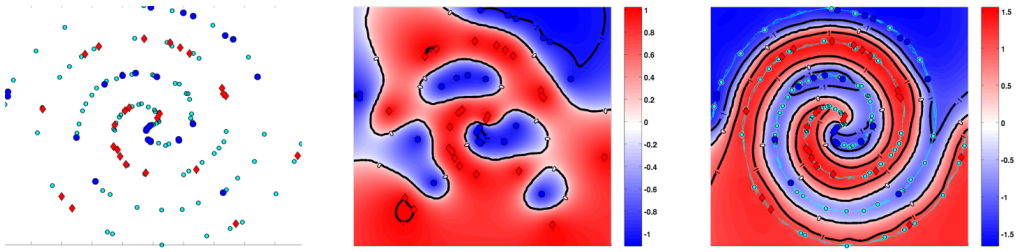
Using the kernel reproducing property in $\|f\|^2$ as

$$\begin{aligned} \|f\|^2 &= \sum_{i,j=1}^{l} \alpha_i \alpha_j \left(k(\mathbf{x}_i, .) \cdot k(\mathbf{x}_j, .) \right) \\ &= \sum_{i,j=1}^{l} \alpha_i \alpha_j k(\mathbf{x}_i, \mathbf{x}_j) \\ &= \sum_{i,j=1}^{l} \alpha_i \alpha_j \mathbf{K}_{ij} \\ &= \boldsymbol{\alpha}^T \mathbf{K} \boldsymbol{\alpha} \end{aligned} \tag{2.20}$$

The primal problem (2.17) in terms of $\boldsymbol{\alpha}$ and $\mathbf{K}$ becomes

$$\min_{\boldsymbol{\alpha} \in \mathbb{R}^l} \frac{1}{l} \sum_{i=1}^{l} \ell(y_i, \mathbf{K}_i^T \boldsymbol{\alpha}) + \gamma_A \boldsymbol{\alpha}^T \mathbf{K} \boldsymbol{\alpha} \tag{2.21}$$

To obtain the solution $\boldsymbol{\alpha}^*$, we can now solve this problem (2.21) as an unconstrained optimization using Newton optimization technique or gradient decent. With the same differentiable loss function, the solutions found by maximizing the dual (2.11), i.e., $\boldsymbol{\beta}^* \in \mathbb{R}^l$, and the solution found minimizing the primal (2.17), i.e., $\boldsymbol{\alpha}^* \in \mathbb{R}^l$ can



(a) (b) (c)

Figure 2.4: (a) The 2D two spirals classification problem. One class of spiral is the one with red diamonds and the other is with blue circles. A large set of unlabeled examples (indicated by cyan colored circles). (b) The supervised SVM solution with L_2 distance based RBF kernel. (c) The semi-supervised LapSVM solution with L_2 distance based RBF kernel and (adjacency matrix $\mathbf{W}$ with $k = 3$). The one dimensional manifold is indicated by cyan colored line.

be seen as the same up to the sign difference:

$$\alpha_i = y_i \beta_i. \tag{2.22}$$

In Chapter 4, we will show that training a single instance supervised SVM classifer gives inferior results when trained on images obtained via search engines. The reason is the variable amount of label noise, since search engine provides imperfect supervision and the weaker textual cues don't guaranty about visual contents in the image.

2.2.2 Laplacian Support Vector Machines

LapSVM, introduced by Belkin et al. [12], is a straight forward extension of the supervised SVM to semi-supervised learning. Semi-supervised learning deals with developing algorithms which can use both labeled samples and unlabeled samples [41, 42]. There are two common assumptions made by most of the semi-supervised learning methods. First is the cluster assumption which states that if two points are likely to have the same class label if they can be connected by curve through a high density region. In other words, if the input feature space exhibits cluster then it is more likely to be from the same class. For example, methods such as Transductive SVM [43, 44] try to maximize the margin of unlabeled samples by pushing away the decision boundary from dense regions of the feature space, i.e.,

if the feature space exhibits clusters then it is more likely that each of these cluster belong to the same class. Second is manifold assumption on which LapSVM is based, which states that the input data lies on low dimensional manifolds in the input feature space and each manifold represent a single class. The LapSVM exploits the geometry of the marginal distribution P_X. If two points that are close with respect to the geodesic distances on a manifold $\mathcal{M}$ then their labels should be the same, similar in the sense that the conditional probability distribution $P(y|\mathbf{x})$ between such two points should change smoothly along the manifold. Figure 2.4, explain the basic idea of the LapSVM in comparison to SVM.

Thus, given a set of training examples in the form $\mathcal{S} = \mathcal{L} + \mathcal{U}$ where the first l instances are labeled and rest u are unlabeled $\mathcal{U} = \{\mathbf{x}_i\}_{i=1}^{u}$; each $\mathbf{x}_i$ is assigned a class label by $y_i \in \{+1, -1, 0\}$, if $y_i \in \{+1, -1\}$ then it is labeled otherwise it is unlabeled. LapSVM learns from the manifold structure of the feature space and the objective function to be minimized is defined as

$$\min_{f \in \mathcal{H}_k} \frac{1}{l} \sum_{i=1}^{l} \ell(y_i, f(\mathbf{x}_i)) + \gamma_A \|f\|^2_{\mathcal{H}_k} + \gamma_I \|f\|^2_I, \tag{2.23}$$

where $\ell(y, t)$ is the linear hinge loss function, γ_A and γ_I are regularization parameters. The former controls the smoothness of the functional f in the associated RKHS (ambient space) and the latter controls its penalization along the low dimensional manifold. Comparing with a standard SVM problem (2.17), in this problem $\|f\|^2_I$ is the additional term that extends SVM to semi-supervised learning. Following the manifold assumption, this term also called intrinsic norm,

$$\begin{aligned} \|f\|^2_I &= \frac{1}{(l+u)^2} \sum_{i,j=1}^{l+u} \mathbf{W}_{ij} (f(\mathbf{x}_i) - f(\mathbf{x}_j))^2 \\ &= \frac{1}{(l+u)^2} \left[f(\mathbf{x}_1), \cdots, f(\mathbf{x}_{l+u}) \right]^T \mathbf{L} \left[f(\mathbf{x}_1), \cdots, f(\mathbf{x}_{l+u}) \right], \end{aligned} \tag{2.24}$$

is approximated on the basis of labeled and unlabeled data using graph Laplacian [12] $\mathbf{L} \in \mathbb{R}^{(l+u)\times(l+u)}$ associated with $\mathcal{S}$. The graph Laplacian is a positive semi-definite operator on functions defined over the vertexes of an adjacency graph $\mathbf{W} \in \mathbb{R}^{(l+u)\times(l+u)}$ and is given by $\mathbf{L} = \mathbf{D}^{-\frac{1}{2}} \mathbf{W} \mathbf{D}^{-\frac{1}{2}}$, where $\mathbf{D}$ is a diagonal matrix

given by $\mathbf{D}_{ii} = \sum_{j=1}^{l+u} \mathbf{W}_{ij}$.

According to the extended version of the Representer Theorem (for proof, c.f.[12]), the minimizer of the problem in (2.23) is unique and is equal to the linear combination of kernel evaluation centered on the training data points $\mathcal{S}$:

$$f(\mathbf{x}) = \sum_{i=1}^{l+u} \alpha_i^* k(\mathbf{x}_i, \mathbf{x}). \tag{2.25}$$

If the training data $\mathcal{S}$ is ordered such that the first l points are labeled and next u are unlabeled points, then the LapSVM problem (2.23) can be equivalently written as

$$\boldsymbol{\alpha}^* = \min_{\boldsymbol{\alpha} \in \mathbb{R}^{(l+u)}} \frac{1}{l} \sum_{i=1}^{l} \ell(y_i, \mathbf{K}_i^T \boldsymbol{\alpha}) + \gamma_A \boldsymbol{\alpha}^T \mathbf{K} \boldsymbol{\alpha} + \frac{\gamma_I}{(l+u)^2} \boldsymbol{\alpha}^T \mathbf{K} \mathbf{L} \mathbf{K} \boldsymbol{\alpha} \tag{2.26}$$

in terms of a kernel matrix $\mathbf{K} \in \mathbb{R}^{(l+u)\times(l+u)}$ and $\boldsymbol{\alpha} \in \mathbb{R}^{(l+u)}$. The final decision function becomes

$$f^*(\mathbf{x}) = \operatorname{sgn}\left(\sum_{i=1}^{l} \alpha_i^* k(\mathbf{x}_i, \mathbf{x}) \right). \tag{2.27}$$

In the original original work of Belkin et al. [12], the proposed solution for LapSVM is based on the dual form derived from (2.26), using standard Lagrange Multiplier techniques in a similar way used for the supervised SVM [29]. The training of LapSVM in the dual involves two steps. First using a QP solver, we solve the quadratic program in l dual variables $\boldsymbol{\beta}$:

$$\begin{aligned} \boldsymbol{\beta}^* = \max_{\boldsymbol{\beta} \in \mathbb{R}^l} & \sum_{i=1}^{l} \beta_i - \frac{1}{2} \boldsymbol{\beta}^T \mathbf{Q} \boldsymbol{\beta} \\ \text{subject to: } & \sum_{i=1}^{l} \beta_i y_i = 0, \\ & 0 \le \boldsymbol{\beta}_i \le \frac{1}{l} \quad i = 1, \cdots, l \quad , \end{aligned} \tag{2.28}$$

where

$$\mathbf{Q} = \mathbf{YJK}(2\gamma_A\mathbf{I} + 2\frac{\gamma_I}{(l+u)^2}\mathbf{LK})^{-1}\mathbf{J}^T\mathbf{Y}; \tag{2.29}$$

$\mathbf{J} \in \mathbb{R}^{l\times(l+u)}$ is defined as $\mathbf{J} = [\mathbf{I} \quad \mathbf{0}]$, where $\mathbf{I} \in \mathbb{R}^{l\times l}$ is identity matrix (assuming that the first l points are labeled) and a zero matrix $\mathbf{0} \in \mathbb{R}^{l\times u}$ and $\mathbf{Y} \in \mathbb{R}^{l\times l}$ is a diagonal matrix with $\mathbf{Y}_{ii} = y_i$. In the second step, the $(l+u)$ expansion coefficients $\boldsymbol{\alpha}^*$ that define the decision function f^* in (2.27) are obtained by solving the linear system involving l dual variables $\boldsymbol{\beta}^*$:

$$\boldsymbol{\alpha}^* = (2\gamma_A\mathbf{I} + 2\frac{\gamma_I}{(l+u)^2}\mathbf{LK})^{-1}\mathbf{J}^T\mathbf{Y}\boldsymbol{\beta}^*. \tag{2.30}$$

The relation between the expansion co-efficients $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ is no longer simple as that supervised SVM algorithm (2.21). When $\gamma_I = 0$, the LapSVM ignores the unlabeled data, i.e., the equations (2.28) and (2.30) gives zero expansion co-efficients over the unlabeled data and the final solution becomes similar to that of a supervised SVM.

Several other graph-based semi-supervised learning algorithms have been introduced, e.g., Mincut [45, 46] and local and global consistency [47]. The intuition is that the target classification function should be smooth on the graph or equivalently regularized by the graph. These algorithms are mostly different in the choice of loss functions and the regularizers that are constructed from the training data (for a tutorial overview c.f. [48]) . However, most of them perform transductive inference, i.e., they can only classify the unlabeled data given in training. In contrast, manifold regularization based methods such as Laplacian SVMs are inductive semi-supervised learning algorithms providing out-of-sample extension.

Recently, Melacci et al. [15] have proposed a fast training algorithm for solving the LapSVM problem (2.26) in the primal, using squared hinge loss function. In this thesis, we will will refer it LapSVMp. Specifically for solving the LapSVM (2.26), it has been shown in [15] that the unconstrained primal optimization is more efficient than the dual. Training LapSVM directly in primal allows us to solve the original problem without the need of the computations related to variables switching, i.e., it directly manipulates the primal kernel expansion co-efficients $\boldsymbol{\alpha}$ without passing through the dual $\boldsymbol{\beta}$ ones. It avoids the computations overhead related to variable

switchings, the matrix inversions in (2.29) and (2.30). The computational cost is reduced from $O((l+u)^3)$ to $O(k(l+u)^2)$ where k is significantly smaller than $(l+u)$ [15]. However, if the available training data $\mathcal{L}$ is noisy (images returned by a search engine) then the squared hinge loss may penalize the outliers too much resulting in lower final classification accuracy. In Chapter 6, we will present our proposed algorithm extending LapSVMp with a new a loss function that is more robust to noise in the labels.

The co-regularized LapSVM algorithm (Co-LapSVM) proposed by Sindhwani et al. [13] extends LapSVM by making it applicable to solve classification problems when each instance $\mathbf{x}_i$ can be observed in two of more types of features. For example, in object classification problem where an object could be represented by different or diverse features and every representation is specialized in capturing a certain aspect (e.g., shape, appearance) of an object. Co-LapSVM assumes that graph Laplacian in one representation induces complementary notion of similarity in other representation. This exchange of information across V different representations is made available through a combined graph Laplacian $\mathbf{L}_C$; which is build by taking the mean of individual graph Laplacian constructed with each type of input features, i.e., $\mathbf{L}_C = \sum_{v=1}^{V} \mathbf{L}^{(v)}$.

The final binary decision function of Co-LapSVM is of the following form,

$$\begin{aligned} f_{Co-LapSVM}(\mathbf{x}) &= sgn\left(\frac{1}{V}(f^{*(1)}(\mathbf{x}) + \cdots + f^{*(V)}(\mathbf{x}))\right), \\ &= sgn\left(\frac{1}{V}\left(\sum_{i=1}^{l+u} \alpha_i^{*(1)} k^{(1)}(\mathbf{x}_i, \mathbf{x}) + \cdots + \sum_{i=1}^{l+u} \alpha_i^{*(V)} k^{(V)}(\mathbf{x}_i, \mathbf{x})\right)\right), \\ &= sgn\left(\frac{1}{V}\sum_{v=1}^{V} \mathbf{K}^{(v)}(\mathbf{x})^T \boldsymbol{\alpha}^{*(v)}\right), \end{aligned} \tag{2.31}$$

The co-efficients $\boldsymbol{\alpha}^{*(v)}$ of the corresponding kernel $k^{(v)}$ expansion are obtained either by minimizing the following co-regularized functional :

$$\min_{\boldsymbol{\alpha}^{(v)} \in \mathbb{R}^{(l+u)}} \frac{1}{l}\sum_{i=1}^{l} \ell(y_i, \mathbf{K}_i^{(v)^T} \boldsymbol{\alpha}^{(v)}) + \gamma_A^{(v)} \boldsymbol{\alpha}^{(v)^T} \mathbf{K}^{(v)} \boldsymbol{\alpha}^{(v)} + \frac{\gamma_I^{(v)}}{(l+u)^2} \boldsymbol{\alpha}^{(v)^T} \mathbf{K}^{(v)} \mathbf{L}_C \mathbf{K}^{(v)} \boldsymbol{\alpha}^{(v)} \tag{2.32}$$

which is similar to training LapSVM with combined graph Laplacian $\mathbf{L}_C$ or by

training a supervised SVM with the following kernel function :

$$\tilde{k}^{(v)}(\mathbf{x}_i, \mathbf{x}_j) = k^{(v)}(\mathbf{x}_i, \mathbf{x}_j) - \mathbf{K}_i^{(v)^T}(\mathbf{I} + \frac{\gamma_I^{(v)}}{\gamma_A^{(v)}}\mathbf{L}_C\mathbf{K}^{(v)})^{-1}\frac{\gamma_I^{(v)}}{\gamma_A^{(v)}}\mathbf{L}_C\mathbf{K}_j^{(v)^T}. \tag{2.33}$$

See [13] for the derivation of this kernel function $\tilde{k}$. We will present two algorithms that extends Co-LapSVM to solve multiple instance learning problem, i.e., in Chapter 4 by using (2.32) and in Chapter 5 by using (2.33).

2.3 Multiple Instance Learning

In this section we first formally define MIL problem [8] which is a weaker or relaxed form of a supervised learning and then we review two SVM based algorithms introduced to MIL. The first one is multiple instance SVM (mi-SVM) introduced by Andrew et al. [14] and the second one called sparse MIL (sMIL) [16] which further extends miSVM.

In the MIL paradigm a learner gets training set $\mathcal{L}_B = \{(\mathcal{B}_i, Y_i)\}_{i=1}^{l_b}$ in the form of l_b bag-label pairs where a bag $\mathcal{B}_i = \{\mathbf{x}_{ij}\}_{j=1}^{n_i}$ is a collection of n_i instances and the label $Y_i \in \{-1, +1\}$ is observed for each bag.

The bag label provides information about an instance $\mathbf{x}_{ij}$ label y_{ij} in an asymmetric way which can be explained by the following statements:

$$Y_i = +1 \Rightarrow \qquad \exists\ \mathbf{x}_{ij_o},\ j_o \in \{1, \cdots, n_i\} : y_{ij_o} = +1, \tag{2.34}$$

$$Y_i = -1 \Rightarrow \qquad \forall\ \mathbf{x}_{ij_o},\ j_o\ \in \{1, \cdots, n_i\} : y_{ij_o} = -1, \tag{2.35}$$

A positively labeled bag ensures that it contains at least one instance that can be assigned a positive label; there is no other information about the remaining instances in a positive labeled bag which might be completely unrelated belonging to neither positive nor negative class. A negative labeled bag ensures that all instances it contains are assigned negative labels. The ultimate goal is to learn an instance classifiers $f_* : \mathcal{X} \to \{-1, +1\}$.

2.3.1 Multiple Instance SVMs

Andrew et al. [14] solve the MIL problem with support vector machine, the resulting algorithm is called mi-SVM. Since the total number and the true labels of instances in positive bags are not known a-priori therefore the objective is to optimize over unobserved discrete variables y_{ij} jointly with the SVM parameters. The objective function of mi-SVM is defined as

$$\min_{f \in \mathcal{H}_k, \{y_{ij}\}} \gamma_A \|f\|^2_{\mathcal{H}_k} + \sum_{i=1}^{l_b} \sum_{j=1}^{n_i} \ell(y_{ij}, f(\mathbf{x}_{ij})) \tag{2.36}$$

subject to:

$$\begin{aligned} y_{ij} \in \{-1, +1\}, \forall\ i \qquad &= 1, \cdots, l_b,\ j = 1, \cdots, n_i, \\ y_{ij} = -1, \forall\ i : Y_i \qquad &= -1,\ j = 1, \cdots, n_i, \\ \sum_{j=1}^{n_i} y_{ij} \geq 2 - n_i, \forall\ i : Y_i \qquad &= 1, \end{aligned} \tag{2.37}$$

where $\gamma_A = \frac{1}{2C}$ is a regularization parameter. The constraint (2.37) enforces that at least one instance from each positive bag should be positive.

Due to the presence of unobserved discrete variables the objective function of mi-SVM (2.36) is an integer program that leads to an iterative optimization which is applied in the following way. After starting with initializing the instance labels by their bag labels, the following two steps are alternated until convergence, i.e. until the labeling of instances in positive bags does not change any more. Using the current assignment in iteration t, an SVM is trained and the resulting classifier f_* is used to assign labels to all instances thus determine their new values. If necessary, the constraint (2.37) is enforced by switching the label of the least negative instance in a positive bag for it is violated. This the optimization algorithm has proven to converge to local minima [14].

Bunesco et al. [16] have pointed out that mi-SVM is solely based on initializing instances labels with the bag labels. Consequently, mi-SVM may overestimate the actual number of true positive instances in a bag and may fail when positively

labeled bags are sparse in true positive instances. To remedy this, they represent a positive labeled bag by the mean of its component instances' representation and consider negative bags as singleton bags (i.e., $\forall\ Y_i = -1,\ n_i = 1$). The resulting objective function to be minimized is defined as

$$\min_{f \in \mathcal{H}_k} \gamma_A \|f\|^2_{\mathcal{H}_k} + \sum_{i=1}^{l_b} \acute{\ell}(Y_i, \sum_{j=1}^{n_i} \omega_{ij} f(\mathbf{x}_{ij})) \tag{2.38}$$

where the loss function is $\acute{\ell} = \max(0, \frac{2-n_i}{n_i} - Y_i \sum_{j=1}^{n_i} \omega_{ij} f(\mathbf{x}_{ij}))$ and $\omega_{ij} = \frac{1}{ni}$. It is important to note that the objective function is convex and can be solved directly with standard SVM solver such as QP solver; however the dual variable are now multiplied by $-1 + \frac{2}{n_i}$ instead of $+1$ when the positive bag size n_i is greater than 1 [16].

Comparing to mi-SVM[14], in this approach the positive bags are squashed into a single instance by taking their mean. This squashing or treating every instance in a positive bag uniformly may be harmful in situations such as learning from the Internet image searches where we can expect completely unrelated or junk images, neither positive nor negative, in positive bags. The authors in [9] extends sMIL by proposing an iterative refinement scheme. Whereas in each iteration they learn a new classifier $f_*(x)$ by solving (2.38) and update the weights associated with each instance in a positive bag as

$$\omega_{ij}^{(t)} = \frac{\omega_{ij}^{(t-1)} \exp^{\frac{(f_*(\mathbf{x}_{ij}) \ - \ \arg\max_{\mathbf{x}_{ij} \in B_i^+} f_*(\mathbf{x}_{ij}))}{c(\arg\max_{\mathbf{x}_{ij} \in B_i} f_*(\mathbf{x}_{ij}) \ - \arg\min_{\mathbf{x}_{ij} \in B_i^+} f_*(\mathbf{x}_{ij}))}}}{\sum_{k=1}^{n_i} \omega_{ik}^{(t)}}, \tag{2.39}$$

while the weights are initialized as $\omega_{ij}^{(0)} = \frac{1}{n_i}$ and constant c is a hyper parameter which is set using cross-validation approach following [17].

Chapter 3

Image Categorization by Combining Textual and Visual features

This chapter presents a simple approach for categorizing images by combining visual and textual features. Such method can bring further improvement in the ranking performance of an image search engine. The goal of image categorization task is to decide whether an image containing objects belong to certain desired category or not. This task is performed by a keyword based image search engine, however using only textual information. Given an object class name as a keyword, an image search engine retrieves and rank images. This is task is performed based on the relevance matching between the given keyword and the surrounding text or tags on the web pages containing those images, i.e., without using any visual information. However, due to polysemous nature of text, e.g., a text word having multiple meaning such as "butterfly" could refer to "butterfly valve" or "butterfly shaped fish", it is difficult to retrieve images containing objects which belongs to the desired or target category. This result in ambiguous collections of images; containing unrelated or noisy images even at the higher ranked positions. This chapter investigates this problem by learning classifiers by combining both modalities, i.e., the visual features of images and the text describing their visual contents. We use a supervised MKL [10] which provides an elegant way to combine textual and visual information at kernel levels

for training a class-level image classifier. This classifier is then used to categorized and rank future images. With Experimental evaluation, we demonstrate that the proposed method could be adapted by a text-based image search engine for its further improvement in image ranking quality.

The availability of datasets plays an essential role in evaluating a research. However, there are few datasets available which contain both images and corresponding textual information. We created the TVGraz dataset containing both object images with associated textual description. The dataset is publicly available to research community for benchmarking algorithms which are developed for categorizing images or ranking images by combining textual and visual features.

This chapter is organized as follow: Section 3.1 presents the formulation for MKL. Section 3.2 provides the detailed procedure adapted for creating the TVGraz dataset and discuss its special features. Section 3.3 describes the textual and visual features extraction and their representation by kernels for training classifiers by Multiple Kernel Learning method. Section 3.4, presents the experimental evaluation of the propose method and finally section 3.5 concludes the chapter.

3.1 Multiple Kernel Learning

The supervised Multiple Kernel Learning (MKL) method was originally proposed in [10] and later extended and applied by a number of people (e.g., [1, 4, 6, 7, 36]) to various tasks. MKL provides a natural way to combine and learn from different modalities, i.e., an image represented by its textual and visual features. The basic idea behind MKL is to create a weighted linear combination of kernels from each information source and to adapt these weights in order to achieve improvement in classification performance. Given a labeled training set $\mathcal{L}$, the objective is to optimize jointly over the coefficients $\boldsymbol{\alpha} \in \mathbb{R}^l$ of an SVM and a weighted linear combination of kernels $k^\star(\mathbf{x}_i, \mathbf{x}_j) = \sum_{v=1}^{V} w_v k^{(v)}(\mathbf{x}_i, \mathbf{x}_j)$ in a single optimization problem.

The scope of this chapter is the applicability of MKL method to the task of image classification and using the resulting learned classifiers for image ranking, rather than its optimization. In general and for comparison with the proposed

formulation in Chapter 4, we can write the MKL primal objective function as

$$\min_{\boldsymbol{\alpha}\in\mathbb{R}^l, w} \frac{1}{2}\sum_{v=1}^{V} w_v \boldsymbol{\alpha}^T \mathbf{K}^{(v)} \boldsymbol{\alpha} + C\sum_{i=1}^{l} \ell(y_i, \sum_{v=1}^{V} w_v \mathbf{K}_i^{(v)T} \boldsymbol{\alpha}) \tag{3.1}$$

$$\text{subject to: } \sum_{v=1}^{V} w_v = 1, w_v \geq 0, v = 1..V, \tag{3.2}$$

where $\ell(y_i, t) = max(0, 1 - y_i t)$ is the linear hinge loss function; $\mathbf{K}^{(v)} \in R^{l\times l}$ is the kernel matrix and w_v is the weight for the v'th feature type*; C is the only free constant regularization parameter on the loss function and norm-1 constraint (3.2) on the weights for kernel combination which may result in sparse selection of a kernel.

For solving this MKL problem, i.e., finding the kernel expansion coefficients $\boldsymbol{\alpha}$ and weights $\mathbf{w}_v \in \mathbb{R}^v$, different algorithms have been proposed such as [50] and SimpleMKL [6], which mainly differ in their runtime performance. Here, we use the SimpleMKL [6] † method, which reduce this problem to Lagrangian dual and solve for the dual variables. This method iteratively determines the combination of kernels by a gradient descent wrapping a standard SVM solver (c.f. [6], for SimpleMKL algorithm).

The final binary decision function of MKL is of the following form:

$$F_{MKL}(\mathbf{x}) = sgn\left(\sum_{v=1}^{V} w_v^* \mathbf{K}^{(v)}(\mathbf{x})^T \boldsymbol{\alpha}^*\right). \tag{3.3}$$

For training a classifier with MKL method, we use the standard bag of word models to represent an image using textual features and visual features (SIFT descriptors); the details of features extraction and the corresponding kernels are given in Section 3.3. The convex combination of the two kernels, one per modality, can be seen as

$$\mathbf{K}_{ij}^{\star} = k^{\star}(\mathbf{x}_i, \mathbf{x}_j) = w_{(txt)} k^{(txt)}(\mathbf{x}_i, \mathbf{x}_j) + w_{(vis)} k^{(vis)}(\mathbf{x}_i, \mathbf{x}_j), \tag{3.4}$$

*In our published work [49] the kernel weights are represented by d_m.

†The code is available at `http://asi.insa-rouen.fr/enseignants/~arakotom/code/mklindex.html`.

Nr.	Category	No. positive images	No. negative images
1	brain	209	107
2	butterfly	305	131
3	cactus	217	116
4	deer	324	140
5	dice	272	142
6	dolphin	272	127
7	elephant	223	111
8	frog	333	189
9	harp	230	250
10	pram	207	125

Table 3.1: Summarizes the TVGraz dataset objects images. A positive image contains at least one object of the given category.

where the subscripts (vis) and (txt) indicates the visual and textual component, respectively.

3.2 TVGraz Dataset

We created a dataset, named TVGraz[‡] containing a total of 4030 labeled images and the associated web pages. The dataset contains a total of 10 objects categories which are listed in Table 3.1. The objective of the multi-modal dataset is to provide a common means for evaluation of object categorization research based on text and vision. The different categories are selected from Caltech-256 [51] dataset. Using each object as a keyword, we tried to download the top 1000 web pages and the images retrieved by the Google image search engine. In each collection, we captured the image, the image file name, the web page containing the image and the image URL. We filter those images which are not accessible directly from their respective original URLs (because either the links do not exist or the website is protected). We also remove empty images, images with missing text data, painting images, and line sketched images. The images are provided in their actual form as downloaded from the Internet in order to provide an unbiased and challenging dataset.

For eight categories named human brain, cactus plant, deer animal, elephant

[‡]TVGraz dataset is available at http://www.icg.tugraz.at/Members/kahn/tvgraz .

animal, dice a small cube with each side having different number of spots, pram, dolphin fish, and musical instrument harp, an image is labeled as positive if it contains at least one instance of the corresponding category; otherwise it is labeled as negative (not containing the corresponding objects of the corresponding category). The frog animal and the butterfly insect are difficult categories due to the polygamous nature of the word "frog". We explicitly select these categories as we think that this could be easily solved by combining both text and visual features. The images returned by the search engine for the keyword "frog", contained other visual objects like frog shaped computer mouse, frog shaped puppets, frog an amphibian, frog a cartoon character, and many more. We labeled an image as positive if it contains at least a single object of the amphibian frog. Similarly, for butterfly insect we label the image as positive, if it contains at least a single instance of the butterfly insect; otherwise it is labeled as negative. Figure 3.1, shows examples images for each category.

In addition to the original raw textual data and in order to provide an easy start-up for researchers in the vision community, we also provide a pre-processed form of the textual part of the database. In this respect, it has been shown that the surrounding texts of an image on a web page usually has an important connection to the semantic contents of the image. However, it is hard to clearly define the exact relevant text close to the image on a web page because of the rich content of the web page. A web page may contain various texts in surrounding of an image such as navigation, advertisement and contact information, which are neither related to the image nor to the topic of the web page. However, there exist some methods, which try to provide an automated solution for this problem, such as window-based approaches [52] or VIsion-based Page Segmentation (VIPS) [53].

A window-based approach uses a fixed length window to extract the text before and behind an image by treating the HTML source as a text stream [52]. This method is fast but might not be accurate because of the web page's structure discussed above. It is also difficult to define a fixed length window for every web page. The VIPS [53] method extracts the text close to the image on a web page by analyzing the tree structure of the web page based on its visual presentation [52, 53]. Each node in VIPS tree corresponds to a block and the segments obtained are more

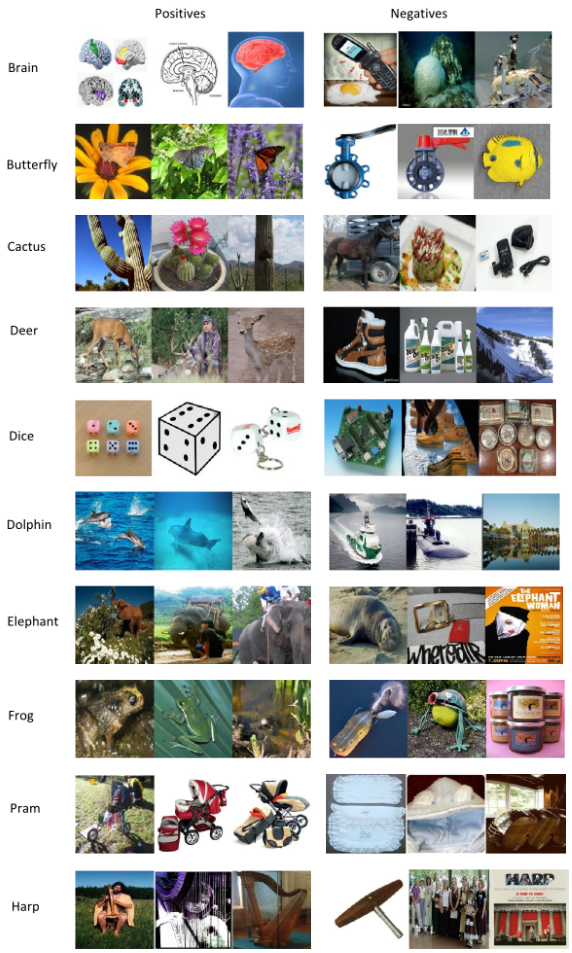


Figure 3.1: Example images per category in TVGraz dataset: left column shows positive images (containing objects of the corresponding class) and right column shows negative images (not containing objects of the corresponding class).

semantically aggregated. Each node is assigned a value indicating the coherency of the contents in the block based on its visual perception. Thus, it is easy to extract the text close to the image by locating the block containing the image. An example of a web page segmented by using VIPS is shown in Figure 3.2.

Therefore, we use VIPS for extraction of the text close to the image and additionally provide these as the pre-processed textual part of the dataset. For each image in the dataset, we use the image file name to locate the visual block containing the image after applying VIPS to the corresponding web page.

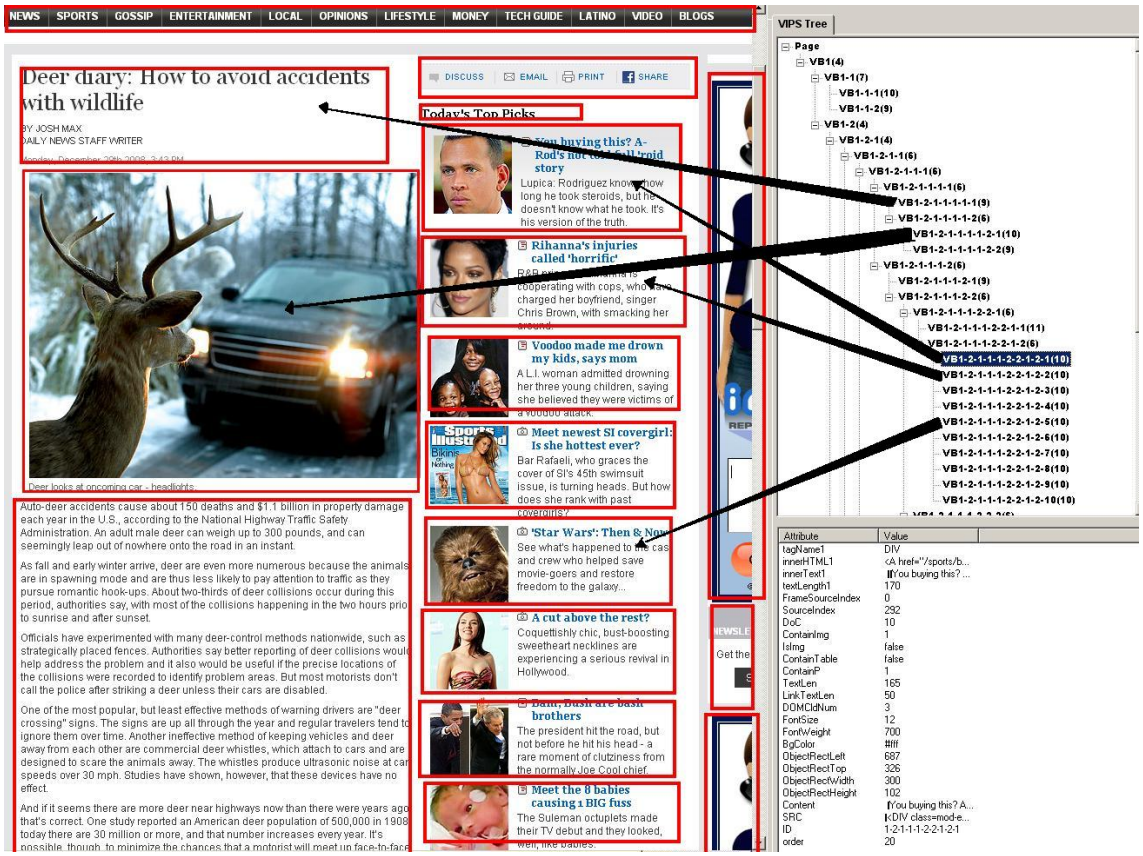


Figure 3.2: An example that shows VIPS based web page segmentation. The boxes show the visual blocks after segmentation. The text of the corresponding visual block may be used to describe the images contents

3.3 Features extraction and Representation

In order to learn classifiers by combining textual and visual modalities, we need a proper representation of both features. The following subsections, provide the details for features extraction and their representation by kernels.

3.3.1 Textual Features

The BOW model is a popular method for a document representation. It is a vector based representation of a document. The size of the vector where each bin counts the number of words occurring in it. In general, the text document is first parsed into words. Then a stop-list is used to remove insignificant words like 'with', 'wonder', 'the', and 'you'. Finally, words are represented by their stem or root by applying a

stemming process, for example 'wait', 'waits','waited', and 'waiting' are represented by the root 'wait'. A unique identifier is then assigned to the remaining words and each text document is then represented by vector with components showing the counts of words it contains.

For each image, the textual features are extracted from the associated web page title, keywords, description, and all text close to the image. For each category all such text documents are parsed into words, then the number of words are reduced by applying the stop-list and stemming process. A dictionary is build from these words and we represent the text data by the histogram of the word counts. The text kernel is represented as the linear kernel:

$$k^{(txt)}(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i \cdot \mathbf{x}_j. \tag{3.5}$$

For textual features we tried out different nonlinear kernels, but the performance of the linear kernel was comparable to the nonlinear kernels due to sparseness. Therefore, for reasons of efficiency we decided to use the simpler kernel.

3.3.2 Visual Features

For the visual feature extraction, we also use the standard visual bag-of-visual-words model. Each image is converted to gray scale and resized to a 300 pixel width, keeping the same aspect ratio. We then apply a regular dense grid with 10 pixels spacing and extract SIFT descriptor [2]. Each grid point is represented by four SIFT descriptors with circular support patches of radii 4, 8, 12, and 16. These multiple scale descriptors are used to provide a relative scale invariance. The dense descriptors are quantized into visual words using k-means clustering. The size of the codebook for each category is set to $d_{vis} = 600$, which is obtained from 50 randomly selected images positive images and 50 randomly selected negative images. Each image is then represented as a 600-dimensional histogram. We use the χ^2 pairwise distance to build a visual RBF kernel define as

$$k^{(vis)}(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\frac{D_{\chi^2}^{(vis)}(\mathbf{x}_i, \mathbf{x}_j)}{\tau}), \tag{3.6}$$

where $D_{\chi}^{(vis)}$ is the χ^2 pairwise distance which can be calculated as

$$D_{\chi^2}^{(vis)}(\mathbf{x}, \mathbf{z}) = \frac{1}{2} \sum_{k=1}^{d_{vis}} \frac{(x_k - z_k)^2}{x_k + z_k}. \tag{3.7}$$

We set σ as the average of the pairwise distances, $\tau = l^{-2} \sum_{i,j}^{l} D_{\chi^2}^{(vis)}(\mathbf{x}_i, \mathbf{x}_j)$, among l training samples.

3.4 Results

In this section we performed a number of experiments on the TVGraz dataset. To give a quantitative evaluations we measure the image ranking quality by precision-recall curves and mean average precision (mAP) criteria. Precision is the fraction of the retrieved images that are relevant, while recall is the fraction of the relevant images that are retrieved. Precision and recall are single-value performance metric on the whole number of retrieved images. By computing a precision and recall at every position in the ranked images, one can plot a precision-recall curve, plotting precision $p(r)$ as function of recall r. Average precision computes the average value of $p(r)$ over the interval from $r = 0$ to $r = 1$.

For each of the 10 categories of objects, we learned three support vector classifiers using textual representation, visual representation and their combination with MKL. For training each classifier, we randomly selected 50 positives and 50 negatives out the total images per category, shown in Table3.1. and the remaining images per category were used as as test set. This process is repeated for 10 run and for all categories. Table 3.2, compares the classification performance using the textual representation, visual representation and their combination with MKL. We observed that in 9 out of 10 categories the combination with MKL gives superior performance. This could also be verified by the precision-recall curves shown in the Figure 3.3.

Table 3.2 shows the average classification accuracy for each category. It can be observed that in 9 out of 10 categories a significant improvement is achieved by using combined representation with MKL. Figure 3.3, shows the average precision-recall curves and along with learned weights with MKL.

Figures 3.4, 3.5, and 3.6, compare the top ranked 100 images for the cactus class

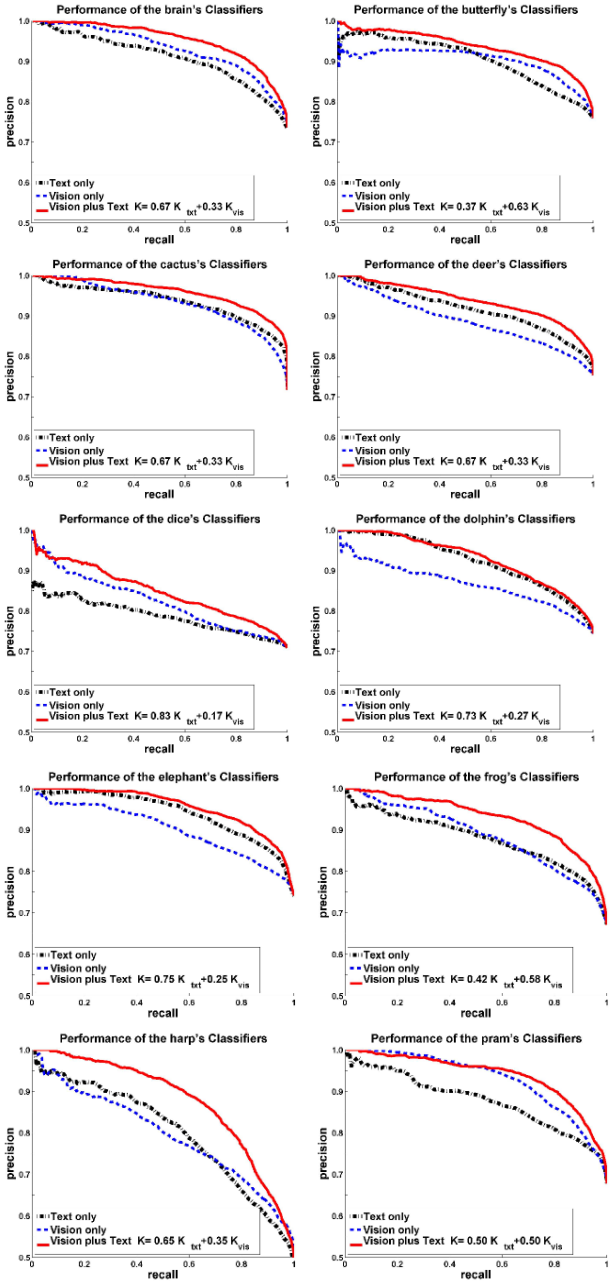


Figure 3.3: TVGraz dataset: Precision-recall curves for the 10 object categories, with textual kernel (Text only), visual kernel (Vision only) and with combined kernel (Vision plus text). For the combined kernel the weights learned with (MKL) are shown in the corresponding legend.

Category	Text	Vision	Text+Vision
brain	0.73	0.74	**0.81**
butterfly	0.72	0.71	**0.81**
cactus	0.78	0.76	**0.84**
deer	0.74	0.75	**0.80**
dice	0.67	0.61	**0.71**
dolphin	0.77	0.75	0.77
elephant	0.76	0.74	**0.79**
frog	0.64	0.73	**0.76**
harp	0.63	0.69	**0.73**
pram	0.73	0.77	**0.80**

Table 3.2: The mean AP scores on the TVGraz dataset with textual kernel (Text), with visual kernel (Vision) and with the combined kernel (MKL). The best scores are marked in bold.

with textual kernel, visual kernel, and with combined kernel respectively. It can be seen, that ranking images with combined kernel (MKL) removes many of the unrelated or outliered images and provides a better image search results than the individual kernel. These result confirms that image ranking quality can be improved using all available modalities would provide superior results.

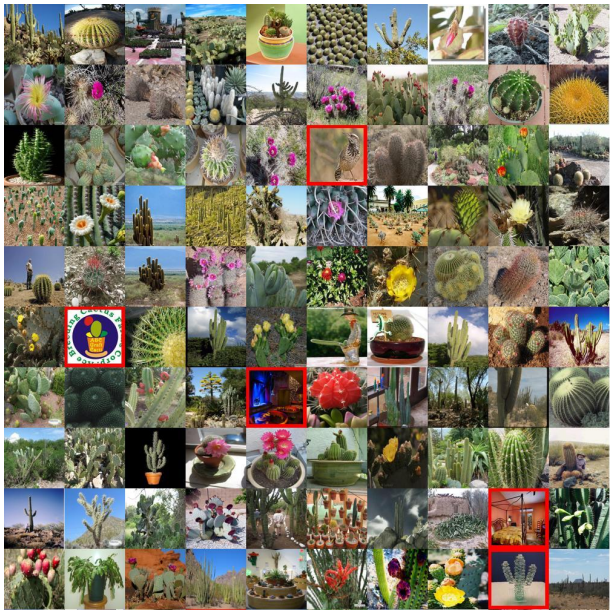


Figure 3.4: 100 highest ranked images for "cactus" categorization task with **textual kernel**, (the noisy images (not containing "cactus" are marked with red boxes).

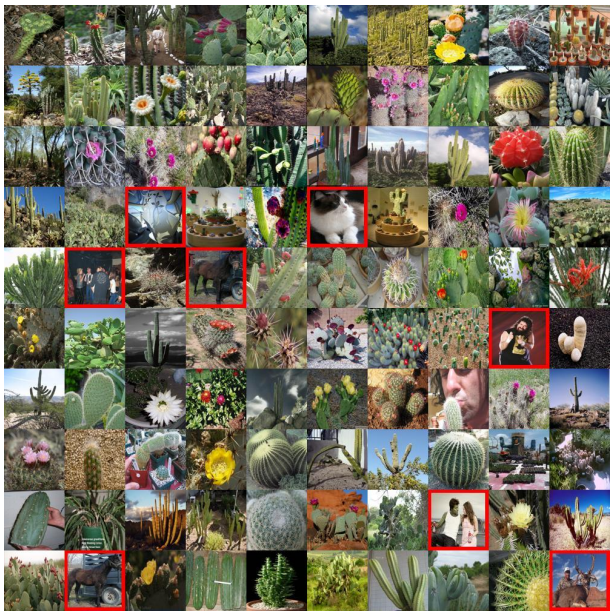


Figure 3.5: 100 highest ranked images for "cactus" categorization task with **visual kernel**, (the noisy images (not containing "cactus" are marked with red boxes).

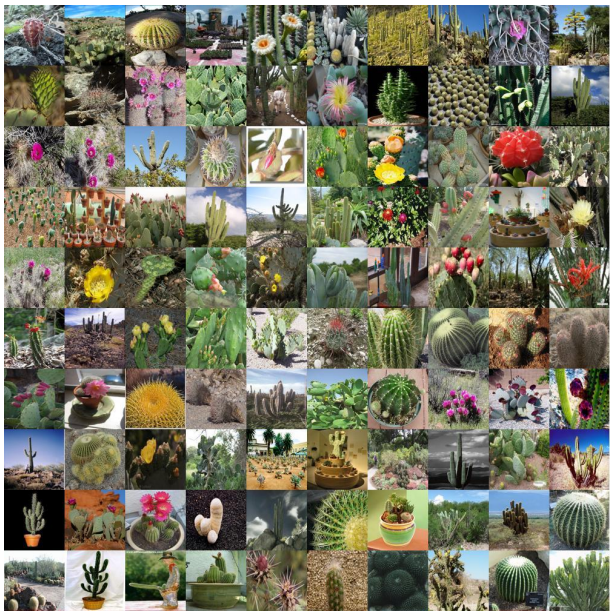


Figure 3.6: 100 highest ranked images for "cactus" categorization task with **combined kernel(MKL)**, (the noisy images (not containing "cactus" are marked with red boxes).

3.5 Conclusion

This chapter presented a MKL method for categorizing images by combining visual and textual features.A multiple kernel learning provides a natural solution for learning from heterogeneous data sources. We created the TVGraz dataset for benchmarking the web-based visual object category learning algorithms, which utilize both textual and visual information. The experimental results showed that the combined strategy outperforms the methods using only one of the available data streams.

The idea of combining textual and visual features requires however the availability of both data sources. Therefore, for classifying images where the text data is missing or is very sparse, further steps and algorithms should be developed. For the task of web image search this is not a major issue; even filtering out those images without text information, there will be enough images left for the given task. The Internet is a huge supply of images which are available at the typing of a single phrase or word, e.g., "car", using image search engines. However, the ranking quality an image search engine could be further enhanced if visual features are used along the textual data. This will decrease in the ambiguity in the true-class labels of images returned by a search engine.

Chapter 4

Co-regularized Multiple Instance Learning

This chapter presents a new method for learning object categories using the images returned by Internet image searches as training examples. The approach explicitly acknowledges and simultaneously accounts for their ambiguities in the true class-label of the images as well as for the large variations in visual characteristics of objects contained in them. The main contribution is a novel multiple instance learning algorithm which we call Co-miSVM (Co-regularized multiple instance SVM). The learning method is based on the principle of manifold regularization [12, 13], which extends the supervised multiple instance SVM [14] to semi-supervised multiple instance domain. This extension enables us to utilize different visual features and at the same time cope with the label noise in training an instance classifier. We collect groups of images called bags via multiple image search engines. Instead of grouping all images or instances returned by a search engine in one huge bag, we encapsulate the first few images in a positive bag, assuming that at least one of them may contain an object belonging the target category, and treat the rest as unlabeled images or unlabeled singleton bags; the negative images or singleton bags can be obtained by using keywords unrelated to the desired object category. The learned object models can be used for image ranking as well as for object category level image classification.

This chapter is organized as follows: Section 4.1 introduces the problems dealt

in this chapter. Section 4.2 reviews the previous work related to learning models from Internet images for object categorization. Section 4.3, presents the detailed formulation of the proposed learning algorithm, called co-regularized multiple instance SVM. In Section 4.4, we discuss our approach for obtained training bags of images directly from the Internet searches and the different types of features for image representations. Section 4.5, describes experiments performed to evaluate the proposed method and demonstrates its usefulness on a number of benchmark test datasets. Finally, Section 4.6 concludes the chapter.

4.1 Introduction

The performance of an object classifier is limited by the amount of labeled images available for training. Typically, its performance improves with increase in amount of labeled images during training. However, hand labeling or annotating large number of images is a laborious and an expensive task. The cost rises with increase in number of object's categories to learn. Nevertheless, keyword based Internet image searches such as Google, Yahoo, and, Bing are vital sources for collecting large amount of training images which can ultimately help in automatic or direct learning of object category classifiers on the fly [9, 17, 19, 20, 22, 27, 54, 55] using category specific queries.

There are two main problems in training category classifiers directly with Internet images which have been rarely addressed jointly so far. Firstly, the true class label of the returned images could be ambiguous; every image may not contains the objects of the desired category. A keyword-based image search engine returns images based on the relevance matching between the given keyword and the surrounding text or tags on the web pages containing them, without using any visual contents. Whereas the text does not guaranty about their visual contents because of difficulties in automatic association with an image contents, e.g., due to polysemic nature of text, i.e., a word or phrase could have multiple meanings. Secondly, there are wide varieties in the visual characteristics of objects within the same category found in the retrieved images, e.g., shape, appearance, scale, pose, and, number of objects. Different visual features, such as global (e.g., PHOG [1], BOW, PHOW [1], and GIST) as well as

local features (e.g., SIFT, MSER, and LBP) have been developed and specialized in capturing certain visual properties of an image. By using different features for image representation, the diverse visual properties of the Internet searched images can be characterized more precisely. If the available individual features are sufficient on their own for learning a category classifier, they may still provide complementary information to each other. However, for the Internet searched images, the challenge is how to combine different features effectively despite the variable amount of noise or ambiguities in their relevance to the desired category and without using any manually labeled images.

Most of the previous approaches such as [22],[55],[49] or [9], used Support Vector Machnines (SVM) based classifiers because of their ability to cope with the label noise in the training data. Moreover, this allows us to take the benefits of multiple features representation in learning category-level image classifiers (e.g., [4, 5]). A combined kernel is defined as the weighted sum of individual kernels. Multiple kernel learning (MKL) methods such as [6] or [7] aim to optimize the kernel weights while training an SVM classifier. However, a strong MKL classifier may not be directly trained on Internet images without additionally using hand labeled images [49, 55] or filtering out the unrelated and noisy images [22].

Another alternative, which has not been yet applied for object category recognition from the Internet searches, is to exploit different feature-based representations by using co-regularization such as [56], (discussed in Chapter 2), where support vector classifiers are learned in each representation through different types of features based representations. However being a single-instance semi-supervised approach, it still needs clearly labeled images.

To address these problems jointly, we extend the Multiple Instance Learning (MIL) based approach proposed by [9] for learning of object's categories from the Internet searches. In supervised learning such as SVM there is a clear knowledge about class-labels of the training images, there is a one to one mapping between the label and training image. In contrast, MIL is a relaxed form of supervised learning where training images are provided in the form of collections, called bags, where each bag is a collection of images and the class-label is assigned to bag instead of an image. All images in a negatively labeled bag are ensured to be negative while

the correspondence between label information and images in a positively labeled bag is ambiguous or uncertain, i.e., at least one of them is positive. The final task is to deliver image classifiers which can then be used to categorize any presented images other than the one obtained from the Internet as well for improving the performance of the keyword based image search engine by re-ranking the returned images based on their relevance to the desired categories. MIL explicitly solves learning problems with ambiguity in training samples and hence is most suitable for learning categories directly from Internet searches by collecting labeled bags of images via multiple keyword-based image search engines.

However, supervised MIL techniques such as [14] or [9] may fail to learn reasonable models if the noise level in positive bags is too high. The main contribution of this chapter is a novel *co-regularized* semi-supervised MIL algorithm allowing for learning from both labeled and unlabeled bags. It is important to note that encapsulating images in unlabeled bags carry no useful information, therefore we consider each image as unlabeled singleton bag. To exploit the power of multiple feature based image representations we apply ideas from single instance co-regularization [56]. In general, the proposed method is not limited to learning visual category models from Internet searches but could also be applied to other tasks such as action recognition [57], content based image retrieval [58], semantic segmentation [59], or text classification [14].

4.2 Related Work

Exploiting the Internet for learning category models has become quite popular. For instance, probabilistic clustering methods such as [17], [19], and [20] cluster images obtained via Internet search using a visual vocabulary. While [17, 19] extend the probabilistic Latent Semantic Analysis (pLSA) [18] by including spatial information, [20] uses a Latent Dirichlet Allocation (LDA) model to learn mixture of visual themes (topics). Then these methods re-rank the images based on dominant proportion of one of the learned topics in every image and such dominant topic is either selected manually or using a validation set of images. Such methods are mostly suitable for unsupervised object discoveries in a collection of images and the resulting high

level representation of an image as mixture of visual topics can be used in our discriminative approach in addition to other available representations.

Single instance supervised (SIS) approaches, e.g., [21–23, 49], have mostly selected to use SVM classifiers, since they allow to learn despite the noise in the training images up to certain extent. However, due to the variable amount of label noise in the keyword-based search images, a multi-stage learning framework is usually adapted where the strategy is to filter out some of the junk images which are totally unrelated to the desired category and then use the remaining as noisy training images for learning SVM [22, 23, 49] or voting based classifiers [21] with varying degree of supervision. The authors of [21] have used LDA to cluster the noisy images based on the textual features using the words occurring on the web-pages containing them. These clusters are then manually labeled into positives and negatives. The images and the associated text from these clusters are used to train a classifier using hybrid visual features (shape,color, and texture) and textual features. Finally, the classifier is used to re-rank the downloaded images. To reduce the manual labeling efforts, Schroff et al. [22] extend this approach for automatic collection of noise free images from the Internet image searches. They initially trained a radial basis function SVM based filter on hand labeled images to discriminate between abstract images such as plots, maps, charts, drawing and sketches and non abstract images. They filtered out the junk images using supervised trained filter and then used Bayes estimation using the text associated with the remaining images on the web pages to re-rank them and finally the top ranked images are used as noisily-labeled data to train a final SVM based visual classifier for the desired category, an image is represent by concatenation of normalized histogram of BOW and HOG with χ^2 radial function based kernel. The final visual classifier is then used to re-rank the originally downloaded images excluding the junk images for improving an image search engine performance. Guillaumin et al. [23] showed that other source of information such as surrounding text or tags can help to learn object categories but at the expense of manual labeling efforts. The images and accompanied tags are downloaded from Flickr photo sharing website. First, they train MKL classifiers using both text and 15 different image representations (L1 distance for the color histograms, L2 for GIST and χ^2 for PHOW) and use it to score the unlabeled images. Finally, they learn

classifiers, either SVM or least square regression using only visual features, from the MKL output on both labeled and unlabeled images. However, in comparison to all of these methods our approach is more simple and direct without using any hand labeled images for training an object classifier for the desired category.

A number of authors have proposed graph-based learning algorithms such as [24], [25], or [26] as well the recent one [27], which are especially developed for re-ranking the noisy Internet images. The basic idea is to exploits the data graph that reflect the intrinsic manifold structure [28] hidden in the images that tends to capture the underlying semantics. In these single instance learning based methods, the higher ranked few images are assumed as weakly labeled images and then conduct the re-ranking on the whole set of images returned by an keyword-based image search engine. The authors of [27] argued that the underlying manifold structure is beneficial to improve the quality of a search engine only if the outliers are removed. Consequently, they formulate a spectral filter to remove the outliers before using a manifold ranker [28]. Instead of removing outliers from images associated with keywords, the proposed method for visual category learning explicitly acknowledges and accounts for their ambiguity. Our approach is more closely related to [9] which is based on multiple instance SVM (discussed in Chapter 2). However, we extend single co-regularized Laplacian SVM (to be presented in the next section) to solve MIL problem. We will show that our method outperforms most of the related work by conducting experiments on benchmark datasets.

4.3 Co-regularized Multiple Instance SVM

The co-regularized SVM extends a SIS-SVM to semi-supervised learning inheriting their benefits to deal with different type of features based instances' representation and strengthen the classifier by utilizing labeled and unlabeled instances. Whereas MIL provides a learning framework from ambiguously labeled data, i.e., images obtained from the Internet where the supervision is porovided by the keyword-based image search engines. We can define problem of learning of object categories from Internet images as semi-supervised MIL. In this section, we formulate the proposed learning algorithm which we call co-regularized multiple instance SVM (co-miSVM).

We consider a binary classification problem. Given training examples containing l_b labeled bags $\mathcal{L}_B = \{(B_i, Y_i)\}_i^{l_b}$ where each B_i consists n_i instances and u unlabeled singleton bags $\mathcal{U} = \{(\mathbf{x}_i)\}_{i=1}^{u}$ (a singleton bag contains only one instance). The training set consists a total of $M = \sum_{i=1}^{l_b} n_i + u$ instances where each instance $\mathbf{x}_i = (\mathbf{x}_i^{(1)}, \cdots, \mathbf{x}_i^{(V)})$ is represented by V different features $\mathbf{x}_i^{(v)} \in X^{(v)}$. We want obtain a final instance classifier of the form:

$$\begin{aligned} f_*(\mathbf{x}) &= sgn\left(\mathbf{w}^{*T}\mathbf{h}(\mathbf{x}) + b\right) \\ &= sgn\left(\mathbf{w}^{*T}\left[h^{*(1)}(\mathbf{x}), \cdots, h^{*(V)}(\mathbf{x})\right] + b\right) \\ &= sgn\left(\mathbf{w}^{*T}\left[\sum_{i=1}^{l_b+u}\sum_{j=1}^{n_i}\alpha_{ij}^{*(1)}k^{(1)}(\mathbf{x}_{ij}, \mathbf{x}), \cdots, \sum_{i=1}^{l_b+u}\sum_{j=1}^{n_i}\alpha_{ij}^{*(V)}k^{(V)}(\mathbf{x}_{ij}, \mathbf{x})\right]\right), \\ &= sgn\left(\mathbf{w}^{*T}\left[\sum_{i=1}^{M}\alpha_{i}^{*(1)}k^{(1)}(\mathbf{x}_{i}, \mathbf{x}), \cdots, \sum_{i=1}^{M}\alpha_{i}^{*(V)}k^{(V)}(\mathbf{x}_{i}, \mathbf{x})\right]\right), \end{aligned} \tag{4.1}$$

where the $\mathbf{w} \in \mathbb{R}^V$ is the weight vector combining the results of the V classifiers $h^{*(v)}$ and b is the bias. Comparing with MKL based decision function (introduced in Chapter 3 Section 3.1), our final classifier can be seen as a multiple kernel classifier where the weights are defined by $\mathbf{w}$. The decision function (5.10) for MKL share the same set of co-efficients $\boldsymbol{\alpha}$ for all participating kernels. In contrast to MKL, we use separate sets of co-efficients $\{\boldsymbol{\alpha}^{*(v)}\}_{v=1}^{V}$ for each kernel where these co-efficients are expanded over unlabeled instances. To learn the weights $\mathbf{w}$, and $\{\boldsymbol{\alpha}^{(v)}\}_{v=1}^{V}$, we minimize the following objective functional

$$\min_{\{y_{ij}\},\mathbf{w},b} \min_{\{h^v\}_{v=1}^{V}} \frac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{i=1}^{l_b}\sum_{j=1}^{n_i}\ell(y_{ij}, \mathbf{w}^T\mathbf{h}(\mathbf{x_{ij}}) + b) \tag{4.2}$$

subject to:

$$\begin{aligned} & y_{ij} \in \{-1, +1\},\ \forall\ i = 1, \cdots, l_b,\ j = 1, \cdots, n_i \\ & y_{ij} = -1,\ \forall\ iY_i = -1,\ \ j = 1, \cdots, n_i \\ & \sum_{j=1}^{n_i} y_{ij} \geq 2 - n_i,\ \forall\ i : Y_i = 1, \end{aligned} \tag{4.3}$$

Note that the labels y_{ij} of instances in positive bags are unknown integer variables and are treated as optimization variables here. The first set of constraints ensures that all instances in negative bags are negative while the second set of constraints ensures that there exists at least one positive instance in every positively labeled bag. Due to unknown true labels of instances in positive bags this optimization problem is non-convex and difficult to solve. Therefore, our learning strategy is to decompose the problem (4.2) into smaller optimization tasks and iteratively refine the results.

4.3.1 Co-regularized Feature-specific Classifiers

We build an instance level graph where each vertex corresponds to an instance and the edge weight is defined by a similarity $S_{i,j}^{v}$ between a pair of instances as follows:

$$\mathbf{W}_{ij}^{(v)} = \begin{cases} \exp(-\frac{\|\mathbf{x}_i^{(v)} - \mathbf{x}_j^{(v)}\|^2}{2\tau^2}) & \text{if instance } j \text{ is in among the } k\text{-nearst} \\ & \text{neighbors of } \mathbf{x}_i \\ 0 & \text{otherwise} \end{cases} \tag{4.4}$$

We define $\mathbf{D}_{ii}^{v} = \sum_{j=1}^{n} \mathbf{W}_{ij}^{v}$ as diagonal matrix and the individual graph Laplacian $\mathbf{L}^{v} = \mathbf{D}^{v} - \mathbf{W}^{v}$.

4.3.2 Iterative Optimization

Once we obtain a set of solutions, we can solve the optimization problem in Eq. (**??**), which is similar to miSVM [14]. However, the features representing each instance are the responses obtained from co-regularized feature-specific classifiers. The proposed algorithm called *Co-miSVM*, is outlined in Algorithm **??**. In detail, we initialize instances labels by the corresponding bags labels and then starts with the iterative optimization. In each iteration first we learn $\{\boldsymbol{\alpha}^{(v)}\}_{v=1}^{V}$. Then we represent each instance in a bag by the response of feature specific classifiers. Based on this combined representation of instances we train our final SVM instance classifier with labeled instances contained in labeled bags. Then we predict class labels and compute confidence for all instances $\mathbf{x} \in \mathcal{L}_B$. If the predicted labels for all instances do not change and the MIL constraint is satisfied then we terminate the iterations.

Algorithm 1 Co-miSVM

Input: training labeled bags $\mathcal{L}_B = \{(B_i, Y_i)\}_{i=1}^{l_b}$ and
unlabeled bags $\mathcal{U} = \{B_i\}_{i=l}^{u}$ s.t. $\mid B_i \mid = 1$
where an instance $\mathbf{x}_i = (\mathbf{x}_i^{(1)}, \cdots, \mathbf{x}_i^{(V)})$
Input: hyper parameters: C, $\gamma_A^v, \ldots, \gamma_A^V$ and $\gamma_I^v, \ldots, \gamma_I^V$
Output: $\mathbf{w}$,b and $\{\boldsymbol{\alpha}^{(v)}\}_{v=1}^{V}$
$\mathbf{L} \leftarrow \frac{1}{V}\sum_{v=1}^{V} \mathbf{L}^v$
$y_{ij} \leftarrow Y_i,\ \forall i : i = 1, \cdots, (l_b + u)$ and $j = 1, \cdots, n_i$ (initialize instance labels)
repeat
  for $v = 1$ to V **do**
    $h_*^v \leftarrow$ LapSVM solution using instance labels $\{y_{ij}\}$
  end for
  $\mathbf{h}(\mathbf{x_{ij}}) \leftarrow [h_*^1(\mathbf{x_{ij}}), \ldots, h_*^V(\mathbf{x_{ij}})]^T\ \forall i : i = 1, \cdots, l_b$ and $j = 1, \cdots, n_i$
  $f_* \leftarrow$ compute SVM solution $\mathbf{w}, b$ with current $\mathbf{h}$ and instance labels$\{y_{ij}\}$
  $y_{ij} \leftarrow sign(\mathbf{w}^T\mathbf{h}(\mathbf{x_{ij}}) + b)\ \forall i : i = 1, \cdots, l$ and $j = 1, \cdots, n_i$
  for (every positive bag) **do**
    if $\sum_{j=1}^{n_i} y_{ij} < 2 - n_i$ **then**
      $j^* \leftarrow \arg\max_{j \in n_i} \left(\mathbf{w}^T\mathbf{h}(\mathbf{x_{ij}}) + b\right) \forall i Y_i = 1$
      $y_{ij^*} \leftarrow 1$
    end if
  end for
until estimated instance labels $\{y_{ij}\}$ are unchanged
Return $\mathbf{w}$,b and $\{\boldsymbol{\alpha}^{(v)}\}_{v=1}^{V}$

4.4 Collecting Bags of Images and Image Representations

In this Section, first we present our approach for collecting training data directly from the Internet without using any an manual labeling efforts and then we give the implementation details for image representation.

4.4.1 Collecting Bags of Images via the Internet Searches

We explicitly model this ambiguity by collecting bags of images via a number of different image search engines and assuming that the desired category in present in at least one of those images in a bag instead of all, an example for "motorbikes" is shown in Figure 4.1. The quantity of labeled bags of images can be increased by translating a given category name into a number of languages using an automatic translation tool * before querying a search engine. Rather than including all of the images returned by a search engine in one huge bag, we assume that a fixed

*`http://www.google.com/language_tools?hl=en`

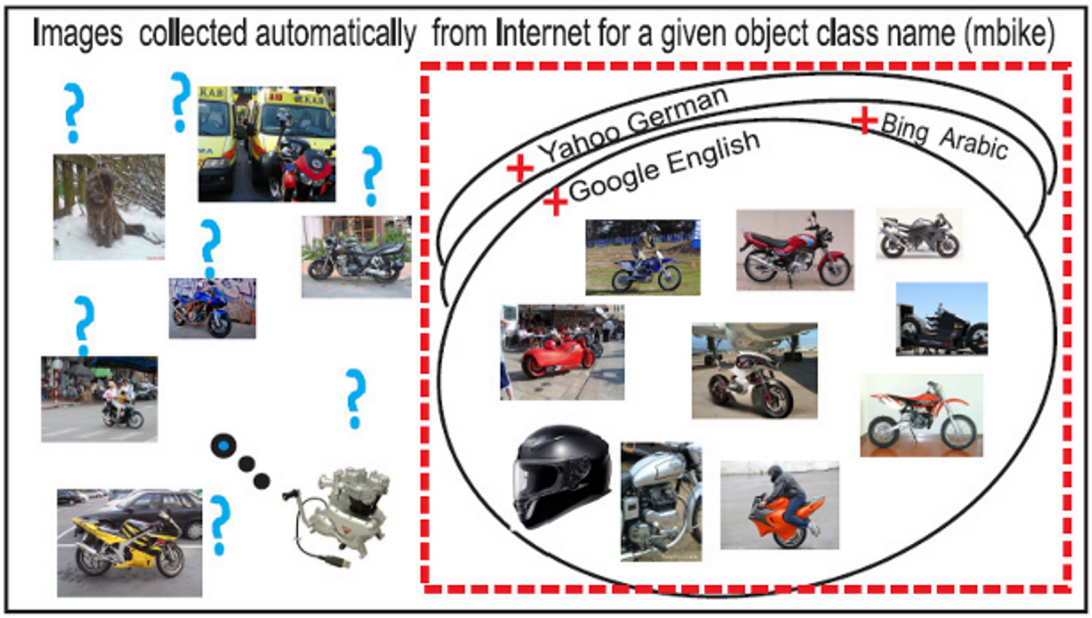


Figure 4.1: Learning object categories from weakly labeled images: The name of the object category is entered in a search machine and corresponding images are returned. To cope with weakly labeled data the proposed semi-supervised MIL approach is applied for learning.

small number of top ranked images may contain at least one positive image and the rest are considered as unlabeled bags. Similarly, the negative bags of images can be collected either from Internet searches of other categories or any back-ground images, if available. Encapsulating images in unlabeled bags carry no useful information therefore we consider each image as unlabeled singleton bag, such images can be helpful in discriminating the desired category from un-related images. Collecting bags of images in this way requires no manual labeling efforts; instead requires only the name of categories. These bags are used in training of binary object category classifier using the proposed algorithm co-miSVM. The resulting classifier can then be used for re-ranking the images returned by a search engine or classifying any other unseen image.

4.4.2 Image Representations

In experiments presented in Sections 4.5.1 and 4.5.2, we use two types of features to represent an image. To capture shape and appearance together with the image

spatial layout we use pyramid of histogram of oriented gradients (PHOG) and pyramid of histogram of visual words (PHOW) [1]. We select these features as similar representation have been used in [9, 20] for the tasks in hand. In following, we give implementation details for these features.

Shape

We extract edge contours using the Canny edge detector and then compute the orientation gradients using a 3×3 Sobel mask. We use shape descriptors with all the orientations in a range $[0; 360]$. The number of histogram bins is set to 40 where each bin represents the number of edges that have orientations within the range. Finally, we obtain PHOG, an entire image descriptor to capture shape together with spatial layout while forming a pyramid we fix the number of levels equal to 2.

Appearance

The gray scaled SIFT descriptors are computed densely on a regular grid with each cell of five pixels. At each grid point they are computed over circular patches with four different radial scales of 4,8, 12 and 16 pixels. Thus each point is represented by 4 SIFT descriptors to allow for scale variations between images. We build a visual vocabulary of size 1000 using k-means clustering to vector quantize the dense features into visual words. It is important to note that we build separate visual vocabularies for each experiment, using the training images downloaded from the Internet. We carry out the k-means clustering by random sampling of SIFT descriptors over 10 training images per category. Finally, we obtain PHOW descriptors to capture appearance together with spatial layout of image while a pyramid is formed by setting the number of levels equal to 3.

4.5 Experiments

In this section, we conduct experimental analysis in three ways to show the benefits of the proposed approach. First, we compare various SVM-based classification methods for automatic learning of various categories from Internet searched images. Then we provide comparisons with state-of-the art methods on a task of re-ranking

the Google searched images and evaluate on web image test datasets. Finally we evaluate it on a datasets used in most studies of MIL methods. In all experiments, we use the evaluation metrics and procedures as used by others in similar experiments.

4.5.1 Automatic Training of Category Classifiers

To validate the proposed approach for learning object categories when the training images are automatically obtained from the Internet searches, we compare the following seven SVM based methods. The first two, SVM-1 and SVM-2, are standard single instance supervised SVMs. The next two, LapSVM [12] and Co-LapSVM [56], are single instance semi-supervised SVM which can use of both labeled and unlabeled images. The next two are supervised multiple instance SVMs, mi-SVM-1 [14] and mi-SVM-2 which can make use of labeled bags of images. The last one is the proposed Co-miSVM that makes use of both labeled and unlabeled bags of images during the training stage. For all the methods we use an equal number of training images which are downloaded from Internet image searches. For each method however, they are assumed to be either labeled and unlabeled images or labeled and unlabeled bags of images.

For given names of 7 categories of objects, i.e., airplanes, motorbikes, trains, cars, buses, cows and horse; the training set of images from the Internet. We retrieve these images using keywords: airplane, motorbike, train, car, bus, cow and horse via 3 image search engines where each keyword is translated from English language into other 12 different languages before querying a text based image search engine. In this way, we have 36 different collections of images for each category. In each collection we downloaded 60 images that result in 2160 images per category. To represent training and later test images, we use features presented in Section 4.4.2. We use χ^2 distance RBF based kernels in all of classification methods, $k^{(1)}(\mathbf{x}_i, \mathbf{x}_j) = exp(-\frac{D^{(1)}_{\chi^2}(\mathbf{x}_i,\mathbf{x}_j)}{\tau})$, where the bandwidth τ is set to the median of the pairwise instances distances. We train them as binary image classifiers for every object's category with the searched images by assuming that the images are labeled in following way:

SVM-1 and SVM-2: We assumed all of 2160 images obtained for a certain object category as positively labeled, ignoring their ambiguity. The negative images are taken from the searches for other categories. To see the effect on accuracy, we

Methods ↓	Images			Num of Bags / images per bag		
	pos	neg	unlabeled	pos	neg	unlabeled
SVM-1	2160	12960				
SVM-2	2160	12960				
LapSVM	1080	6480	7560			
Co-LapSVM	1080	6480	7560			
miSVM-1				36 / 60	216 / 60	
miSVM-2				36 / 60	216 / 60	
Co-miSVM				36 / 30	216 / 30	7560 / 1

Table 4.1: The images downloaded from the Internet Searches for the 7 categories and used in training for the 7 classification methods, for detail see Experiement 4.5.1

represent an image by concatenation of both features in case of SVM-1 while for SVM-2 we simply take the mean of the two χ^2 based kernels values obtained from the individual representation.

LapSVM and Co-LapSVM: Here for a certain object category, we assumed the top ranked 30 images per image search to be positive, and the remaining 30 images per image search as unlabeled images. The negative images and additional unlabeled images are taken from the searched returns for other categories. In case of LapSVM, we represent an image by concatenation of PHOG and PHOW while for Co-LapSVM an image is represented by the individual measure.

mi-SVM-1 and mi-SVM-2: The top 60 images are taken as member of a positive bag and the lower ranked images as unlabeled singleton bags. This results in a collection of 36 positive bags for a target category. The negative bags are created from searches for other categories. For mi-SVM-1 we represent an image by concatenation of both features while for mi-SVM-2 we use two supervised SVMs as feature specific classifiers.

Co-miSVM The top 30 images are taken as member of a positive bag and the lower ranked images as unlabeled singleton bags. This results in a collection of 36 positive bags and 1080 unlabeled bags for a target category. The negative bags and additional unlabeled bags are taken from the search for other 6 categories. For all seven methods the training data is summarized in Table 4.1.

The graph Laplacian matrices used in Co-miSVM, LapSVM and Co-LapSVM are constructed with the number of nearest neighbors $k = 10$, $\tau = 0.5$, γ_A is set to

	aeroplane	mbike	train	car	bus	cow	horse
SVM-1	80.80	78.88	73.58	77.68	84.11	75.88	59.57
SVM-2(avg kernel)	80.12	79.17	74.04	77.34	83.27	74.90	59.28
LapSVM	80.07	77.69	78.57	74.10	84.94	78.82	66.61
Co-LapSVM	81.12	78.81	78.43	77.50	84.01	79.02	66.10
miSVM-1	80.80	78.88	73.58	77.68	84.11	75.88	59.57
miSVM-2	82.01	81.17	76.04	79.34	85.37	76.09	61.27
Co-miSVM	**86.87**	**88.65**	**82.54**	**84.77**	**86.16**	**84.04**	**68.09**

Table 4.2: Performance scores in percent mAP on the entire test images, taken from ImageNet [60] dataset, for the 5 binary classification methods when trained directly on Internet searched images. Best scores are indicated by boldface numbers.

1, and $\gamma_I = l_b/M^2$. For all the seven methods we set the hyper-parameter C=10. These values are fixed with the intension to fairly compare the methods for the task of object categories learning in an automatic or unsupervised setting. No model selection method is applied which would otherwise increase performance equally of each method.

We evaluate these methods on ImagNet[60] test dataset after they are trained with images collected using keyword-based image searches. ImageNet [60] is a benchmark dataset containing images of various common object categories and is provided with ground truth labels. We choose a test set from it, containing 7926 images from the seven object's categories where each category has more than 1000 images. Testing on all these images allows us to compare the binary classification accuracies of all methods which are trained directly on ambiguous Internet searched images. A test image is considered as a singleton bag when presented to a MIL based classifier. The binary classification performance of the methods as measured by the mean Average Precision (mAP) (The PASCAL metric) on the entire test set images is tabulated in Table 4.2.

It can observed from the results presented in Table 4.2 that in spite of the same number of training images the individual performance scores gradually improve starting from single instance supervised methods then single semi-supervised and

finally multiple instance methods. Multiple instance learning based method nicely model the ambiguity in the true labels of images obtained from the Internet. Finally, the overall results suggest that Co-miSVM effectively made use of the unlabeled images using the two representations during training.

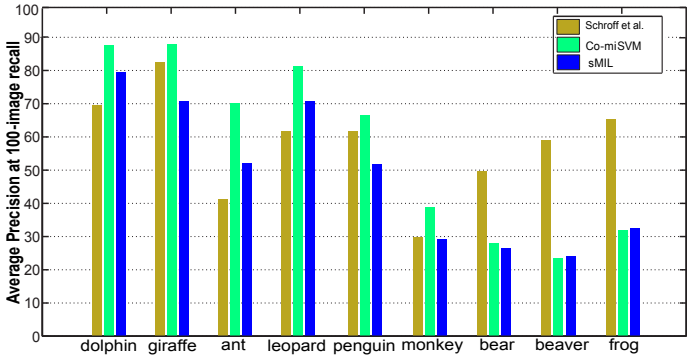
4.5.2 Re-ranking Google Searched Images

4.5.2.1 Berg Dataset

First we compare the performance of our method with sMIL [9] and an SIL based method proposed by Schroff et al. [22]. We consider the re-ranking of the entire Berg [21] dataset images after training on images automatically collected from the Internet searches. The Berg dataset [21] contains 10000 images for a total of 10 different categories of animals. The images are noisy as they were downloaded from the Google image searches and include abstract images such as comics, graphs, plots and sketches. These are provided with ground truth labels by the authors [21] indicating their relevance to a certain animal's category.

For the 10 categories, we collect bags of images from Internet searches. Each object's category name is translated into five different languages (English, German, Spanish, Italian and French) before querying each of three search engines (Yahoo, Google and Bing). This results in 15 positive bags of images for each category. For training sMIL [9] method, we download top ranked 50 images per search query and group them to form a positive bag. Similarly, for training Co-miSVM method, we used the top ranked 20 images to build a positive bag and remaining 30 images as unlabeled singleton bags. The negative bags of images and additional unlabeled bags are taken from the search return for other categories. Note, all these images obtained automatically by executing a script without any manual supervision and the same number of total images are used for training of both models.

To represent an image we use the same features, as presented in Section 4.4.2. For Co-miSVM, the graph Laplacian matrices are constructed with the number of nearest neighbors $k = 10$, $\tau = 0.5$, γ_A is set to 1, $\gamma_I = l_b/M^2$ and C=10. For all the seven methods we set the hyper-parameter C=10. While hyper-parameters C for sMIL training is selected via cross-validation using one-held out bag, as done in [9].



(a) Berg [21] test dataset

Figure 4.2: Performance comparison of Co-miSVM and sMIL [9] with Schroff et al. [22], on the Berg [21] test dataset. The plot shows the precision in percent at 100-image recall level.(**Higher is the better**).

We use two χ^2 based kernels for Co-miSVM where the width of each kernel is set to the mean of χ^2 pairwise distances between images. While for sMIL we take the mean two χ^2 based kernels values obtained from both representations.

Figure 4.2 compares the results reported in [22] with the proposed method and sMIL [9]. It can be seen, that our approach achieves higher precision for most of animals categories with only bear, beaver and frog being out performed by Schroff et al.[22] method. However in those cases where we performed equally with sMIL may be explained by high level noise images. Compared with sMIL, the results show that our assumptions regarding the noise level of images retrieved automatically from web is correct and unreliable images as unlabeled singleton bags results in the best performance.

4.5.2.2 Fergus Dataset

Second, we evaluate our proposed method by training with Fergus dataset containing a total of 4091 images for the seven categories (airplanes, cars rears, faces, guitars, leopards, motorbikes and wristwatches). These images were downloaded from Google image search using each category name as a keyword. The images are provided with ground truth labels by the authors indicating their relevance to the queried category. The image set contains junk images which are totally unrelated to the keyword categories. As judged in [17], e.g., the images collected for airplanes

	aero	cars rear	face	guitar	leopard	mbike	wrist watch	**Mean**
Google	50	41	19	31	41	46	70	43
SVM [22]	35	-	–	29	50	63	93	54
TSI-pLSA [17]	57	77	82	50	59	72	88	69
LDA [20]	**100**	83	**100**	91	65	97	**100**	91
MRank [28]	39	53	66	32	50	79	75	56
SpecFilter + MRank [27]	86	**100**	75	58	63	79	**100**	80
sMIL [9]	**100**	95	75	72	72	80	93	84
Co-miSVM	99	**100**	83	**96**	**92**	**100**	95	**95**

Table 4.3: Performance Comparison on Fergus dataset [17]. Ranking precision, in percent, at 15% recall.

contain 73% junk images. The task is first to train a certain category model using the raw Google data and then re-rank the entire dataset by estimating the relevance of images to the target category.

Following the experimental protocol used in [9], we also group the images into multiple bags of images by random selections of bags of size 25 images. We trained binary image classifiers with one third of these bags per category as positively labeled and the rest are taken as unlabeled images while the negative bags are taken from other categories.

The width of the kernel is set to the mean of χ^2 distances between the pairwise instances. The graph Laplacian matrix is constructed with $k = 10$, $\tau = 0.5$. While hyper-parameters C, γ_A and γ_I are via cross-validation using one-held out bag, as done in [9].

The final results are presented in Table 4.3 for comparison with two probabilistic topic based models (TSI-pLSA [17] and LDA [20]), an SVM-based method proposed by Schroff et al. [22] using both text and image features, two graph based models (MRank [28], SpecFilter + MRank [27]) and an MIL based method (sMIL [9]).

The results show that our Co-miSVM achieves best mean precision and the performance is more consistent then other methods.

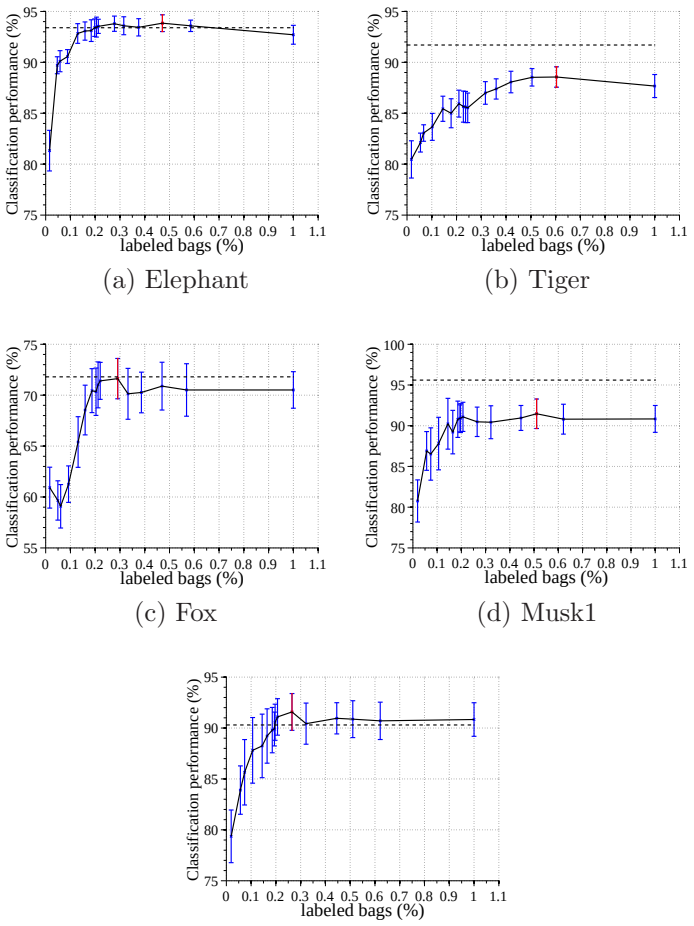
4.5.3 MIL Benchmark Datasets

This section present experiments on five datasets, i.e., three image classification datasets (Tiger, Elephant and Fox) and two drug activity prediction datasets (Musk1 and Musk2). These datasets are used as benchmark by different, supervised and

semi-supervised, state-of-the-art MIL algorithms designed for various tasks. Including mi-SVM [14], MI-SVM [14], AW-SVM [61], miGraph [62], CRF [63], PPMM [64], MILBoost [65] and IL-SML [58]. Our purpose of evaluation is twofold: firstly, to show the generality of the proposed algorithm by giving comparison to various methods. Secondly, we want to show the impact of unlabeled bags on the classification performance. The classification performance is measured using the mean area under the receiver operating curve (AUC).

In the image classification datasets, an image is considered as a bag of segmented regions. There are about 100 positive and 100 negative bags, each containing 2 to 13 instances where an instance is represented by one 230 dimensional vector. The Musk1 dataset has 47 positive bags and 45 negative bags with an average of about 5.2 instances per bag. The Musk2 has 39 positive bags and 63 negative bags, however the number of instances per bag is much larger, i.e., an average of about 64.5 instances per bag. For the drug activity datasets, an instance is represented by a 166 dimensional feature vector.

We performed experiments on each dataset via repeating 10-fold cross validation for 20 times. In order to analyze the impact of using unlabeled training bags on performance, in each 10-fold cross validation we pick a fixed percent out of the 9-training folds and split it into labeled and unlabeled bags, keep the labels of the labeled bags and remove the label information in the unlabeled bags. For example, we select 40 percent out of the 9-training folds as labeled training bags and the remaining 60 percent as unlabeled singleton bags for training Co-miSVM. As others have used only one kernel because of the given type of instance representation. Therefore, we use an euclidean distance based RBF kernel $k^{(1)}(\mathbf{x}_i, \mathbf{x}_j) = exp(-\frac{D^{(1)}(\mathbf{x}_i,\mathbf{x}_j)}{\tau})$, where the bandwidth τ is set to the median of the pairwise instances distances. The value of C is set to 10. The sparse kNN adjacency graph $\mathbf{W}$ is constructed with k is set to 10 and the euclidean distance measure with $\sigma = 0.05$. The values for parameters, $\gamma_A^{(1)}$ and $\gamma_I^{(1)}$ are tuned once using a coarse grid search from 10^{-3} to 10^4 for each dataset. These 20 runs of individual 10-fold cross validation are then further repeated for various percentages of labeled and unlabeled training bags. The per-fold average test classification performance at various percent of labeled and unlabeled training bags is shown in Figure 4.3. It can be seen that for each dataset the performance



(a) Elephant (b) Tiger

(c) Fox (d) Musk1

(e) Musk2

Figure 4.3: The classification performance, in percent, on five MIL benchmark datasets. For each percent of label bags out of the 9-training folds, the plots show average performance over 20 runs of the individual 10-fold cross validation.

increases up to a certain peak point, marked as red, where the ratio of label and unlabeled bags, out of the 9-training folds, is appropriate. Notice, that the number of unlabeled training bags decreases with increase in percent of labeled bags. The final best results are shown in Table 4.4 in comparison with other methods. The results show that our method achieves either the best competitive results compared to other state-of-the art methods. In majority of the dataset we give either the best or the second best performance.

		Elephant	Tiger	Fox	Musk1	Musk2
MISVM	RBF [14]	73.1	66.6	58.8	77.9	84.3
	linear [14]	81.4	84.0	57.8	-	-
	poly [14]	79.0	81.6	59.4	-	-
miSVM	RBF [14]	80.0	78.9	57.9	87.4	83.6
	linear [14]	82.2	78.4	58.2	-	-
	poly [14]	78.1	78.1	55.2	-	-
AW-SVM(RBF) [61]		82.0	83.0	64.0	86.0	84.0
IL-SMIL [58]		82.1 ± 2.7	80.3 ± 3.3	57.1 ± 4.6	84.1 ± 4.8	83.8 ± 4.2
PPMM[64]		82.4	82.4	60.3	**95.6**	81.2
MILBoost[65]		**93.43**	**91.70**	**71.80**	81.98	81.87
CRF[63]		85.0	83.0	67.5	88.0	86.3
miGraph[62]		86.8 ± 0.7	86.0 ± 1.6	61.6 ± 2.8	90.0 ± 3.8	90.3 ± 2.6
Co-miSVM		$\mathbf{93.9 \pm 0.8}$	$\mathbf{88.7 \pm 1.0}$	$\mathbf{71.6 \pm 1.9}$	$\mathbf{91.5 \pm 1.8}$	$\mathbf{91.6 \pm 1.8}$

Table 4.4: Classification performance comparisons, in percent, on five MIL benchmark datasets. We report the average performance over 20 runs of individual 10-fold cross validation. For each dataset two best results are shown in bold.

4.6 Conclusion

In this chapter, we proposed a new method for learning object categories directly from Internet images by just entering the objects' name. Since in this way noisy labeled data is obtained we introduced a semi-supervised MIL method, Co-miSVM, that is additionally able to use multiple feature representations. We evaluated our method on standard MIL benchmark and realistic image datasets to show the effect of the different layers involved and achieved good performance against other approaches.

Chapter 5

Training Supervised Object Detectors

This chapter presents a multi-stage framework for training supervised part-based object detectors. The system requires a list of target object categories names as input. Using only this information, training bags of images are collected via Internet searches following the method presented in Chapter 4. In particular, we first train an image classifier that specifies the presence or absence of a target object. Then in a localization step we crop image patches describing the object, which are finally used to train an object detector.

This chapter is organized as follow: Section 5.1 gives a review of the problem. Section 5.2, presents the details about the types of features, kernels and the formulations of the proposed extension to multiple instance SVM. Section 5.3, discusses the multi-stage framework. Section 5.4, conducts experiments to demonstrate the proposed system on benchmark datasets. Finally, Section 5.5 concludes the chapter.

5.1 Introduction

In Chapter 1 it has been discussed that the key word-based image search engines allow for gathering large amounts of unlabeled images without any manual effort. The images are filtered and ranked by the search engine according to the textual information in their close surroundings. Even if most of the retrieved images are

related to the visual concept, they also contain completely unrelated images or noisy labeled images. While category classification models can be learned (e.g., it is demonstrated in Chapter 4) using such images it is more difficult to train object detector, since it often requires the exact location of the object of interest. For this reason, previous work (see Chapter 2.1) has mainly focused on learning models for object classification.

In this chapter, we build on these ideas and propose a system to learn an object detector from images automatically obtained from the Internet, requiring only the name of the target class for training a sophisticated part based detector, i.e., LSVM-part-based detector proposed in [66]. In particular, we first learn an object model that specifies the presence or absence of a target object, for this stage the training images are collected using multiple images search engines. This object classification model is then used to rank the collected images, which are provided as input training images to the next stage. Then in the localization step, a classifier is trained and then applied to localize the target objects in the given images. The system then crops those image patches that contains and describes the target objects. Finally in the last stage, those cropped images are used to train the LSVM-part-based detector proposed in [66].

In the classification and localization stage, multiple instance SVM [9] is used for training a classifier and a detector. This method is extended to semi-supervised MIL in order to learn a better model by exploiting different types visual features. To demonstrate the benefits of the overall system, we show results for two different publicly available datasets.

5.2 Representation and Multiple Instance Learning Method

To learn from Internet images, we need a robust method which is able to cope with un-surely labeled data. Thus, we use multiple instance learning. First, we define formally the problem we want to solve:

Definition of the problem

Given a training set $\mathcal{L}_B = \{(\mathcal{B}_i, Y_i)\}_{i=1}^{l_b}$ in the form of l_b bag-label pairs and $\mathcal{U}_B = \{\mathcal{B}_i\}_{i=l_b+1}^{u_b}$; where a labeled bag $\mathcal{B}_i$ is a collection of n_i instances $\mathcal{B}_i = \{\mathbf{x}_{ij}\}_{j=1}^{n_i}$; an unlabeled $\mathcal{B}_i \in \mathcal{U}_B = \{\mathbf{x}_{ij}\}_{j=1}^{1}$ is a singleton bag with one instance, $\mathbf{x}_{ij} \in \mathcal{X}$; and $Y_i \in \{-1, +1, 0\}$ is its class label. The bag label Y_i provides information about an instance $\mathbf{x}_{ij}$ label y_{ij} which can be explained by the following statements:

$$Y_i = +1 \Rightarrow \exists\ \mathbf{x}_{ij_o}, j_o \in \{1, \cdots, n_i\} : y_{ij_o} = +1, \tag{5.1}$$

$$Y_i = -1 \Rightarrow \forall \mathbf{x}_{ij}, j_o \in \{1, \cdots, n_i\} : y_{ij_o} = -1, \tag{5.2}$$

$$Y_i = 0 \Rightarrow \forall\ \mathbf{x}_{ij_o}, j_0 = 1 : y_{ij} = 0. \tag{5.3}$$

A positively labeled bag ensures that it contains at least one instance that can be assigned a positive label; there is no other information about the remaining instances in a positive labeled bag which might be completely unrelated belonging to neither positive nor negative class, While a negative labeled bag ensures that all instances it contains are assigned negative labels. Additionally, we are given a set of V features $\Phi^{(v)} : \mathcal{X} \rightarrow \mathbb{R}^{d_v}$, $\mathbf{x} \mapsto \mathbf{x} := \Phi^{(v)}(x)$ where d_v represents the dimensionality of the v'th features. For each feature type we can define kernel function that measure the similarity between instances that belong to labeled bags $\tilde{k}^{(v)} : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}, (\mathbf{x}, \mathbf{x}) \mapsto \tilde{k}(\mathbf{x}_{ij}, \mathbf{x}_{ij})$, which are elements of the RKHS space of functions defined as

$$\mathcal{H}_{\tilde{K}^{(v)}} = \left\{ f^{(v)}(.) = \sum_{i=1}^{l_b} \sum_{j=1}^{n_i} \alpha_{ij}^{(v)} \tilde{k}^{(v)}(\mathbf{x}_{ij}, .),\ l_b \,\text{and}\, n_i \in \mathbb{R},\ \alpha_{ij}^{(v)} \in \mathbb{R} \right\} \tag{5.4}$$

Now, the problem of learning an instance classification function $y : \mathcal{X} \rightarrow \{+1, -1\}$ from the kernels and the training bags of instances, is called semi-supervised multiple instance multiple kernels combination problem.

In the Stage 1: a bag is collection of images so an instance is an image, these bags of images are collected via key-word based search engines. In Stage 2: a bag representing an image is a collection of its sub-regions. We use two types of a features, for $\Phi^{(v)}$ PHOG and PHOW to represent an instance in a bag. In the following, first we discuss the details of for these representation and then we give

our proposed solution for solving this problem as multiple instance multiple kernel problem.

5.2.1 Representation

We use two types of features to capture the shape and appearance of an object, i.e., pyramid of histogram of oriented gradients (PHOG) and pyramid of histogram of visual words (PHOW) [1]. For PHOG the number of histogram bins is set to 40 and pyramid level is 2*. Similarly for each image, the color SIFT descriptors on a dense regular grid of 16×16 pixel patches are computed with a spacing of five pixels and at four different scales. Such a dense grid is also necessary to capture any uniform regions such as sky and a road surface while the multi-scale allow for scale variations between images. We perform a k-means clustering of a random subset of patches collected from the training images to build a visual vocabulary of 1000 visual words. Each instance in a bag, is then represented by a PHOW with 2×2 sub-divisions.

We use a χ^2 distance RBF kernel since χ^2 this has been extensively used for such feature comparisons. This kernel $\mathbf{K}^{(1)} \in \mathbb{R}^{M \times M}$ is defined as

$$\mathbf{K}_{ij}^{(1)} = k^{(1)}(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\frac{D_{\chi^2}^{(1)}(\mathbf{x}_i, \mathbf{x}_j)}{\tau}), \tag{5.5}$$

where $D_{\chi^2}^{(1)}$ is the χ^2 pairwise distance defined as

$$D_{\chi^2}^{(1)}(\mathbf{x}, \mathbf{z}) = \frac{1}{2} \sum_{k=1}^{d_1} \frac{(x_k - z_k)^2}{x_k + z_k}. \tag{5.6}$$

and the width of the kernel σ is taken as the average pair wise distance, $\tau = M^{-2} \sum_{i,j}^{M} D_{\chi^2}^{(1)}(\mathbf{x}_i, \mathbf{x}_j)$, where M is the number of training instances in all the labeled and unlabeled training bags, $M = \sum_{b=1}^{l_b+u_b} n_b$.

Similarly, a χ^2 distance based RBF kernel is used for PHOW type of features.

*We use these fixed numbers based on the best results reported in [1].

5.2.2 MKL method with Multiple Instance Learning

Since, we unlabeled singleton bags with no labeled information. Therefore, we used a co-regularized a kernel function, discussed in Chapter 2, that can can exploit manifold structures of the feature spaces. For a kernel function $\tilde{k}^{(v)}$ to measure similarity between two real valued vectors, we define it as

$$\tilde{\mathbf{K}}_{ij}^{(v)} = \tilde{k}^{(v)}(\mathbf{x}_i, \mathbf{x}_j) = k^{(v)}(\mathbf{x}_i, \mathbf{x}_j) - \mathbf{K}_i^{(v)^T}(\mathbf{I} + \frac{\gamma_I^{(v)}}{\gamma_A^{(v)}}\mathbf{L}_C\mathbf{K}^{(v)})^{-1}\frac{\gamma_I^{(v)}}{\gamma_A^{(v)}}\mathbf{L}_C\mathbf{K}_j^{(v)^T}. \quad (5.7)$$

where $\tilde{\mathbf{K}} \in \mathbb{R}^{(M-u_b)\times(M-u_b)}$ is the kernel matrix defined over instances from labeled bags l_b, and $L_C \in \mathbb{R}^{M\times M}$ is the combined graph Laplacian, see Chapter 2.2.2.

$$\min_{\{y_{ij}\}_{i=1}^{l}} \min_{\mathbf{w},b} \gamma_A\|\mathbf{w}\|^2 + \sum_{i=1}^{l_b}\sum_{j=1}^{n_i} \ell(y_{ij}, (\mathbf{w}^T\mathbf{h}(\mathbf{x}_{ij}) + b)) \quad (5.8)$$

$$\text{subject to } y_{ij} = -1,\ \forall i : Y_i = -1, j = 1, \cdots, n_i,$$
$$Y_i \geq \frac{2 - n_i}{n_i},\ \forall i : Y_i = 1, \quad (5.9)$$

where $\gamma_A = \frac{1}{C} > 0$ is a free regularization parameter controlling the trade-off between error minimization and margin maximization, and $\ell(y_{ij}, t)$ is a hinge loss or soft-margin loss function. In our case, we slightly extend the formulation in Eq.(4.2) by replacing $\mathbf{h}(\mathbf{x}_{ij})$ with the responses of V kernels evaluations as $\mathbf{h}(\mathbf{x}_{ij}) = [f^{*(1)}(\mathbf{x}_{ij}), \cdots, f^{*(V)}(\mathbf{x}_{ij})]^T$ based on multiple feature representations of each instance. We then solve the problem in Eq.(4.2) using the same iterative optimization as proposed by Andrew et al. [14], detailed are presented in 2.3.1.

The final binary decision function of miSVM is of the following form

$$F_{miSVM}^{*}(\mathbf{x}) = sgn\left(\mathbf{w}^{*T}\left[\tilde{\mathbf{K}}^{(1)}(\mathbf{x})^T\boldsymbol{\alpha}^{*(1)}, \cdots, \tilde{\mathbf{K}}^{(V)}(\mathbf{x})^T\boldsymbol{\alpha}^{*(V)}\right]\right). \quad (5.10)$$

In the multi-stage system, we use this classifier for ranking images when trained as image category classifier, and the next stage for scoring of subregions in an image when trained as sub-region classifier.

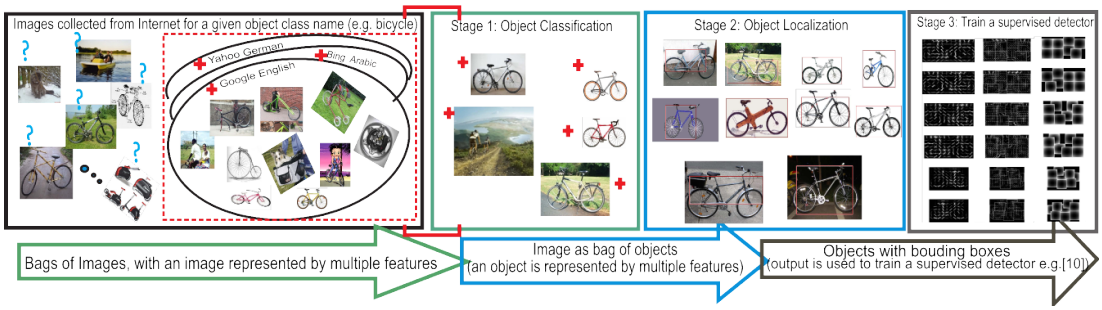


Figure 5.1: Three-stage framework- for training an object detector with example images collected from the Internet searches: (1) object classification, (2) object localization, and (3) Training a supervised part-based object detector.

5.3 Object Classification and Localization

To learn an object detector from weakly-labeled Internet images. We created a three-stage framework, which is illustrated in Figure 5.1, Object classification, Object localization and Training a supervised part-based object detector.

5.3.1 Stage 1: Object Classification

In this stage, first bags of images are collected for a given object's class name from the Internet and then a binary image classifier is trained with these bags of images. The final classifier in then used to rank the images.

There is an ambiguity that the queried image may not contain the object-of-interest. Therefore, we explicitly model this ambiguity by collecting bags of images via a number of different image search engines (i.e., Yahoo, Google and Bing). Such bags of images can be obtained by translating the given object's name from English into 12 different languages, e.g., English, Spanish, German, Arabic, and French, using an automatic translation tool[†]. The details for collection of bags of images, are presented in Chapter 4.

Using these bags of images, we trained a binary object category based image classifiers. The classifiers are used to rank the collected images based on the confidence scores. We threshold the confidence scores for the next stage of our framework and consider highly confident images as labeled and less confident ones as unlabeled

[†]http://www.google.com/language_tools?hl=en

images. This helps us, to some extent, to prevent the propagation of noise from one stage to the other.

5.3.2 Stage 2: Object Localization

In order to train a classifier as an object detector, we need to know the location of the target object in the training images, which may contains more than one objects of different or same classes. Such an ambiguity can also be cast as a MIL problem, where each image can be considered as a bag containing its sub-regions. Therefore, a positive bag ensures that at least one sub-region covers the target object. We need some reasonable guess about the object's location so that we can group these sub-regions to form a bag. There are a number of approaches available to produce a set of possible locations of an object in an image: the objectness measure proposed by Alexe et al. [67], the saliency detection proposed by Bruce et al. [68], the hierarchical segmentation proposed by Arbeláez et al. [69], and the method proposed by Endres et al. [70]. Here, in particular, we use the objectness measure [67] and the hierarchical segmentation [69] to represent an image by a group of sub-regions.

5.3.3 Stage 3: Training a Supervised Object Detector

It has been shown that part-based models such as [66] yield state-of-the-art results in various object detection tasks. However, these models need bounding boxes for the objects in training images. In our case we take the output of Stage 2 and train a detector using the Latent SVM part-based model [66]. This detector is then used to detect objects in any test image[‡].

5.4 Experiments

To demonstrate the proposed system, we give a detailed experimental analysis of the framework based on two publicly available datasets, i.e., on the ETHZ [72] and the PASCAL VOC [73] benchmark datasets. We show that our framework allows

[‡]Please note that we are aware that recently approaches such as Nguyen et al. [71] have been proposed, which could directly been trained on the output of Stage 2

us to train an object detector without having any visually labeled image data. We first substantiate the benefits of using multiple features on the ETHZ dataset and then give a detailed analysis of the full system on the PASCAL dataset.

5.4.1 ETHZ Test Dataset

The ETHZ test dataset was mainly built for object localization based on shape features. It contains five object classes (apple-logos, bottles, giraffes, mugs, and swans), totally a number of 255 images. The total number of object instances is 289, as in some images objects appear multiple times. The data set is highly challenging with a high intra class variability and the objects appear in various scales. Most of the images are photographs but there are also some drawings and paintings, where in majority of the images the target object occupies only a small fraction of the image.

In the following, we mainly want to demonstrate that using multiple features in parallel can significantly increase the object detection rate compared to the baseline approach [9]. For this experiment, we train MIL-based binary object detectors using the images collected by Stage 1 of our framework. We use the hierarchical segmentation [69] to decompose an image into a bag of 100 sub-regions. Each sub-region is represented by separate PHOG and PHOW descriptors for our model; for the baseline method the feature vectors are concatenated. The final detection results are shown in Figure 5.2.

We show the results obtained by the original sMIL approach and by our slightly adapted approach. The performance is measured by the detection rate against the number of false positives per image (FPPI) averaged over all 255 test images. A detection is correct if the intersection-over-union ratio with the ground truth bounding box is greater than 50%. The plots show that the proposed method performs well on all classes compared to the baseline at the moderate false-positive rate of 0.5 FPPI as a reference point. This shows that using multiple features in parallel can improve the detection results.

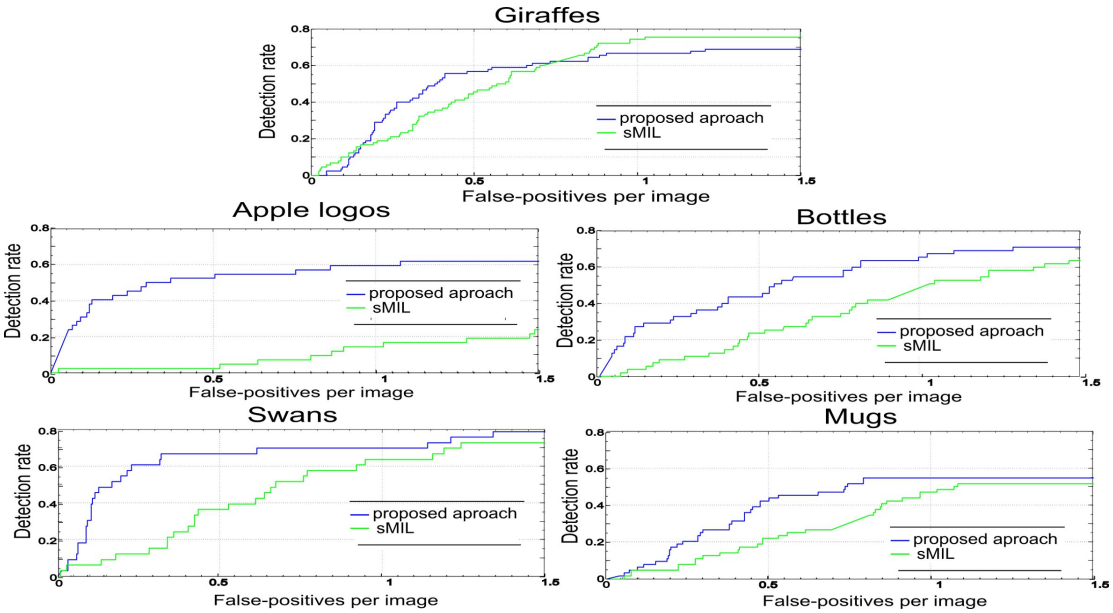


Figure 5.2: Object detection results on ETHZ dataset based on the detection-rate under 50% intersection-over-union criterion. The results compares the detection rates, which are trained using the sMIL and the proposed multi-features extension to MIL methods, (see 5.2.2 for details).

5.4.2 PASCAL VOC Test Dataset

For this experiment, we use a subset of the object classes from the PASCAL VOC 2007 dataset. In particular, we select mainly compact objects where fully supervised methods perform reasonably well [73]. The selected objects are horse, airplane, bicycle, motorbike, bus, train, boat, and car. We selected PASCAL VOC 2007 because this dataset has been used for other state-of-the object detection evaluations and the dataset is still widely used for evaluation [74].

In order to decompose an image into a bag of objects in Stage 2, we sample 100 windows from the 10000 windows per image generated using the publicly available code for OM [67]. Based on this input (labeled and unlabeled bags of objects) we train a MIL-based binary object localization model and use it to detect the target objects. Illustrative examples are shown in Figure 5.3. Finally in Stage 3, this output is used to train a fully supervised part-based detector (LSVM-part-based) [66].

The localization model learned in Stage 2 can also be used to detect objects without training the LSVM-part-based detector. Therefore, to show the possible

Method	bike	mbike	bus	aero	horse	car	train	boat	mAP
Stage 2	30.1	27.5	29.7	9.5	6.0	29.4	8.3	2.4	17.9
LSVM + OM	31.0	28.6	30.8	7.1	4.2	32.3	8.4	2.4	18.1
LSVM + Stage 3	39.6	34.5	36.7	11.8	7.0	34.2	11.9	3.4	22.4
LSVM-part-based [66]	59.5	48.7	49.6	28.9	56.8	57.9	45.1	15.2	45.2
Best VOC2007 [73]	40.9	37.5	39.3	26.2	33.5	43.2	45.3	9.4	34.4
MKL Detector [5]	47.8	45.5	50.7	37.6	51.2	50.6	45.3	15.3	43.0

Table 5.1: Object detection results, AP in percent, for different methods in the seven of the PASCAL VOC 2007 [73] challenge categories.

performance gain, we also directly apply this model for object detection, providing a meaningful baseline. Additionally, it can be observed from Figure 5.3 that in the images fed to Stage 2 the target object is either the central focus or sometimes largely fills the frame. Hence, the objectness measure (OM) may produce a good bounding box around the object. Therefore, as a second baseline, we use the highest confident window per image generated by the OM as bounding box annotation to train the LSVM-part-based detector. The thus obtained results in comparison to other state-of-the-art methods are given in Table 5.1.

We give a comparison to the LSVM-part-based [66], the challenge winners (Oxford, INRIA_PlusClass and UoCTTI) [73], and an MKL-based approach [5]. In contrast to our approach, these methods follow the *comp3* protocol and thus are fully supervised using both, the image label and the location of the object during training. The detection performance for each object class is measured by the average precision (AP) on the entire PASCAL VOC 2007 test set (4952 images). Notice that the results for the LSVM-part-based model are taken from the currently improved results[§], where an object class is represented by a three component mixture of deformable part models.

It can be seen that the proposed method (Stage 3) outperforms the baselines for all object classes. But the performance in case of horse, train and boat is not satisfactory. The main reason is the diversity in appearance, shape, and scale of these objects, which our model does not capture very well using the standard representation. As expected, our model cannot outperform the state-of-the-art detectors, which use high quality annotations for training. However the results are still encour-

[§]http://people.cs.uchicago.edu/ pff/latent/

Figure 5.3: Object locations of Internet images obtained automatically by Stage 2: bicycle, motorbike, bus, aeroplane, horse, car, train and boat. Correct detections are shown on the left side and bad detections (the bounding boxes cover either only parts of the object or is much larger than the object size) on the right side.

aging, as we learn discriminative visual object models provided only with a weak textual prior.

5.5 Conclusions

In this chapter we proposed a system which is able to learn from weakly labeled or nosily labeled images downloaded from the Internet. In fact, the goal was to learn an complex part based object detector just by using a textual prior, i.e., not using any manually class-labeled images. Our system works in three stages. In the first stage, we determine the presence of a target object; in the second stage, we localize the object; finally, in the third stage, we learn a detector using a supervised method. Since in the two stages uncertainly labeled data has to be handled we apply an extend MIL method for the actual learning tasks. To demonstrate the benefits of the approach, we evaluated it on two benchmark datasets. We showed that we

are able to train a reasonable detector without using any manually labeled data, i.e., the training images are provided by a weak supervisor the Internet via images searches.

Chapter 6

Huberized LapSVM

The Laplacian Support Vector Machine [12] (LapSVM), introduced in Chapter 2, has shown state-of-the-art performance in semi-supervised classification. To overcome some issues of the original dual formulation, recently Mellacci et al. [15] have proposed an efficient method, the LapSVMp, for training the LapSVM in the primal with preconditioned conjugate gradient. This reduces the training time and allows for a fast computation with approximately the same classification accuracy as the original one. However, LapSMp uses a squared hinge loss function for the labeled examples, which is not twice differentiable and penalizes noisy labeled examples too much. The accuracy of such an approach may decrease in situations when the training data contains outliers or the labeled data is heavily contaminated by noise. One of such scenario, as explained in Chapter 1, is learning models from Internet images for object categorization, since the true class label of the images collected could be ambiguous and the desired object may not be present in the queried images.

This chapter, presents the detailed formulations for training the LapSVM problem in the primal with a continuously differentiable Huber hinge loss function. The resulting method is called the Huberized LapSVMp. The Huber hinge loss gives a milder penalty than the squared hinge loss, making the proposed solution fit in situations when the available labeled training data is noisy.

Section 6.1 reviews the formulations for solving the LapSVM problem in dual and also highlight the benefits of solving this problem directly in the primal. Section 6.1, presents in detail the Huberized hinge loss function in comparison with the linear

and squared hinge loss functions. Section 6.3 provides the detailed formulations for training the Huberized LapSVMp in the primal, solving the learning problem with Newtons' and preconditioned conjugate gradient method. Section 6.4 presents the experimental analysis of the proposed method on a number of datasets, specially the PASCAL VOC 2007 where the Huberized LapSVMp outperform the winning scores. This section also presents experimental analysis for automatic learning of object category classifiers. Finally, Section 6.5 concludes this chapter.

6.1 Laplacian Support Vector Machines

The LapSVM, discussed in Chapter 2, is a straight forward extension of the supervised SVM learning to semi-supervised learning. It is based on a manifold assumption, which states that the data lies on low dimensional manifolds in a high dimensional input feature space where each manifold may represent a single class. It implies that if two points that are close with respect to the geodesic distances on a manifold $\mathcal{M}$ then their labels should be the same, similar in the sense that the conditional probability distribution $P(y|\mathbf{x})$ between such two points should change smoothly along the manifolds. Since the true $\mathcal{M}$ is unknown [12], the intrinsic norm is estimated from the point cloud of labeled and unlabeled points using the graph Laplacian.

The Graph Laplacian is a discrete approximation to a weighted Laplace-Beltrami operator on a manifold, which has numerous geometric properties and induces a smoothness functional. The LapSVM uses a local neighborhood adjacency graph to build the graph Laplacian [75] to model the high-dimensional data lying on a low dimensional manifold. Given a training dataset $\mathcal{S}$, the adjacency matrix $\mathbf{W} \in \mathbb{R}^{(l+u)\times(l+u)}$ of the widely used kNN graph is defined as

$$\mathbf{W}_{ij} = \begin{cases} exp(-\frac{D(\mathbf{x}_i,\mathbf{x}_j)}{\sigma}) & i \in N_j \text{ or } j \in N_i \\ 0 & \text{otherwise} \end{cases}, \tag{6.1}$$

where $D()$ is a pairwise distance measure in the input feature space $\mathbb{R}^d$ such as euclidean distance or χ^2 distance; $N_i \subset [1, \cdots, l+u]$ consists of k indexes of k nearest neighbors of instance $\mathbf{x}_i$ in $\mathcal{S}$; the scaling factor σ is set to the mean of the

k nearest neighbors pairwise distances. An edge between two vertexes, $\mathbf{W}_{ij} > 0$, represents the similarity of two instances . A larger edge weight means that the labels or class of the two connecting instances are the same. The graph Laplacian, which is a positive semi-definite operator, is defined in terms of this adjacency matrix as $\mathbf{L} = \left(\mathbf{D}^{-\frac{1}{2}}\mathbf{W}\mathbf{D}^{-\frac{1}{2}}\right)^p$, where $\mathbf{D}$ is a diagonal matrix whose diagonal element are given by $\mathbf{D}_{ii} = \sum_{j=1}^{l+u} \mathbf{W}_{ij}$ and $p \geq 1$ is the degree of the graph Laplacian.

The LapSVM algorithm learns from the manifold structure of the input feature space and uses it as an additional regularization term in the supervised SVM learning process. Assuming that the training examples, $\mathcal{S} = \mathcal{L} + \mathcal{U}$, are available in an ordered form in such away with the first l instances are labeled and the next u are unlabeled. Each instance $\mathbf{x}_i$ is associated with a label $y_i \in \{+1, -1, 0\}$, if $y_i \in \{+1, -1\}$ otherwise it is unlabeled. In terms of a kernel $\mathbf{K} \in \mathbb{R}^{(l+u)\times(l+u)}$, the primal object functional to be minimized is defined as

$$\min_{\boldsymbol{\alpha}\in\mathbb{R}^{(l+u)}} \frac{1}{l}\sum_{i=1}^{l} \ell_h(y_i, \mathbf{K}_i^T\boldsymbol{\alpha}) + \gamma_A\boldsymbol{\alpha}^T\mathbf{K}\boldsymbol{\alpha} + \frac{\gamma_I}{(l+u)^2}\sum_{i,j=1}^{l+u} \mathbf{W}_{ij}(\mathbf{K}_i^T\boldsymbol{\alpha} - \mathbf{K}_j^T\boldsymbol{\alpha})^2 \quad (6.2)$$

or equivalently as

$$\min_{\boldsymbol{\alpha}\in\mathbb{R}^{(l+u)}} \frac{1}{l}\sum_{i=1}^{l} \ell_h(y_i, \mathbf{K}_i^T\boldsymbol{\alpha}) + \gamma_A\boldsymbol{\alpha}^T\mathbf{K}\boldsymbol{\alpha} + \frac{\gamma_I}{(l+u)^2}\boldsymbol{\alpha}^T\mathbf{K}\mathbf{L}\mathbf{K}\boldsymbol{\alpha} \quad (6.3)$$

where $\ell_h(y, t)$ is the linear Hinge loss function: $\max(0, 1 - yt)$. The key property is that this loss function is linear for outliers and noisy labeled examples which encourages the classification model to balance the number of outliers per class by treating all outliers equally; γ_A and γ_I are regularization parameters. The former controls the smoothness of the decision functional f in the associated RKHS and the latter controls its penalization along the low dimensional manifold structure modeled via the graph Laplacian. The normalization constant $\frac{1}{(l+u)^2}$ is the scaling factor for the empirical estimate of $\mathbf{L}$ [12].

With the extended version of the Representer Theorem (for proof, c.f.[12]), the minimizer of the problem in (6.3) is unique and is the linear combination of kernel

functions centered on the training data:

$$f^*(\mathbf{x}) = \sum_{i=1}^{l+u} \alpha_i^* k(\mathbf{x}_i, \mathbf{x}). \tag{6.4}$$

This real valued function is then used for binary classification.

In the original work of Belkin et al. [12] the proposed solution for LapSVM is based on the dual form derived from (6.3) using standard Lagrange Multiplier techniques in a similar way used for a supervised SVM. Since, the difficulty with the linear Hinge loss is that direct optimization is difficult due the discontinuity in the derivative of the loss function at $y_i f(\mathbf{x}_i) = 1$. The training of LapSVM in the dual involves two steps. First using a QP solver, we solve the quadratic program in l dual variables $\boldsymbol{\beta}$:

$$\begin{aligned} \boldsymbol{\beta}^* &= \max_{\boldsymbol{\beta} \in \mathbb{R}^l} \sum_{i=1}^{l} \beta_i - \frac{1}{2} \boldsymbol{\beta}^T \mathbf{Q} \boldsymbol{\beta} \\ &\text{subject to: } \sum_{i=1}^{l} \beta_i y_i = 0, \\ &0 \leq \boldsymbol{\beta}_i \leq \frac{1}{l} \quad i = 1, \cdots, l \quad , \end{aligned} \tag{6.5}$$

where

$$\mathbf{Q} = \mathbf{YJK}(2\gamma_A \mathbf{I} + 2\frac{\gamma_I}{(l+u)^2}\mathbf{LK})^{-1}\mathbf{J}^T\mathbf{Y}; \tag{6.6}$$

$\mathbf{Y} \in \mathbb{R}^{(l)\times(l)}$ is a diagonal matrix with $\mathbf{Y}_{ii} = y_i$, a matrix $\mathbf{J} \in \mathbb{R}^{l\times(l+u)}$ is defined as $\mathbf{J} = [\mathbf{I} \quad \mathbf{0}]$, where $\mathbf{I} \in \mathbb{R}^{l\times l}$ is an identity matrix, and $\mathbf{0} \in \mathbb{R}^{l\times u}$ is a zero matrix. In the second step, the optimal expansion coefficients $\boldsymbol{\alpha}^*$ that define the decision function f^* in (6.4) are obtained by solving the linear system involving the l dual variables:

$$\boldsymbol{\alpha}^* = (2\gamma_A \mathbf{I} + 2\frac{\gamma_I}{(l+u)^2}\mathbf{LK})^{-1}\mathbf{J}^T\mathbf{Y}\boldsymbol{\beta}^*. \tag{6.7}$$

When $\gamma_I = 0$, LapSVM ignores the unlabeled data and becomes similar to that of

a supervised SVM problem defined in (2.21). The final binary decision function for the LapSVM is of the following form:

$$F_{LapSVM}(\mathbf{x}) = sgn(\mathbf{K}(\mathbf{x})^T \boldsymbol{\alpha}^*). \tag{6.8}$$

However, the inversion of matrices in (6.7) and (6.6) leads to complexity $O((l+u)^3)$ and the inversion can be expensive when $(l+u)$ is large. A variety of techniques have been developed which address issues related to scalability such as [76] and efficiency such as [15]. We are more interested in an efficient and good approximate solution for the LapSVM which is obtained by training it in the primal [15] and the approach is closely related to this work.

6.1.1 The LapSVMp: Training LapSVM in the Primal

The primal and dual are two ways to solve the same optimization problem. In general, there are two main reasons to solve the problem in the dual. First, the duality provides a simple way to deal with the constraints (by introducing Lagrangian variables or dual for each constraint). Secondly, the dual problem can be written in terms of dot products, thereby making it possible to use kernel functions. The motivation behind the primal is that it directly minimizes the primal variables we are interested in, its implementation does not require any commercial solver or particular complex libraries, and in some cases it may be faster to converge [38]. The training of supervised SVM in the primal have been extensively studied and analyzed by a number of authors such as [39] or [38]. In contrast, Mellacci et al. [15] are the only ones who recently explored training LapSVM in the primal.

Specifically for solving the LapSVM, it has been shown in [15] that the unconstrained primal optimization is more efficient than the dual. Firstly, it allows us to solve the original problem without the need of the computations related to variables switching, i.e., it directly manipulates the primal kernel expansion co-efficients $\boldsymbol{\alpha}$ of f without passing through the dual $\boldsymbol{\beta}$ ones. Secondly, the computational cost for solving the problem is reduced from $O((l+u)^3)$ to $O(k(l+u)^2)$ where k is significantly smaller than $(l+u)$ [15].

To minimize the objective function (6.3) by gradient descent, it is necessary

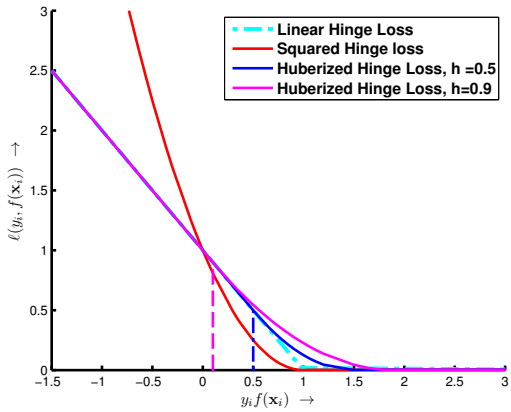


Figure 6.1: Loss Functions: Linear hinge loss $\ell_h(y_i, f(\mathbf{x}_i)) = \max(0, 1 - y_i f(\mathbf{x}_i))$, Squared hinge loss $\ell_{sqh}(y_i, f(\mathbf{x}_i)) = \max(0, 1 - y_i f(\mathbf{x}_i))^2$, and Huberized Hinge loss (6.10) for $h = 0.5$ and $h = 0.9$. When $h \to 0$ the Huberized hinge loss looks more the Linear hinge loss.

for the loss function to be continuously differentiable with respect to the second argument. Note that this is an unconstrained optimization problem. For the labeled examples, Mellacci et al. [15] have used the squared hinge loss function, $\ell_{sqh}(y_i, f(\mathbf{x}_i))^2 = \max(0, 1 - y_i f(\mathbf{x}_i))^2$, since it makes the LapSVM problem differentiable in f so in $\boldsymbol{\alpha}$. After substituting $\ell = \ell_{sqh}$ in (6.3), the LapSVMp problem is defined as

$$\min_{\boldsymbol{\alpha} \in \mathbb{R}^{(l+u)}} \frac{1}{2l} \sum_{i=1}^{l} \ell_{sqh}(y_i, \mathbf{K}_i^T \boldsymbol{\alpha})^2 + \gamma_A \boldsymbol{\alpha}^T \mathbf{K} \boldsymbol{\alpha} + \frac{\gamma_I}{(l+u)^2} \boldsymbol{\alpha}^T \mathbf{K} \mathbf{L} \mathbf{K} \boldsymbol{\alpha}. \tag{6.9}$$

It is solved by preconditioned conjugate gradient (PCG) method [77] the training with PCG is presented in Section 6.3. To get a fast solution, PCG is coupled with an early stopping criterion based on the stability of the classifier decision. In this chapters refers the final classifier by the LapSVMp.

6.2 Huberized Hinge Loss Function

The unconstrained objective function (6.3) can be solved with PCG for various loss functions for the labeled examples, Figure 6.1 shows some example loss functions. However, one would like to use a loss function that is more robust to possible outliers or noise in the training data, in order to improve the classification accuracy and at

the same time keep the solution efficient. The standard linear hinge loss function is not differentiable due to the kink in $\ell_h(y_i, f(\mathbf{x}_i)) = \max(0, 1-y_i f(\mathbf{x}_i))$ at $y_i f(\mathbf{x}_i) = 1$, although it is convex and robust to noise or outliers. While the squared hinge loss which is differentiable (although not twice differentiable) may not be well suited in situations with possible contamination of the labeled training data. It penalizes noisy examples too much, since it rises quadratically on the left tail as shown in Figure 6.1.

Inspired by the robust Huber loss [78], Chapelle [38] has proposed a loss function called the Huberized hinge loss for the standard supervised SVM. It is the differentiable approximation of the hinge loss function and is defined as

$$\ell_{hubh}(y_i, f(\mathbf{x}_i)) = \begin{cases} 0 & \text{if } y_i f(\mathbf{x}_i) > 1 + h \\ \frac{(1+h-y_i f(\mathbf{x}_i))^2}{4h} & \text{if } |1 - y_i f(\mathbf{x}_i)| \leq h \\ (1 - y_i f(\mathbf{x}_i)) & \text{if } y_i f(\mathbf{x}_i) < 1 - h \end{cases}, \tag{6.10}$$

where $h \geq 0$ is a parameter whose values can bet set between 0.01 and 0.9. This loss function has both a quadratic and a linear parts as shown in Figure 6.1. if $h \to 0$ then the loss function approaches to linear Hinge loss . The important property of this loss function is that it gives a milder penalty than the squared Hinge loss does. This makes the LapSVM robust to label noise or to outliers present in the labeled part of training examples. We solve the LapSVM problem in the primal with the Huber hinge loss function.

6.3 Training Huberized LapSVM in the Primal

This section presents the primal training algorithm of the Huberized LapSVM classifier along the lines of Melacci et al. [15] using both PCG and Newton's method. Instead of using the squared squared hinge loss, we use the Huberized hinge loss function (6.10) to measure the misclassification. Note that we are not minimizing the hinge loss, since from the machine learning point of view there is no reason to prefer hinge or square hinge loss anyway. However, in our eyes the classification accuracy of LapSVM classifier can be improved after introducing Huber hinge loss, since it is more resistance to labeled noise or outliers or possible contamination in the

labels during training. By substituting $\ell = \ell_{hubh}$, the unconstrained optimization problem (6.3) can be written as

$$\Omega(\boldsymbol{\alpha}) = \gamma_A \boldsymbol{\alpha}^T \mathbf{K}\boldsymbol{\alpha} + \frac{\gamma_I}{(l+u)^2}\boldsymbol{\alpha}^T \mathbf{KLK}\boldsymbol{\alpha} + \frac{1}{l}\sum_{i=1}^{l} \ell_{hubh}(y_i, \mathbf{K}_i^T \boldsymbol{\alpha}). \tag{6.11}$$

It can be minimized by a variety of optimization techniques such as Newton's optimization method as well by PCG method. Newton's method should be the first choice for the unconstrained problem, since the objective function is strictly convex, twice continuously differentiable and piece wise quadratic. However, it requires an inversion of a large matrix in every iteration leading to a complexity of $O(n^3)$. We, minimized the objective functional (6.11) to obtain the solution $\boldsymbol{\alpha}^*$, with PCG which is more efficient method in finding a good approximate solution and considerably reduce the training time.

6.3.1 Newton's Optimization

Starting from some initial value for $\boldsymbol{\alpha}$, each Newton step consists of the following update:

$$\boldsymbol{\alpha}^t = \boldsymbol{\alpha}^{t-1} - s\mathbf{H}^{-1}\nabla, \tag{6.12}$$

where t is the iteration number, s is the step size, and ∇ and $\mathbf{H}$ are the gradient and the Hessian of the objective function (6.11).

It is important to introduce the concept of error vectors [38], before presenting the detail formulations of Newton's method. A labeled point $\mathbf{x}_i$ is an error vector if a loss function on this point generates a non zero value. For a given $\mathcal{L}$, the Huber hinge loss defined in Equation (6.10) generates two sets of error vectors, since it penalizes the given point either quadratically and linearly. It is worth to remind that we have assumed that the training examples $\mathcal{S}$ are available in a pre-ordered form such that the first l are labeled and the next u are unlabeled points. The loss function does not penalize all the remaining labeled points for which its value is greater than zero, since for a given value of h the classifier $f(\mathbf{x_i})$ generates such that $y_i f(\mathbf{x}_i) > 1 + h$ *. Let $\mathcal{E}_{qd}$ be a set of error vectors which fall in the quadratic part

*In the case of supervised SVMs the errors vectors are equivalent to the support vectors, however

and $\mathcal{E}_{lin}$ be the set of error vectors which fall in the linear part of the Huber hinge loss function.

The gradient of (6.11) with respect to $\boldsymbol{\alpha}$ is given by

$$\nabla = (\nabla_{\boldsymbol{\alpha}}) = 2\gamma_A \mathbf{K}\boldsymbol{\alpha} + \frac{2\gamma_I}{(l+u)^2}\mathbf{K}\mathbf{L}\mathbf{K}\boldsymbol{\alpha} + \frac{1}{l}\sum_{i=1}^{n}\mathbf{K}_i \frac{\partial \ell}{\partial t}(y_i, \mathbf{K}_i^T\boldsymbol{\alpha}), \tag{6.13}$$

where $\frac{\partial \ell}{\partial t}$ is the first order partial derivative of the Huberized hinge loss function (6.10) with respect to the second argument and is equal to

$$\frac{\partial \ell}{\partial t}(y_i, \mathbf{K}_i^T\boldsymbol{\alpha}) = \frac{2}{4h}y_i(y_i\mathbf{K}_i^T\boldsymbol{\alpha} - (1+h)) - y_i. \tag{6.14}$$

By substituting (6.14) into (6.13) we obtain

$$\begin{aligned}\nabla \quad &= 2\gamma_A\mathbf{K}\boldsymbol{\alpha} + \tfrac{2\gamma_I}{(l+u)^2}\mathbf{K}\mathbf{L}\mathbf{K}\boldsymbol{\alpha} + \tfrac{2}{4hl}\mathbf{K}\mathbf{I}_e^0(\mathbf{K}\boldsymbol{\alpha} - (1+h)\mathbf{y}) - \tfrac{1}{l}\mathbf{K}\mathbf{I}_e^1\mathbf{y} \\ &= 2\gamma_A\mathbf{K}\boldsymbol{\alpha} + \tfrac{2\gamma_I}{(l+u)^2}\mathbf{K}\mathbf{L}\mathbf{K}\boldsymbol{\alpha} + \tfrac{2}{4hl}\mathbf{K}\mathbf{I}_e^0\mathbf{K}\boldsymbol{\alpha} - \mathbf{K}\left(\tfrac{2(1+h)}{4hl}\mathbf{I}_e^0 + \tfrac{1}{l}\mathbf{I}_e^1\right)\mathbf{y}.\end{aligned} \tag{6.15}$$

Here $\mathbf{y} \in \{-1, 0, 1\}$ is a vector whose components represent the labels of the training data, whereas $\mathbf{I}_e^0$ and $\mathbf{I}_e^1$ are two diagonal matrices each of size $n \times n$. In $\mathbf{I}_e^0$ the corresponding entries for points which are in $\mathcal{E}_{qd}$ are set to one and zeros for all others points. Whereas in $\mathbf{I}_e^1$ the corresponding entries for points which are in $\mathcal{E}_{lin}$ are set to one and zero for all others.

The Hessian $\mathbf{H}$ is given by

$$\mathbf{H} = (\nabla_{\boldsymbol{\alpha}^2}) = \left(2\gamma_A\mathbf{K} + \frac{2\gamma_I}{(l+u)^2}\mathbf{K}\mathbf{L}\mathbf{K} + \frac{2}{4hl}\mathbf{K}\mathbf{I}_e^0\mathbf{K}\right). \tag{6.16}$$

Substituting (6.16) and (6.15) in (6.12), the update for each Newton's step becomes

$$\begin{aligned}\boldsymbol{\alpha}^t &= (1-s)\boldsymbol{\alpha}^{t-1} + s\left(2\gamma_A\mathbf{K} + \frac{2\gamma_I}{(l+u)^2}\mathbf{K}\mathbf{L}\mathbf{K} + \frac{2}{4hl}\mathbf{K}\mathbf{I}_e^0\mathbf{K}\right)^{-1}\mathbf{K}\left(\frac{2(1+h)}{4hl}\mathbf{I}_e^0 + \frac{1}{l}\mathbf{I}_e^1\right)\mathbf{y} \\ &= (1-s)\boldsymbol{\alpha}^{t-1} + s\mathbf{K}^{-1}\left(2\gamma_A\mathbf{I} + \frac{2\gamma_I}{(l+u)^2}\mathbf{L}\mathbf{K} + \frac{2}{4hl}\mathbf{I}_e^0\mathbf{K}\right)^{-1}\mathbf{K}\left(\frac{2(1+h)}{4hl}\mathbf{I}_e^0 + \frac{1}{l}\mathbf{I}_e^1\right)\mathbf{y} \\ &= (1-s)\boldsymbol{\alpha}^{t-1} + s\left(2\gamma_A\mathbf{I} + \frac{2\gamma_I}{(l+u)^2}\mathbf{L}\mathbf{K} + \frac{2}{4hl}\mathbf{I}_e^0\mathbf{K}\right)^{-1}\left(\frac{2(1+h)}{4hl}\mathbf{I}_e^0 + \frac{1}{l}\mathbf{I}_e^1\right)\mathbf{y}.\end{aligned} \tag{6.17}$$

in case of the LapSVM all the points $\mathcal{S}$ contribute to the expansion of f.

The step s must be found with exact line search or backtracking [79], using one dimensional minimization of (6.11) restricted to the ray from $\boldsymbol{\alpha}^{t-l}$ to $\boldsymbol{\alpha}^{t}$. By setting $s = 1$ in our experiments did not result in any convergence problem. Convergence, which take not more than a few iterations, is declared when the number of linear and quadratic error vectors do not change any more. The complexity with the Newton's method is $O(n^3)$ due matrix inversion in the update rule. Note, we have assumed that $\mathbf{K}$ (and thus the Hessian) is invertible and non-singular, otherwise the optimal solution f will not not be unique and will produce one of the possible solution of expansion of (6.11).

6.3.2 Preconditioned Conjugate Gradient Method

In the following, we present the formulation of training of the Huberized LapSVM with PCG [77]. To avoid costly Newton's update we can directly minimize (6.11) by PCG method, a fast iterative method for solving a large system of equations. Following [15], we solve the primal LapSVM with huberized hinge loss by PCG, since PCG allows us to quickly compute approximate solutions, considerably reducing the training time.

Looking to (6.15) and (6.16), we can also write the gradient as

$$\nabla = \mathbf{H}\boldsymbol{\alpha} - \mathbf{K}\left(\frac{2(1+h)}{4hl}\mathbf{I}_e^0 + \frac{1}{l}\mathbf{I}_e^1\right)\mathbf{y}. \tag{6.18}$$

Setting the gradient $\nabla = 0$ in (6.18), we must compute the solution vector $\boldsymbol{\alpha}$ by solving a system of equations $\mathbf{H}\boldsymbol{\alpha} = \mathbf{c}$,

$$\begin{aligned} \nabla &= \mathbf{H}\boldsymbol{\alpha} - \mathbf{K}\left(\frac{2(1+h)}{4hl}\mathbf{I}_e^0 + \frac{1}{l}\mathbf{I}_e^1\right)\mathbf{y} = 0 \\ \Longrightarrow \mathbf{H}\boldsymbol{\alpha} &= \mathbf{K}\left(\frac{2(1+h)}{4hl}\mathbf{I}_e^0 + \frac{1}{l}\mathbf{I}_e^1\right)\mathbf{y} \\ \Longrightarrow \mathbf{H}\boldsymbol{\alpha} &= \mathbf{c}, \end{aligned} \tag{6.19}$$

where $\mathbf{c} = \mathbf{K}\left(\frac{2(1+h)}{4hl}\mathbf{I}_e^0 + \frac{1}{l}\mathbf{I}_e^1\right)\mathbf{y}$. To find the solution, the first choice is to use an iterative conjugate gradient CG method. However, the convergence rate of CG depends on the condition number κ of the Hessian matrix $\mathbf{H}$ defined in (6.16). A

condition number defines the ratio of the maximum to the minimum eigenvalues of a matrix and and is always greater or equal to one, $\kappa \geq 1$. A larger value of κ leads to a slower convergence rate of CG and vice versa (c.f. [77]). The presence of the terms $\mathbf{KLK}$ and $\mathbf{KI}_e^0\mathbf{K}$ in $\mathbf{H}$ leads to a not so well conditioned system of equations, resulting in large or bad condition number of $\mathbf{H}$. So, optimizing on $\boldsymbol{\alpha}$ with CG will results in a slower convergence rate. The PCG provides an easy easy fix to this problem: preconditioning by a matrix $\mathbf{P}$.

The PCG is an iterative method that indirectly solves an initial systems of equations (6.19) by solving

$$\hat{\mathbf{H}}\boldsymbol{\alpha} = \hat{\mathbf{c}} \tag{6.20}$$

where $\hat{\mathbf{H}} = \mathbf{P}^{-1}\mathbf{H}$ and $\hat{\mathbf{c}} = \mathbf{P}^{-1}\mathbf{c}$. This implies that preconditioned gradient $\hat{\nabla} = \hat{\mathbf{H}}\boldsymbol{\alpha} - \hat{\mathbf{c}}$. In general, PCG requires that $\mathbf{P}$ must be selected so that the condition number of (6.20) is improved with respect to (6.19). Moreover, $\mathbf{P}^{-1}\nabla$ must be computed efficiently [77].

Due to the quadratic form of the intrinsic regularizer we can follow a similar approach used in [38] for supervised nonlinear SVM. Looking to (6.19) we can take $\mathbf{K}$ as a factor of both $\mathbf{H}$ and $\mathbf{c}$:

$$\begin{aligned}\mathbf{K}\left(2\gamma_A\mathbf{I} + \frac{2\gamma_I}{(l+u)^2}\mathbf{LK} + \frac{2}{4hl}\mathbf{I}_e^0 K\right)\boldsymbol{\alpha} &= \mathbf{K}\left(\frac{2(1+h)}{4hl}\mathbf{I}_e^0 + \frac{1}{l}\mathbf{I}_e^1\right)\mathbf{y} \\ \Longrightarrow \mathbf{K}\hat{\mathbf{H}}\boldsymbol{\alpha} &= \mathbf{K}\hat{\mathbf{c}}. \end{aligned} \tag{6.21}$$

This also implies that $\hat{\mathbf{H}} = \left(2\gamma_A\mathbf{I} + \frac{2\gamma_I}{(l+u)^2}\mathbf{LK} + \frac{2}{4hl}\mathbf{I}_e^0\mathbf{K}\right)$. We select $\mathbf{P} = \mathbf{K}$ as preconditioned matrix. Note that we are assuming zero bias and $\mathbf{K}$ as non singular matrix otherwise a small ridge can be added to fix. The preconditioned gradient $\hat{\nabla} = \mathbf{P}^{-1}\nabla = \hat{\mathbf{H}}\boldsymbol{\alpha} - \hat{\mathbf{c}}$, can be efficiently calculated without explicitly performing any matrix inversion. This can be done by first computing the preconditioned gradient as $\hat{\nabla} = \hat{\mathbf{H}}\boldsymbol{\alpha} - \hat{\mathbf{c}}$ and then gradient vector as $\nabla = \mathbf{P}\hat{\mathbf{H}}\boldsymbol{\alpha} - \mathbf{P}\hat{\mathbf{c}}$. The condition number of the preconditioned system is decreased with respect to the one in (6.19), since $\mathbf{KLK}$ and $\mathbf{KI}_e^0\mathbf{K}$ are reduced to $\mathbf{LK}$ and $\mathbf{I}_e^0\mathbf{K}$. It is worth to note that the preconditioned $\hat{\mathbf{H}}$ is not a symmetric matrix, and it would not be possible to simply remove the $\mathbf{K}$ from both sides of (6.21) and solve it by CG method.

The pseudo-code for finding solution of the Huberized LapSVM problem by

means of PCG is given in Algorithm 2. We would like to point out that in this algorithm we have used the classical rules called Polak-Rieber (PR) [77] formula for the update of conjugate direction. This has been discussed in [77], that after several iterations the conjugacy of the descent direction may lost due to floating point error therefore a restart of the preconditioned algorithm is necessary. The restart is automatically performed when the update term becomes negative. The co-efficient ρ in the Algorithm 2 becomes zero and the following iteration corresponds to a steepest descent, as when PCG starts. For solving the LapSVM problem, generally the PR update is the best choice both for both convergence speed and numerical stability [15]. In our case we found that PR update is necessary due to the piece wise nature defined by the sets of error vectors.

Algorithm 2 PCG method for finding solution of the primal Huberized LapSVM problem

$h = 0.5, \boldsymbol{\alpha} = 0$ and $\mathbf{d}_{old} = \hat{\nabla}_{old} = -\mathbf{y}$
repeat
  Find s^* by the minimizing (6.11) on the line $\boldsymbol{\alpha} + s\mathbf{d}_{old}$.
  $\boldsymbol{\alpha} \leftarrow \boldsymbol{\alpha} + s^*\mathbf{d}$.
  Let $\mathcal{E}_{qd} = \{\mathbf{x}_i \in \mathcal{L} : |1 - \mathbf{K}_i\boldsymbol{\alpha}^T| \leq h\}$. Update $\mathbf{I}_e^0$
  Let $\mathcal{E}_{lin} = \{\mathbf{x}_i \in \mathcal{L} : \mathbf{K}_i\boldsymbol{\alpha}^T < 1 - h\}$. Update $\mathbf{I}_e^1$
  $\hat{\mathbf{H}} \leftarrow \left(2\gamma_A I + \frac{2\gamma_I}{(l+u)^2}\mathbf{LK} + \frac{2}{4hl}\mathbf{I}_e^0\mathbf{K}\right)$
  $\hat{\mathbf{c}} \leftarrow \left(\frac{2(1+h)}{4hl}\mathbf{I}_e^0 + \frac{1}{l}\mathbf{I}_e^1\right)\mathbf{y}$
  $\hat{\nabla}_{new} \leftarrow \hat{\mathbf{H}}\boldsymbol{\alpha} - \hat{\mathbf{c}}$.
  $\nabla \leftarrow \mathbf{K}\hat{\nabla}_{new}$
  $\rho \leftarrow \max(0, \left(\frac{\nabla^T(\hat{\nabla}_{new} - \hat{\nabla}_{old})}{\hat{\nabla}_{old}^T K \hat{\nabla}_{old}}\right))$
  $\mathbf{d}_{new} \leftarrow -\hat{\nabla}_{new} + \rho\mathbf{d}_{old}$
  $\hat{\nabla}_{old} = \hat{\nabla}_{new}$
  $\mathbf{d}_{old} = \mathbf{d}_{new}$
until Goal Conditions met

6.3.2.1 Line search

At each PCG iteration, we can use a one dimensional backtracking or an exact line search procedure to select the optimal step size s^* such that the objective function (6.11) along the line $\boldsymbol{\alpha} + s\mathbf{d}$ is minimized. Since the outside PCG iteration is based

on the approximation of the objective function, the inside line search procedure must be satisfactory enough that the selection of the step size s^* significantly reduces the final objection function. Inside each of the PCG iterations we have to solve

$$s^* = \underset{s \geq 0}{\operatorname{argmin}}\ \psi(s) = \min_{s \geq 0} \Omega(\boldsymbol{\alpha} + s\mathbf{d}), \tag{6.22}$$

to find an optimal step size s^*. Let us define a function $f_d(\mathbf{x}_i)$ that indicates the function $f(\mathbf{x})$ but whose coefficients are in $\mathbf{d} = [\mathbf{d}_{\boldsymbol{\alpha}}]$, that is, $f_d(\mathbf{x}_i) = \mathbf{K}_i^T \mathbf{d}_\alpha$ and $f_d = [f_d(\mathbf{x}_i), \mathbf{x}_i \in \mathcal{S}]$. Now given the search direction, we can rewrite the objective function (6.11) along the line $\boldsymbol{\alpha} + s\mathbf{d}$ as

$$\begin{aligned}
\psi(s) &= \gamma_A(2s\mathbf{d}_\alpha^T \mathbf{K}\boldsymbol{\alpha} + s^2 \mathbf{d}_\alpha^T \mathbf{K}\mathbf{d}_\alpha) + \frac{\gamma_I}{(l+u)^2}(2sf_d^T \mathbf{L}f + s^2 f_d^T \mathbf{L}f_d) + \\
&\frac{1}{4hl} \sum_{i \in \mathcal{E}_{eqd}} (1 + h - y_i(f(\mathbf{x}_i) + sf_d(\mathbf{x}_i)))^2 + \frac{1}{l} \sum_{i \in \mathcal{E}_{lin}} (1 - y_i(f(\mathbf{x}_i) + sf_d(\mathbf{x}_i))) \\
&= \gamma_A(2s\mathbf{d}_\alpha^T \mathbf{K}\boldsymbol{\alpha} + s^2 \mathbf{d}_\alpha^T \mathbf{K}\mathbf{d}_\alpha) + \frac{\gamma_I}{n^2}(2s\mathbf{d}_\alpha^T \mathbf{KLK}\boldsymbol{\alpha} + s^2 \mathbf{d}_\alpha^T \mathbf{KLK}\mathbf{d}_\alpha)) + \\
&\frac{1}{4hl} \sum_{i \in \mathcal{E}_{eqd}} (1 + h - y_i(f(\mathbf{x}_i) + sf_d(\mathbf{x}_i)))^2 + \frac{1}{l} \sum_{i \in \mathcal{E}_{lin}} (1 - y_i(f(\mathbf{x}_i) + sf_d(\mathbf{x}_i)))
\end{aligned} \tag{6.23}$$

Once the products $\mathbf{K}\boldsymbol{\alpha}, \mathbf{K}\mathbf{d}_\alpha, \mathbf{LK}\boldsymbol{\alpha}$ and $\mathbf{LK}\mathbf{d}_\alpha$ have been precomputed, this function $\psi(s)$ takes only $O(n)$ operations to evaluate. Such an iterative way for finding an optimal set size is necessary, since we have to deal with two sets of error vectors for different values of s. Due to its quadratic form we directly minimize this function by a one dimensional Newton's line search method, since the cost of each of the line search iterations per PCG step is negligible with respect to the $O(n^2)$ of a PCG iteration. However, other more efficient search methods such as proposed in [39] can also be used.

6.3.2.2 Goal condition to met

A number of choices for a common goal conditions for PCG are available and selecting a proper goal condition may enable us to discard unnecessary iterations in Algorithm 2 that may not lead to a significant improvement in the quality of the

classifier. In general, the convergence to an optimal or closest to an optimal solution is declared when the norm of the gradient $\|\nabla\|$ [79] or preconditioned gradient $\|\hat{\nabla}\|$ falls below a given threshold [38]. The relative decrease in the value of the objective function between connective iterations can also be checked or a maximum number of PCG iterations can be specified in advance. The adjustment parameters such as gradient threshold or maximum number of iterations, is extremely important both for reducing the training time and for the quality of the final classifier. However, such adjustment is strictly problem dependent and practically hard to find a trade-off between a good approximation and low number of iterations. Particularly in case of LapSVMs, either with squared Hinge loss or Huber hinge loss function, the surface of the objective function varies among different choices of parameters. Increasing or decreasing the values of γ_A and γ_I can lead to a less flat or more flat region around the optimal point. This has been thoroughly investigated (c.f. [15]), which is also true in our case, that fixing in advance the values of gradient threshold and maximum of iterations may either stop the PCG iterations far from the optimal solution or it may result an increase in the number of iterations without significant improvement in the quality of the final classifier. A good enough remedy to this problem is to use the stability check or the stability of the decision function $y(x) = sign(f(\mathbf{x})), \mathbf{x} \in \mathcal{U}$ on available unlabeled training data as an early stopping condition for the PCG iterations.

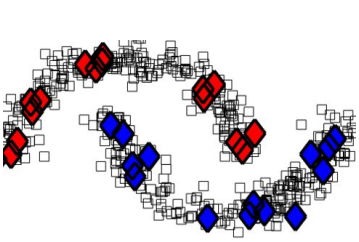
As we have discussed that the intrinsic norm or the manifold regularization term $\mathbf{f}^T\mathbf{L}\mathbf{f}$ enforces a soft constraint $f(\mathbf{x}_i) = f(\mathbf{x}_j)$ for near by points $\mathbf{x}_i$ and $\mathbf{x}_j$ along the underlying manifold. This results in a kind of graph transduction and hence allows the algorithm to diffuse the labels from points in $\mathcal{L}$ to the unlabeled data in $\mathcal{U}$ [15]. When this diffusion becomes complete and the classification hyperplane obtains a quite stable shape around the training data, the intrinsic norm will keep contributing to the gradient until a balance with respect to the ambient norm and to a loss function on error vectors is found. Compared to other described common goals, the early stop of the PCG iterations by stability check has been found in [15] to achieve accuracy similar to the optimal solution. Hence, we use the stability check for early stopping of the PCG Algorithm 2, whose pseudo-code is shown in Algorithm 3.

Algorithm 3 The stability check for PCG iterations [15]

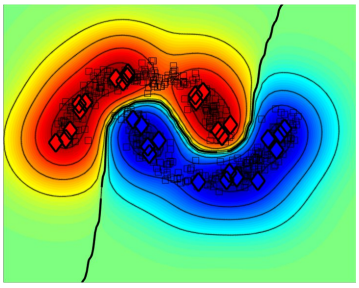
1: $\mathbf{d}_{old} = \mathbf{0} \in R^n, \eta \leftarrow 1.5\%, \theta \leftarrow \frac{\sqrt{n}}{2}$
2: Every θ iterations do the following:
3: $\mathbf{d}_{new} = [f(\mathbf{x}_j), \mathbf{x}_j \in \mathcal{U}, j = 1, \cdots, u]^T$
4: $\tau \leftarrow (100.\|\mathbf{d}_{new} - \mathbf{d}_{od}\|_1/u)\%$
5: **if** $\tau < \eta$ **then**
6: Stop PCG
7: **else**
8: $\mathbf{d}_{old} = \mathbf{d}_{new}$
9: **end if**

However, if the loss function is not robust to noise then the intrinsic norm may diffuse the noise in labeled points to the unlabeled points which will result in a bad quality of the final solution. To clear this intuition, we perform experiments on a 2D "two moons" dataset. This dataset depict two classes, data points on the upper moon belong class $+1$ and that on the lower moon belong to class -1. This is shown in Figure 6.2a, where the marked points are used for training LapSVMp and Huberized LapSVMp as labeled and unlabeled points. Similarly, the training dataset with noisy labels (shown in maginta color for which we switch their labels during training) is shown in Figure 6.2d. We train the Huberized LapSVM classifiers and the squared hinged LapSVMp [15] classifiers using these datasets. The regularization parameters γ_A and γ_I and the RBF kernel width $\tau = 0.35$ for both classifiers are set with the same values, these values were fixed after tunning the squared hinged LapSVM [15] on noiseless dataset. Similarly, the number of nearest neighbors is set to 6 for the construction of graph Laplacian L. The Figures 6.2b and 6.2c show the corresponding contour surfaces or decision boundaries of our Huberized LapSVMp and the squared hinged LapSVMp [15] classifiers, after training them with noiseless dataset. We can see that both classifiers correctly classify the two moons. In contrast, Figures 6.2e and 6.2f shows their corresponding final decision boundaries after training both the classifiers with the noisy two-moons 2D dataset, shown in Figures 6.2d. We can see that only the squared hinge LapSVM classifier wrongly diffuses the noise in labels to the unlabeled points, since the squared hinge loss penalizes the noisy labeled examples too much compared to Huberized loss.

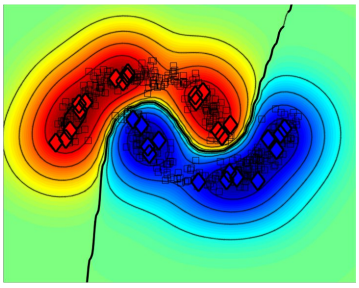
Another option that could be applied as PCG early stopping conditions, is to



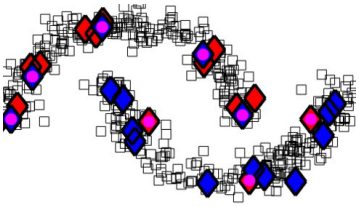
(a) The noise less "two moons" 2D data set, two classes upper moon (class +1) and lower moon (-1).



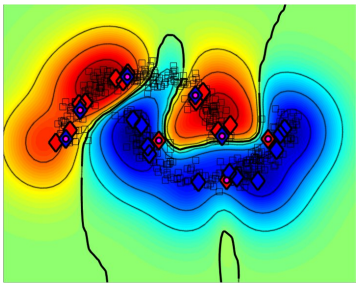
(b) A squared hinged LapSVM classifier trained with noise less data set using PCG algorithm, the final solution obtained with an early stopping stability check after 4 iterations (0% test error).



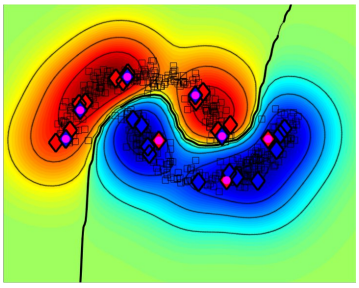
(c) A Huberized hinged LapSVM (for $h = 0.5$) classifier trained with noise less data set using PCG algorithm, the final solution obtained with an early stopping stability check after 4 iterations (0% test error).



(d) The "two moons" 2D data set with some noisy labeled points, two classes upper moon (class +1) and lower moon(-1).



(e) A squared hinged LapSVM classifier trained with noisy labeled data set using PCG algorithm,the final solution obtained with an early stopping stability check after 4 iterations (4.5% test error).



(f) A Huberized hinged LapSVM (for $h = 0.5$) trained with noisy labeled data set using PCG algorithm,the final solution obtained with an early stopping stability check after 4 iterations (**0**% test error).

Figure 6.2: (a) The "two moons" data set (500 points, 2 classes, 30 labeled points indicated with red and blue diamonds, 470 are unlabeled points indicated by black squares)-(d) the same data set but introduced noise in the labels,(8 points indicated by magenta colored circles are the noisy labeled points in the corresponding upper and lower moon classes)- (b and c) are the LapSVMs classifiers, trained with PCG using squared hinge loss and huber hinge loss functions respectively, the classifiers are trained with data set (a). (e and f) The LapSVMs classifiers trained with data set (d). The dark continuous line shows the decision boundary ($f(x) = 0$) and the confidence of the classifiers ranges from red ($f(x) \geq 1$) to blue ($f(x) \leq -1$).

use some labeled validation data set $\mathcal{V}$, whose pseudo-code is shown in Algorithm 4, for early stopping of the PCG iterations. In detail, when the error rate on $err\mathcal{V}$ on $\mathcal{V}$ is non decreasing in consecutive or after every θ iterations then the PCG iterations should be stopped. However, as usually the labeled validation data is small with respect to whole training data, $\mathcal{S}$, it may not be enough to represent the structure of the dataset(, and it may also be contaminated with noise). Therefore, we preferred to use only the stability check as an early stopping condition, since the $K\boldsymbol{\alpha}$ product must be calculate in every PCG iteration and its cost is only $O(u)$. The overall complexity of the Huberized LapSVM is the same as that the squared hinge LapSVMp, see for detail [15], the only difference is that we have to maintain two error vectors sets instead of one and in return the classification quality is increased.

Algorithm 4 The validation check for PCG iterations [15]

Input: Validation data set $\mathcal{V}$

1: $\text{err}\mathcal{V}_{old} \leftarrow 100\%$
2: $\eta \leftarrow 100.|\text{err}\mathcal{V}|^{-1}\%, \theta \leftarrow \frac{\sqrt{n}}{2}$
3: **if** $err(\mathcal{V}) > (\text{err}\mathcal{V}_{old} - \eta)$ **then**
4: Stop PCG
5: **else**
6: $\text{err}\mathcal{V}_{old} = \text{err}\mathcal{V}_{old}$
7: **end if**

6.4 Experiments

We performed experiments on a toy dataset, USPS, ISOLET, PASCAL VOC 2007 [73] and ImageNet [60] datasets. USPS and ISOLET represent two real world classification problems arising in visual and speech recognition. We select them since they were previously used (e.g., in [12, 15]) for evaluation of several semi-supervised classification algorithms. We make comparisons of the quality of the Huberized LapSVMp with two baselines, the linear hinge loss LapSVM [12] and the squared hinge loss LapSVMp [15]. LapSVM [12] is trained as a classifier with the original dual formulations where as both the Huberized LapSVMp and LapSVMp [15] are trained in the primal with PCG algorithms. For the

qualitative comparisons between the Huberized LapSVMp and the LapSVMp for the tasks of object classification, a number of experimental results are presented for PASCAL VOC 2007 [73] dataset and the ImageNet [60] test dataset. We repeat experiments performed in section 4.5.1 for the Huberized LapSVMp and use the same experimental setup as used for the LapSVMp.

In all the experiments, we have used the stability check as an early stopping criteria for the PCG, since it does not required a cleanly labeled validation dataset. Our code will be made available online for a replication of the results. We thank Belikin et al. [12] and Mellacci et al. [15] who have made their code and benchmarking scripts available which we use in the following experiments. In conducting these experiments our purpose is to evaluate the performance of the proposed algorithm with the two baselines in situations when a learner is provided with training examples (labeled and unlabeled) where the observed labels may be possibly contaminated with labeled noise.

6.4.1 2D Two Moons Datasets

To compare the different loss functions and their sensitivity to the amount of label noise, we conduct an experiment using a synthetic 2D Two moons dataset an example of which is shown in Figure 6.2a. We generated a training dataset containing 1000 points (500 points per class, upper and lower moon) and a test dataset containing 200 points (100 points per class) using the code[†] that is provided by Belkin et al. [12].

We created 9 different types of subsets from the training dataset by uniform sampling of 30 points per class as labeled and the remaining as unlabeled points. Where in each subset, we introduced noise in the labels with varying fractions $p = 0, 0.05, \cdots, 0.4$: for each fraction p parts out of the labeled points were randomly chosen and their true class labels were flipped. We trained three classifiers using each of the subsets as training points, the LapSVM is trained in dual where the LapSVMp and the Huberized LapSVMp with $h = 0.5$ were trained with PCG algorithm. For all the classifiers, we set $\gamma_A = 0.03125$, $\gamma_I = 1$ and the width $\tau = 0.35$ for an RBF kernel of the form $k(\mathbf{x}_i, \mathbf{x}_j) = exp(-\frac{D(\mathbf{x}_i, \mathbf{x}_j)}{2\tau^2})$, where the euclidean distance D

[†] `http://manifold.cs.uchicago.edu/manifold_regularization/software.html`

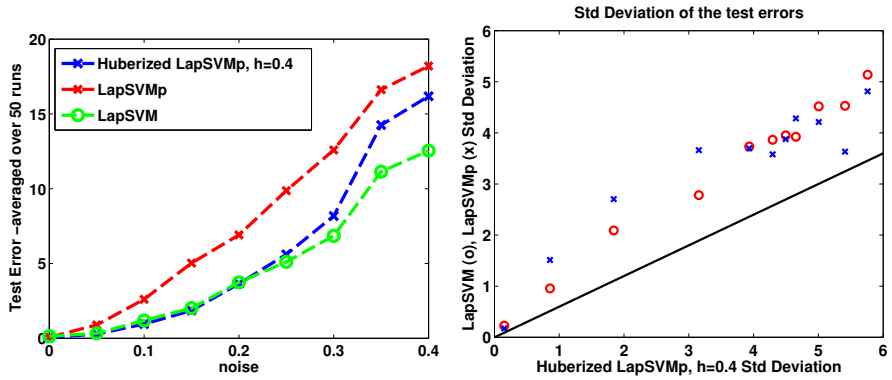


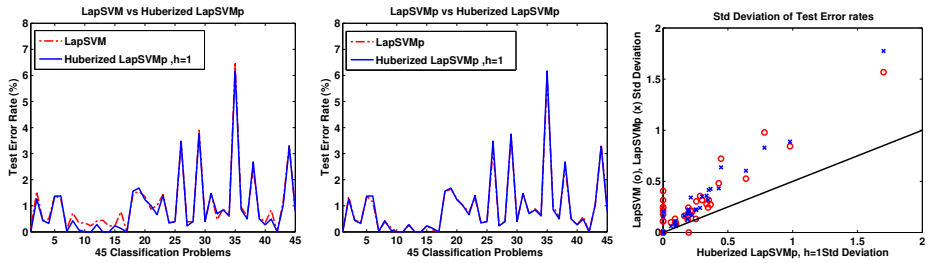
Figure 6.3: 2D-Two Moons Experiment: Test classification errors measured in percent and average over 50 runs for each fraction of noise in the training data.

between two examples is taken as $D(\mathbf{x}_i, \mathbf{x}_j) = \|\mathbf{x}_i - \mathbf{x}_j\|^2$. Additionally, we use a kNN graph for the construction the graph Laplacians L of degree $p = 2$ where $k = 6$. The performance is evaluated using the classification test error rates, averaged over 50 independent runs for each fraction of noise.

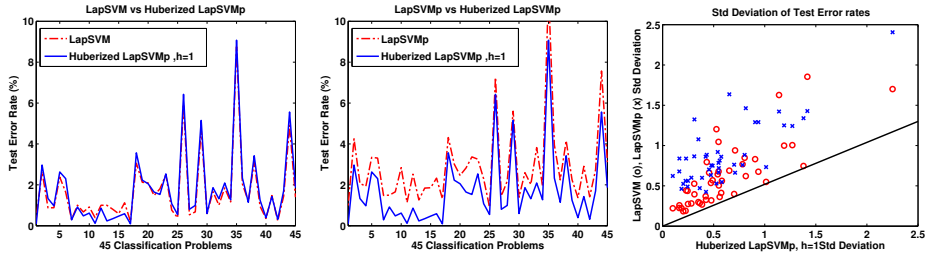
The final results are shown in Figure 6.3. We can see that the Huberized LapSVMp performs better than LapSVMp and mostly equal to the LapSVM as the level of noise rises in the training data. It confirms our intuitions that the squared hinge loss penalizes mislabeled points harder than either the Huber hinge loss or the linear hinge loss function.

6.4.2 USPS: Datasets for Handwritten Digit Recognition

In this set of experiments we applied the Huberized LapSVMp and the baselines to 45 binary classification problems that arise in pairwise classification of 10 classes of handwritten digits. The images of each digit in USPS training set has been preprocessed using PCA to 100 dimensions. The first 400 images of each digit in the training set were taken to form the training dataset for the three classifiers, the LapSVM, the LapSVMp and the Huberized LapSVMp. The remaining images form the test set, a total of over 3000 images. 50 images for each class were randomly taken as labeled ($l = 50$) and the rest were left unlabeled ($u = 350$) during training of the classifiers. We conducted three types of experiments. First, we train all the classifiers with cleanly labeled data: the purpose is to evaluate the quality



(a) The three classifiers were trained with noise free labeled data



(b) The three classifiers were trained with the labeled data for each classification problem is contaminated by 20% noise

Figure 6.4: USPS Experiment: Test error rates, measured in percent, at precision-recall break even points for 45 classification problems

of the Huberized LapSVMp from the point of view of the approximate vs optimal solution. Next, we train all of the three classifiers on the training set after its labeled part is contaminated with 20% noise. Finally, we investigate the influence of h on the classification performance of the Huberized LapSVMp when trained with noisy labeled dataset.

Following the experimental protocol in [12], we used a polynomial kernel $k(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i^T \mathbf{x}_j + 0)^3$ with degree 3 for training of the three classifiers. We set $\gamma_A = 0.005$, $\gamma_I = 1$ and used binary kNN graphs with $k = 6$ for building the graph Laplacians. Its worthy to note that the observations reported here hold consistent by a wide choice of hyper parameters.

In Figure 6.4, we compare the error rates of the Huberized LapSVMp and the two baselines, at the precision-recall breakpoints in the ROC curves for the 45 classification problems. These results are averaged over 10 random choices of labeled examples. Figure 6.4a shows the comparison of the three classifiers when they are trained with noiseless labeled examples. Figures 6.4b, compares the tests error-rates of the Huberized LapSVMp with the baselines after introducing 20% of noise in the

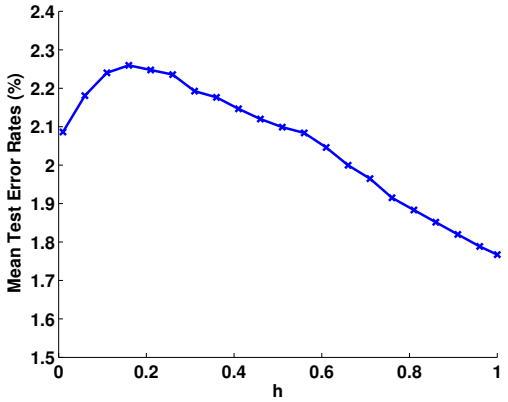


Figure 6.5: USPS Experiment: Influence of the parameter h of the Huberized LapSVMp classifier, on the mean test error at precision-recall break even points

labeled part of the training examples: in each classification problem 20% out of the labeled points are randomly chosen and their labels are flipped. Here, we can see that for all classification problems the Huberized LapSVMp performs equally well compared to the baselines when the available training data is noise free. Note that LapSVM solves first a quadratic problem and then solves a linear system of equations which involves the inversion of large matrix while both the Huberized LapSVMp and the LapSVMp are training directly in the primal with PCG which do not involve any matrix inversions. This results in faster training times for the primal solutions in this experiment at equal accuracy. In Figures 6.4b, we can see that the performance of the basline LapSVMp drops in comparison to Huberized LapSVMp after introducing noise in the training data. As shown on scatter plots in Figures 6.4 on standard deviation of error rates, the performance of the Huberized LapSVMp is quite stable with respect to the baselines in both noise free and noisy cases. To further investigate the influence of the parameter h on the mean test error rate, we trained the Huberized LapSVMp for various values of h with 20% noisy labeled data. The final results are shown in Figure 6.5, which shows an improvement in performance with increase in the value of h beyond 0.2.

6.4.3 ISOLET: Datasets for Spoken Letter Recognition

This experiment is performed on the ISOLET dataset of letters of the English alphabet spoken in isolation. It contains vocal sounds of 60 people who spoke twice

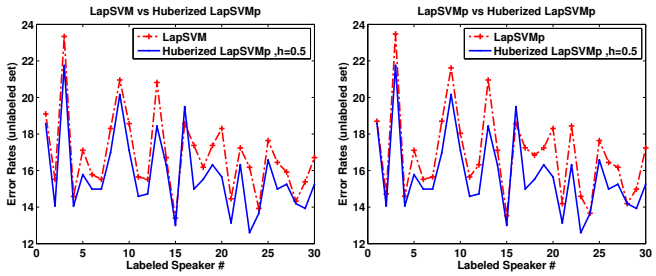
the name of each letter. The people are grouped into 2 sets, and each set contains 30 people. Following the experimental procedure proposed in [12], we chose to train on the first group of 30 speakers forming a training set of 1560 examples and a test set containing 1560 examples are taken from the second group. The task is to classify the first 13 alphabets sounds from the from the last 13. We considered the 30 binary classifying problems, each corresponds to the 30 splits of the training set where the 52 sounds of one speaker were taken as labeled set and the rest 1508 sounds were taken as unlabeled set. The test dataset were completely new speakers that are seen in the training set.

We trained the three classifiers, LapSVM, LapSVMp and our Huberized LapSVMp for each of the 30 classification problems. We first trained them with the clean training set and then on the 5% noisy labeled training set. Following the experimental protocol in [12], we chose an RBF kernel of $k(\mathbf{x}_i, \mathbf{x}_j) = exp(-\frac{D(\mathbf{x}_i,\mathbf{x}_j)}{2\tau^2})$; where the euclidean pairwise distance D between two examples is taken as $D(\mathbf{x}_i, \mathbf{x}_j) = \|\mathbf{x}_i - \mathbf{x}_j\|^2$ and the kernel width $\tau = 10$ for training of the three classifiers. We set $\gamma_A = 0.005$, $\gamma_I = 0.005$ and used a binary kNN graphs with $k = 6$ for building normalized graph Laplacians.

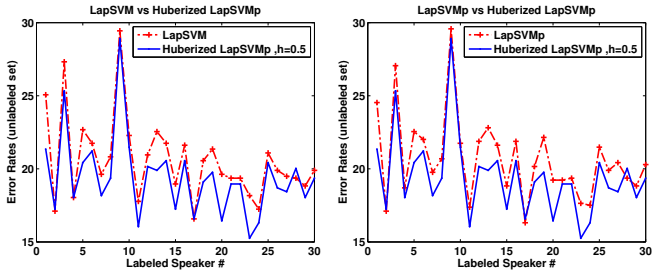
In Figure 6.6, we compare the error rates on the unlabeled set measured at precision-recall break even points for the three classifier. In Figure 6.6, we found that accuracy of the Huberized LapSVMp is consistently better than the baselines.

6.4.4 PASCAL VOC Dataset for Object Classification

To give a quantitative evaluation of the classifiers the Huberized LapSVMp and the LapSVMp we perform experiments using the PASCAL VOC 2007 [73] dataset. The dataset contains around 10000 images which were downloaded by querying for images of 20 object categories. These images are then manually annotated or labeled for each of the 20 categories and are provided with training and test splits of approximately 5000 images. Each test image contains an object instance of more than one class. The task is to predict for each of the twenty object classes the presence or absence of at least one of that the object classes in a test image. We consider a semi-supervised learning scenario and train these classifiers for 20 binary classification tasks. We evaluate these methods by measuring their classification



(a) The classifiers were trained on noise free labeled data.



(b) The classifiers were trained on noisy labeled data. The labeled data for each classification problem is contaminated by 5% noise

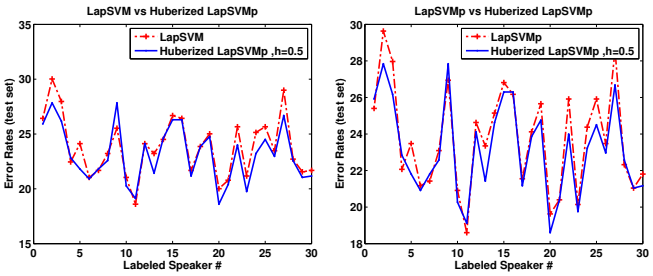
Figure 6.6: ISOLET Experiment: Error rates (Unlabeled set), measured in percent, at precision-recall break even points for 30 classification problems.

performances, on the test split of the dataset, using average precision (AP) criterion for each class and also using the mean AP over all classes.

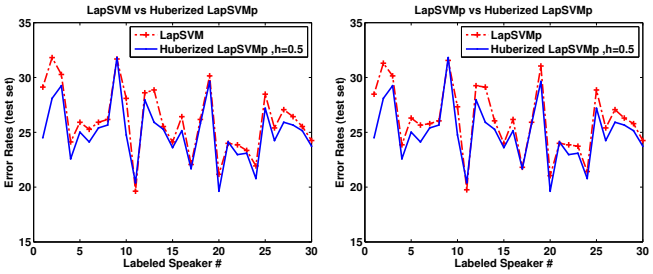
We use local SIFT features for image representation[‡]. For each image these features are computed from 16×16 pixels densely sampled with a step size of 8 pixels and multiple scales. Each local feature is quantized using k-means clustering on samples from the training splits. All the images are then represented by a histogram of bag-of-visual words of size 1000.

For each of the 20 classification tasks, we select a same number of positive (pos) and negative (neg) images from the training split of the dataset while the rest of the images are taken as unlabeled images for training the classifiers. We use the χ^2 distance RBF kernel and the width of the kernel τ is set to their average pairwise distance. To construct to the graph Laplacian, we build the sparse adjacency matrix $\mathbf{W}$ with $k = 20$ and the scaling factor σ is set to the mean of the k nearest neighbors

[‡]The "DenseSift" features for PASCAL VOC 2007 images were extracted and computed by the authors of [23] and are available at `http://lear.inrialpes.fr/people/guillaumin/data/cvpr10/pascal07.20100609.tar.bz2`.



(a) The classifiers were trained on noise free labeled data.



(b) The classifiers were trained on noisy labeled data. The labeled data for each classification problem is contaminated by 5% noise.

Figure 6.7: ISOLET Experiment: Error rates (Test set), measured in percent, at precision-recall break even points for 30 classification problems.

pairwise χ^2 distances. Figure 6.8, shows the sparse adjacency graph constructed using all the training split images where we can see how the images are noisy across the object classes. The regularization parameters, $\gamma_A = 1$ and $\gamma_I = 10$, are fixed in advance in training the classifiers for 20 binary classification tasks. These are chosen for the best "person" classification performance on the test split of the dataset using LapSVMp. The value of h for the Huberized loss function is set to 0.9.

Figure 6.9, shows the comparisons of the classification performances for the individual tasks versus the number unlabeled training images using the Huberized LapSVMp and the LapSVMp classifiers with a fixed number of labeled images. We can see that in most of the object classes, e.g., "bottle", "bus" and "cow", the performance of the Huberized LapSVMp increases and remains stable when increasing the number unlabeled images. These results are summarized and shown in Figure 6.10. The precision/recall curves for the best max AP for each task and the two classification methods results are shown in Figure 6.10. To visually observe the quality of both classification methods, examples images with their ranking number by each of the classifiers in "aeroplane", "bird", "car", "motorbike", "person", "pottedplant",

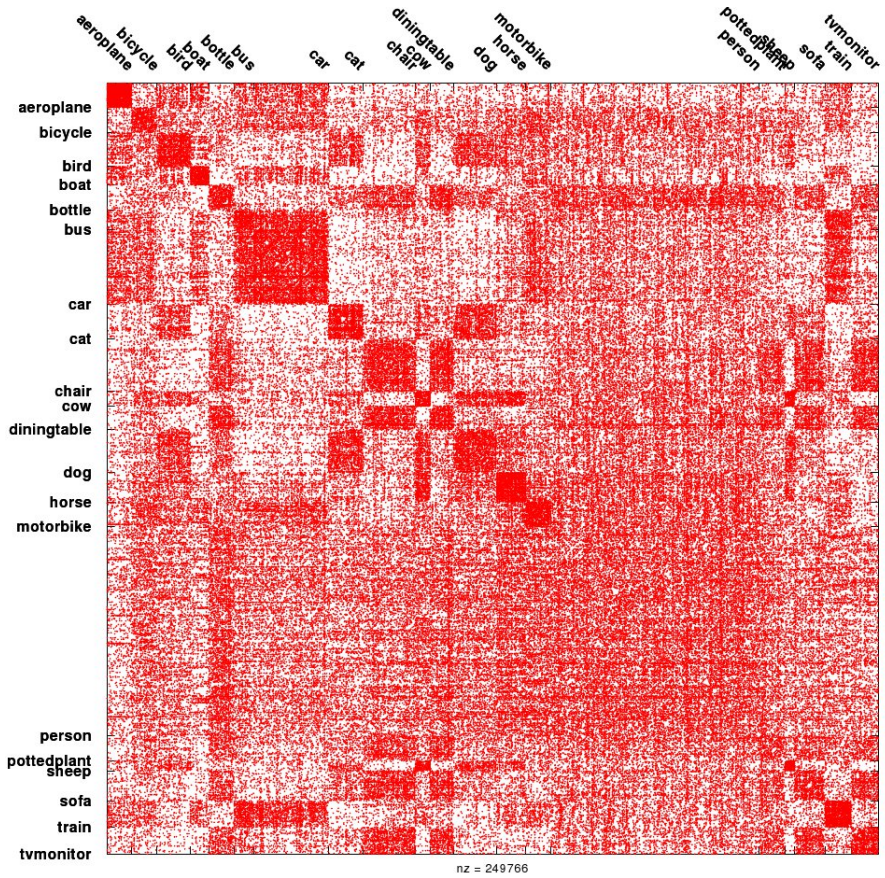


Figure 6.8: PASCAL VOC 2007 [73] Experiment: The sparse kNN adjacency graph $\mathbf{W}$ for all training split images $\mathcal{S}$ used in the Huberized LapSVMp and the LapSVMp classifiers. For building this graph, the number of nearest neighbors k is set to 20 and the χ^2 pairwise distance measure is used in (6.1).

and "sheep" classification tasks are shown in Figures 6.12, 6.13, 6.14, 6.15, 6.16, 6.17, and 6.18 respectively.

The final results for which both the classifiers achieve their maximum AP in each of the 20 binary classification tasks are presented in Table 6.1. It can be observed that in 10 classes the Huberized LapSVM classifier is stronger and have more consistent performance than the LapSVMp classifier. Interestingly, both classifiers achieve higher performance score than the winner.

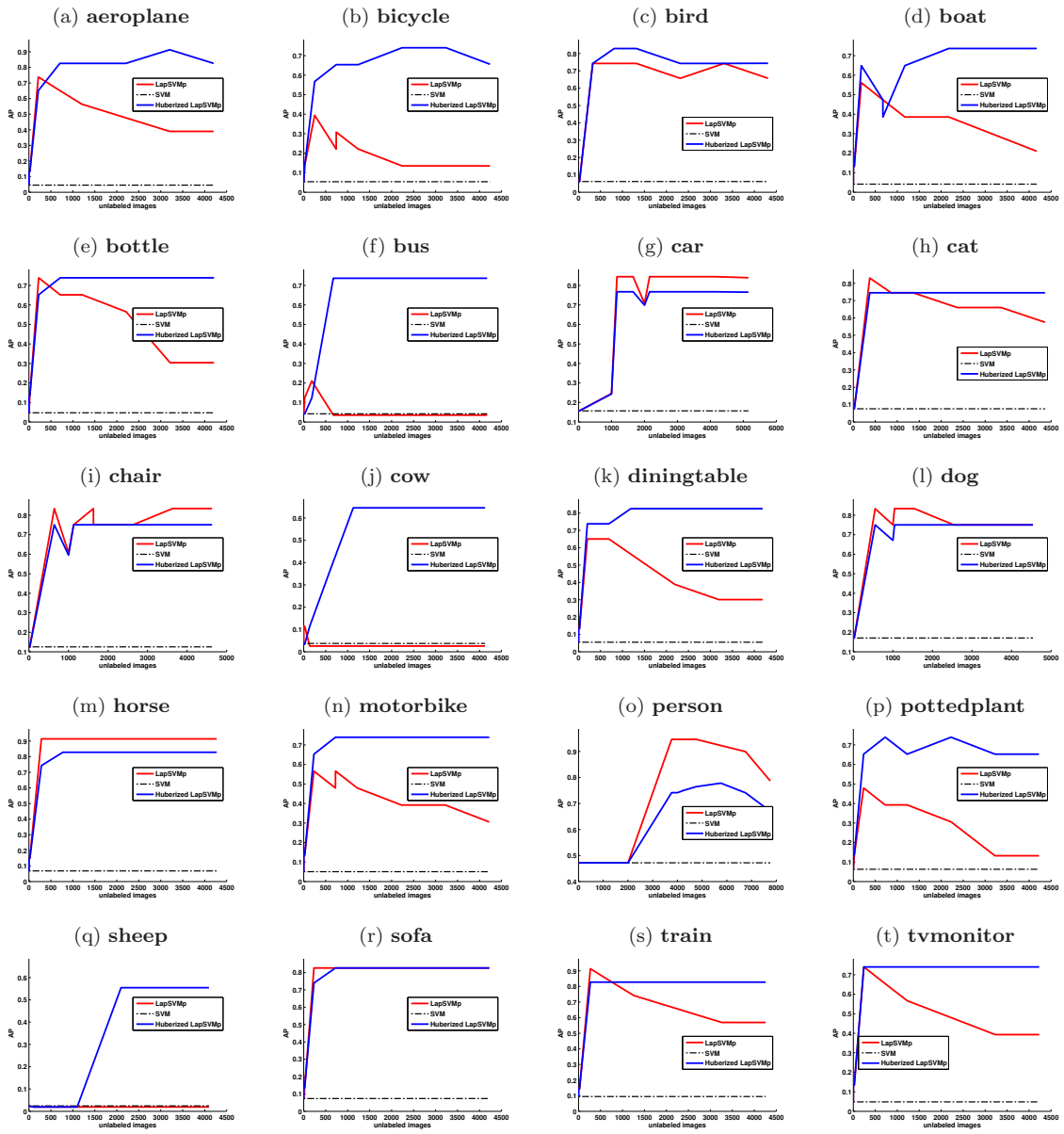


Figure 6.9: PASCAL VOC 2007 [73] Experiment: The AP scores for 20 object classification of the test images versus the number of unlabeled images used in training the classifiers shown in the corresponding legends. For each task 200 to 300 number of positive images (containing the corresponding object class) and 300 negative images (not containing the corresponding object class) are used for training these classifiers.

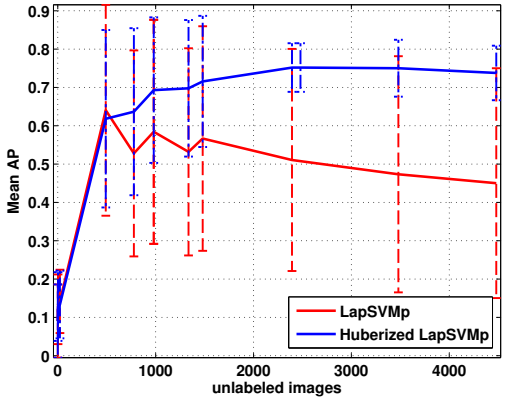
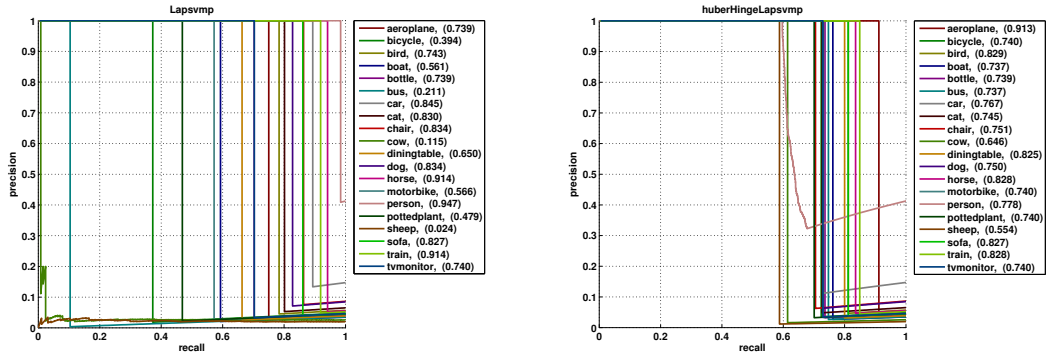


Figure 6.10: PASCAL VOC 2007 [73] Experiment: The mean AP scores over all the classes versus the number of unlabeled images



(a) LapSVMp (b) Huberized LapSVMp

Figure 6.11: PASCAL VOC 2007 [73] Experiment: Precision/recall curves are shown for the twenty object classes. (**a**) shows all results obtained with the Huberized LapSVMp classifiers; (**b**) (**a**) shows all results obtained with the LapSVMp classifiers.

Method ↓	aeroplane	bicycle	bird	boat	bottle	bus	car	cat	chair	cow	diningtable
Best VOC2007 [73]	77.50	63.60	56.10	71.90	33.10	60.60	78.00	58.80	53.50	42.60	54.90
LapSVMp	73.85	39.44	74.28	56.13	**73.90**	21.06	**84.49**	**83.00**	**83.39**	11.55	65.05
Huberized LapSVMp	**91.28**	**74.05**	**82.86**	**73.68**	**73.90**	**73.69**	76.74	74.50	75.08	**64.57**	**82.52**

	dog	horse	motorbike	person	pottedplant	sheep	sofa	train	tvmonitor	Mean
Best VOC2007 [73]	45.80	77.50	64.00	85.90	36.30	44.70	50.60	79.20	53.20	59.94
LapSVMp	**83.36**	**91.41**	56.59	**94.66**	47.94	2.44	**82.66**	**91.38**	74.00	64.53
Huberized LapSVMp	75.04	82.83	**73.95**	77.82	**73.97**	**55.44**	**82.66**	82.77	**74.00**	**76.07**

Table 6.1: PASCAL VOC 2007 [73] Experiment: The AP scores, measured in percent, for the object classification results. The **bold** entries in each column indicate the maximum AP for the corresponding object class. The last column shows the mean AP score over all the classes

.

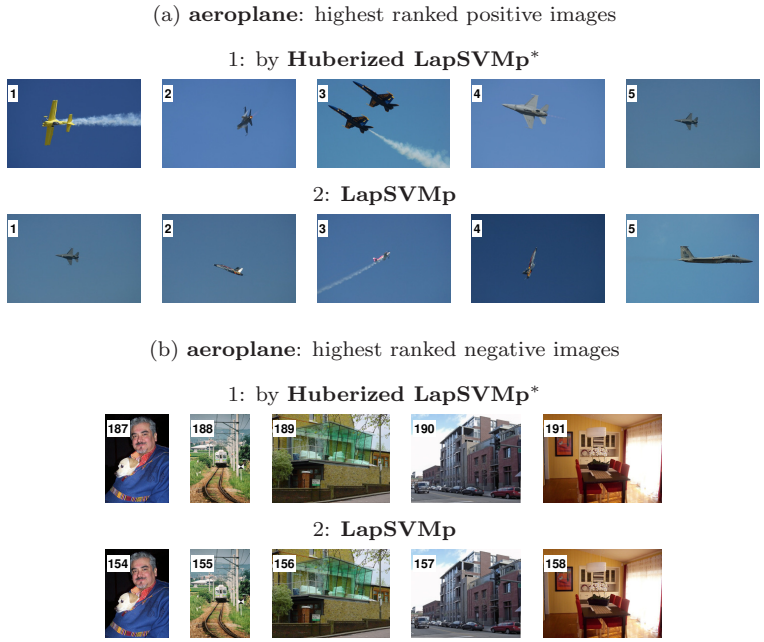


Figure 6.12: Ranked images for "aeroplane" classification task.(**a**) Five highest ranked *positive* images (containing aeroplanes); (**b**) five highest ranked *negative* images (not containing aeroplanes), each by the two classifiers. The number in each image indicates the corresponding image rank number. The classifier with best max AP score is marked with *.

(a) **bird**: highest ranked positive images

1: by **Huberized LapSVMp***

2: **LapSVMp**

(b) **bird**: highest ranked negative images

1: by **Huberized LapSVMp***

2: **LapSVMp**

Figure 6.13: Ranked images for "bird" classification task.(**a**) Five highest ranked *positive* images (containing birds); (**b**) five highest ranked *negative* images (not containing birds), each by the two classifiers. The number in each image indicates the corresponding image rank number. The classifier with best max AP score is marked with *.

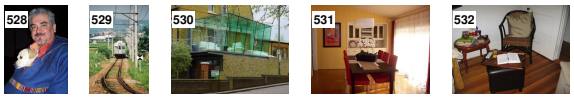
(a) **car**: highest ranked positive images

1: by **Huberized LapSVMp**

2: **LapSVMp***

(b) **car**: highest ranked negative images

1: by **Huberized LapSVMp**



2: **LapSVMp***

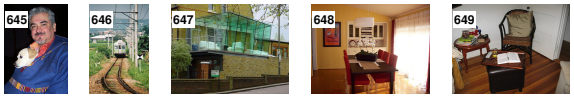


Figure 6.14: Ranked images for "car" classification task.(**a**) Five highest ranked *positive* images (containing cars); (**b**) five highest ranked *negative* images (not containing cars), each by the two classifiers. The number in each image indicates the corresponding image rank number. The classifier with best max AP score is marked with *.

(a) **motorbike**: highest ranked positive images

1: by **Huberized LapSVMp***

2: **LapSVMp**

(b) **motorbike**: highest ranked negative images

1: by **Huberized LapSVMp***

2: **LapSVMp**

Figure 6.15: Ranked images for "motorbike" classification task.(**a**) Five highest ranked *positive* images (containing motorbikes); (**b**) five highest ranked *negative* images (not containing motorbikes), each by the two classifiers. The number in each image indicates the corresponding image rank number. The classifier with best max AP score is marked with *.

(a) **person**: highest ranked positive images

1: by **Huberized LapSVMp**

2: **LapSVMp***

(b) **person**: highest ranked negative images

1: by **Huberized LapSVMp**

2: **LapSVMp***

Figure 6.16: Ranked images for "person" classification task.(**a**) Five highest ranked *positive* images (containing persons); (**b**) five highest ranked *negative* images (not containing persons), each by the two classifiers. The number in each image indicates the corresponding image rank number. The classifier with best max AP score is marked with *.

(a) **pottedplant**: highest ranked positive images

1: by **Huberized LapSVMp***

2: **LapSVMp**

(b) **pottedplant**: highest ranked negative images

1: by **Huberized LapSVMp***

2: **LapSVMp**

Figure 6.17: Ranked images for "pottedplant" classification task.(**a**) Five highest ranked *positive* images (containing pottedplants); (**b**) five highest ranked *negative* images (not containing pottedplants), each by the two classifiers. The number in each image indicates the corresponding image rank number. The classifier with best max AP score is marked with *.

(a) **sheep**: highest ranked positive images

1: by **Huberized LapSVMp***

2: **LapSVMp**

(b) **sheep**: highest ranked negative images

1: by **Huberized LapSVMp***

2: **LapSVMp**

Figure 6.18: Ranked images for "sheep" classification task.(**a**) Five highest ranked *positive* images (containing sheeps); (**b**) five highest ranked *negative* images (not containing sheeps), each by the two classifiers. The number in each image indicates the corresponding image rank number. The classifier with best max AP score is marked with *.

	aeroplane	motorbike	train	car	bus	cow	horse
LapSVMp	80.07	77.69	78.57	74.10	84.94	78.82	66.61
Huberized LapSVMp	**85.40**	**89.01**	**80.28**	**86.70**	**88.16**	**83.88**	**67.05**

Table 6.2: ImageNet [60] dataset: The AP classification scores, in percent, on the entire test set images taken from ImageNet [60]. Both Classifiers are trained on the Internet images. The best scores are marked in bold.

6.4.5 Automatic training of object category classification models

This set of experiments are performed to evaluate the Huberized LapSVMp for the tasks of automatic training of object category classification. We repeat the experiments presented in Section 4.5.1 for the Huberized LapSVMp adopting the experimental training setup for the LapSVMp. To remind, given the names of object category all the training images are automatically collected using multiple search engines. For each object class, the top 30 images ranked by an Internet search engines images are considered as noisy labeled. Since, the supervision is provided by the Internet therefore there is a possibility to get images that may not contain the object of the target class. Similarly, the lower ranked 30 images are taken as unlabeled. The negative labeled images and additional unlabeled images are taken from the searched returns for other categories. Given these collected images, the task is to train object classifiers and then to predict the presence or absence of at least one of that object class in a test images. The test images,more than , for 7 categories of objects are taken from the ImageNet [60] dataset.

Each image is represented by concatenation of PHOW and PHOG descriptors, although an MKL could be applied but objective here is to compare with LapSVMp under the same training conditions (see Chap 4 for details). The parameter h for the Huberized LapSVMp is set to 0.9 and the values of all the rest hyper parameters, the type kernel function, the adjacency graph and graph distance measures are used similar to the LapSVMp.

The final experimental results, presented in Table 6.2, compares classification performances (measured in AP) of the Huberized LapSVMp with the LapSVMp classifiers. Here, we observe that for all categories of objects the Huberized LapSVMp

out the performs the LapSVMp. The edge in performance is due to the robust loss function.

6.5 Conclusions

In this chapter, we presented the Huberized LapSVM a strategy for solving the optimization problem of LapSVM in the primal. Following LapSVMp [15], a fast solution is achieved using the preconditioned conjugate gradient coupled with an early stopping criteria based on the classifier decision. Instead of using the standard squared hinge loss function, we introduced the huber Hinge loss function to measure misclassification during the training process of a classifier. The Huber hinge loss gives a milder penalty than the squared hinge loss making the proposed solution fit in situations when the available training data is noisy. Detailed experimental results on 3 datasets, i.e., USPS, ISOLET and ImageNet, using several performance measures, validate our propose strategy is fit to solve classification problems when available training data is contaminated with label-noise. Additionally, experimental evaluations on ImageNet testdata, validate that the use of Huberized LapSVMp for learning models with images obtained directly from the Internet searches for object categorization. The results also demonstrated the consistent classification performance using the proposed strategy that outperforms the winner of PASCAL VOC 2007.

The overall results can be further improved by exploring the effects of sparse kernel expansion of the classification function. It could be also useful to study the effect on the performance by applying an incremental classifier building technique or user feed back in some kind of interactive setup.

Chapter 7

Conclusions

In this thesis we investigated the limitations in learning models directly with images collected from Internet searches for object categorization and detection. We addressed three main problems in training category-level image classifier and/or object detector using example images collected via image search engines. Firstly, ambiguous true-class label of images. Secondly, ambiguous location of an object contained in an image that belongs to the desired category. Finally, difficulty in selecting a sufficient visual representation specialized in capturing most of visual characteristics of objects within the same category. We focused on support vector machine (SVM) and Laplacian SVM (LapSVM) and proposed several methods to overcome the problems in training a desired category-level image classifier directly with images collected from the Internet.

Firstly, we showed that the image ranking quality of an image search engine could be enhanced when a simple visual features are combined with textual feature based representation using a supervised MKL method. A useful out-come of this work is a multi-modal dataset called the TVGraz. Experiemental evaluation on this dataset demonstrated that the combination of both modalities can boost the image ranking performance of an image search engine that uses only textual information in retrieval of images. For the task of web image search this is not a major issue, such enhancement is necessary in reducing class-label ambiguity in the retrieved images.

Secondly, we proposed a semi-supervised MIL algorithms called Co-miSVM for training category-level image classifiers directly on the searched images. This

method can exploit the underlying manifold semantics shared by different visual features by using a combined graph Laplacian. Experimental evaluations on a number of datasets demonstrated the usefulness of the proposed extensions in comparison with single instance learning methods and supervised multiple-instance SVM based methods.

Thirdly, we proposed a multi-stage system for training complex part based object detectors with the Internet searched images in order to overcome the problem of location ambiguity of an object contained in Internet images. The system works in three stages and requires only the name of the object class name as input. In the first two stages to deal with ambiguity in the true class label and location of objects, we applied our proposed MIL method for the actual learning tasks. This method extends the miSVM method with a semi-supervised kernel. We showed that it is possible to train a reasonable detector without using any manually annotated training examples. Experimental evaluations on challenging datasets demonstrated its effectiveness for training complex object detectors with ambiguously labeled images retrieved by image search engines.

Lastly, we proposed the Huberized LapSVM method for solving the optimization problem of the LapSVM in the primal space. A fast solution is achieved using the preconditioned conjugate gradient coupled with an early stopping criteria based on the classifier decision. Instead of using the standard squared hinge loss function, we used the Huber Hinge loss function to measure misclassification during the training process of a classifier giving a milder penalty to noisy labeled data. With this method through experimental evaluations on 5 datasets, we validated that our propose strategy is fit to solve classification problems when available training data is contaminated with label-noise. We also demonstrated the learning of object categorization models with images obtained directly from the Internet. The results also demonstrated the consistent classification performance using the proposed strategy that outperforms the winner of PASCAL VOC 2007. The overall results can be further improved by exploring the effects of sparse kernel expansion of the classification function. It could be also useful to study the effect on the performance by applying an incremental classifier building technique or user feed back in some kind of interactive setup while learning object models from Internet images.

Appendix A

List of Publications

The following is the list of research publications I, together with my colleagues, have made.

1. Khan, I., Saffari, A., and Bischof, H. (2009a). TVGraz: Multi-modal learning of object categories by combining textual and visual features. In *Proc. Workshop of the Austrian Association for Pattern Recognition*.

2. P. M. Roth, T. Mauthner, I. Khan, and H. Bischof, (2009b). Efficient human action recognition by cascaded linear classification. In *Proc. IEEE Workshop on Video-Oriented Object and Event Classification*.

3. E. Lex, I. Khan, H. Bischof, and M. Granitzer, (2010). Assessing the quality of web content. In Proc. Joint Conf. European ML & Knowledge Discovery in Databases.

4. I. Khan, P. M. Roth, and H. Bischof, (2011a). Learning object detectors from weakly-labeled internet images. In Proc. Workshop of the Austrian Association for Pattern Recognition

5. I. Khan, P. M. Roth, and H. Bischof, (2011b). Learning object categories from internet images. *"to be submitted to PR"*.

Bibliography

[1] A. Bosch, A. Zisserman, X. Munoz, Representing shape with a spatial pyramid kernel, in: Proc. ACM Int'l Conf. on Image and Video Retrieval, 2007.

[2] D. Lowe, Object recognition from local scale invariant features, in: Proc. IEEE Int'l Conf. on Computer Vision, 1999.

[3] A. Pinz, Object categorization, Found. Trends. Comput. Graph. Vis. 1 (4) (2005) 255–353.

[4] M. Varma, D. Ray, Learning The Discriminative Power-Invariance Trade-Off, in: Proc. IEEE Int'l Conf. on Computer Vision, 2007.

[5] A. Vedaldi, V. Gulshan, M. Varma, A. Zisserman, Multiple kernels for object detection, in: Proc. IEEE Int'l Conf. on Computer Vision, 2009.

[6] A. Rakotomamonjy, F. Bach, S. Canu, Y. Grandvalet, SimpleMKL, Journal of Machine Learning Research 9 (2008) 2491–2521.

[7] F. R. Bach, G. R. G. Lanckriet, M. I. Jordan, Multiple kernel learning, conic duality, and the SMO algorithm, in: Proc. Int'l Conf. Machine Learning, 2004.

[8] T. Dietterich, R. Lathrop, T. Lozano-Pérez, Solving the multiple instance problem with axis-parallel rectangles, Artificial Intelligence 89 (1-2) (1997) 31–71.

[9] S. Vijayanarasimhan, K. Grauman, Keywords to visual categories: Multiple-instance learning for weakly supervised object categorization, in: Proc. IEEE Conf. on Computer Vision and Pattern Recognition, 2008.

[10] G. Lanckriet, N. Cristianini, P. Bartlett, L. E. Ghaoui, Learning the kernel matrix with semidefinite programming, Journal of Machine Learning Research 5 (2004) 27–72.

[11] J. Ponce, T. Berg, M. Everingham, D. Forsyth, M. Hebert, S. Lazebnik, M. Marszalek, C. Schmid, B. Russell, A. Torralba, et al., Dataset issues in object recognition, Toward Category-Level Object Recognition 4170 (2006) 29–48.

[12] M. Belkin, P. Niyogi, V. Sindhwani, Manifold Regularization: A geometric framework for learning from labeled and unlabeled examples, Journal of Machine Learning Research 7 (2006) 2399–2434.

[13] V. Sindhwani, P. Niyogi, M. Belkin, Beyond the point cloud: from transductive to semi-supervised learning, in: Proc. Int'l Conf. Machine Learning, 2005.

[14] S. Andrew, I. Tsochantaridis, T. Hofmann, Support Vector Machines for Multiple-Instance Learning, in: Advances in Neural Information Processing Systems, 2003.

[15] S. Melacci, M. Belkin, Laplacian Support Vector Machines Trained in the Primal, Journal of Machine Learning Research 12 (2011) 1149–1184.

[16] R. C. Bunescu, R. J. Mooney, Multiple instance learning for sparse positive bags, in: Proc. Int'l Conf. Machine Learning, 2007.

[17] R. Fergus, L. Fei-Fei, P. Perona, A. Zisserman, Learning object categories from google's image search, in: Proc. IEEE Int'l Conf. on Computer Vision, 2005.

[18] T. Hofmann, Probabilistic latent semantic indexing, in: Proc. ACM SIGIR Int'l Conf. on R & D in Information Retrieval, 1999.

[19] R. Fergus, L. Fei-Fei, P. Perona, A. Zisserman, Learning Object Categories from Internet Image Searches, in: Proc. IEEE Special Issue on Internet Vision, 2010.

[20] M. Fritz, B. Schiele, Decomposition, discovery and detection of visual categories using topic models, in: Proc. IEEE Conf. on Computer Vision and Pattern Recognition, 2008.

[21] T. Berg, D. Forsyth, Animals on the web, in: Proc. IEEE Conf. on Computer Vision and Pattern Recognition, 2006.

[22] F. Schroff, A. Criminisi, A. Zisserman, Harvesting Image Databases from the Web, IEEE Trans. on Pattern Analysis and Machine Intelligence 33 (2011) 754–766, ISSN 0162-8828.

[23] M. Guillaumin, J. Verbeek, C. Schmid, Multimodal semi-supervised learning for image classification, in: Proc. IEEE Conf. on Computer Vision and Pattern Recognition, 2010.

[24] W. Hsu, L. Kennedy, S. Chang, Reranking methods for visual search, IEEE Trans. MultiMedia 14 (2007) 14–22.

[25] Y. Jing, S. Baluja, Visualrank: Applying pagerank to large-scale image search, IEEE Trans. on Pattern Analysis and Machine Intelligence 30 (2008) 1877–1890.

[26] J. Wang, Y. Jiang, S. Chang, Label diagnosis through self tuning for web image search, in: Proc. IEEE Conf. on Computer Vision and Pattern Recognition, 2009.

[27] W. Liu, Y. Jiang, J. Luo, S. Chang, Noise Resistant Graph Ranking for Improved Web Image Search, in: Proc. IEEE Conf. on Computer Vision and Pattern Recognition, 2011.

[28] D. Zhou, J. Weston, A. Gretton, O. Bousquet, B. Schölkopf, Ranking on Data Manifolds, in: Advances in Neural Information Processing Systems, 2004.

[29] V. Vapnik, The nature of statistical learning theory, Springer Verlag, 2000.

[30] B. Schölkopf, A. J. Smola, Learning with kernels : support vector machines, regularization, optimization, and beyond, The MIT Press, 2002.

[31] L. Bottou, C.-J. Lin, Support Vector Machine Solvers, in: L. Bottou, O. Chapelle, D. decoste, J. Weston (Eds.), Large Scale Kernel Machines, MIT Press, Cambridge, MA, USA, 301–320, 2007.

[32] O. Chapelle, P. Haffner, V. N. Vapnik, Support vector machines for histogram-based image classification, IEEE TRANSACTIONS ON NEURAL NETWORKS 10 (1999) 1055–1064.

[33] N. J. Nilsson, Learning machines: Foundations of Trainable Pattern Classifying Systems, McGraw-Hill, 1965.

[34] T. Hastie, R. Tibshirani, J. H. Friedman, The elements of statistical learning: data mining, inference, and prediction: with 200 full-color illustrations, Springer-Verlag, 2001.

[35] C. Cortes, V. Vapnik, Support-Vector Networks, in: Machine Learning, 1995.

[36] S. Sonnenburg, B. S. Bernhard, P. Bennett, E. Parrado-Hernández, Large scale multiple kernel learning, Journal of Machine Learning Research 7 (2006) 1531–1565.

[37] N. Aronszajn, Theory of reproducing kernels, Trans. American Mathematical Society 68.

[38] O. Chapelle, Training a support vector machine in the primal, Neural Computation 19 (5) (2007) 1155–1178.

[39] S. Keerthi, D. DeCoste, A modified finite Newton method for fast solution of large scale linear SVMs, Journal of Machine Learning Research 6 (1) (2006) 341.

[40] G. Kimeldorf, G. Wahba, Some results on Tchebycheffian spline functions, Journal of Mathematical Analysis and Applications 33 (1971) 82–95.

[41] O. Chapelle, B. Schölkopf, A. Zien, Semi-Supervised Learning, Cambridge, MA, 2006.

[42] A. Saffari, Multi-Class Semi-Supervised and Online Boosting, 2010.

[43] T. Joachims, Transductive inference for text classification using support vector machines, in: Proc. Int'l Conf. Machine Learning, 1999.

[44] V. Sindhwani, S. Keerthi, Large scale semi-supervised linear SVMs, in: Proc. Int'l Conf. Machine Learning, 2008.

[45] A. BLUM, Learning form labeled and unlabeled data using graph mincuts, in: Proc. Int'l Conf. Machine Learning, 2001.

[46] A. Blum, J. Lafferty, M. Rwebangira, R. Reddy, Semi-supervised learning using randomized mincuts, in: Proc. Int'l Conf. Machine Learning, 2004.

[47] D. Zhou, O. Bousquet, T. Lal, J. Weston, B. Schólkopf, Learning with local and global consistency, in: Advances in Neural Information Processing Systems, 2004.

[48] U. von Luxburg, A tutorial on spectral clustering, Statistics and Computing 17 (2007) 395–416.

[49] I. Khan, A. Saffari, H. Bischof, TVGraz: Multi-Modal Learning of Object Categories by Combining Textual and Visual Features, in: Proc. Workshop of the Austrian Association for Pattern Recognition, 2009.

[50] G. R. G. Lanckriet, N. Cristianini, P. Bartlett, L. E. Ghaoui, M. I. Jordan, Learning the Kernel Matrix with Semidefinite Programming, Journal of Machine Learning Research 5 (2004) 27–72.

[51] G. Griffin, A. Holub, P. Perona, The Caltech-256, Tech. Rep., California Institute of Technology, 2007.

[52] D. Cai, X. He, Z. Li, W.-Y. Ma, J.-R. Wen, Hierarchical clustering of WWW image search results using visual, textual and link information, in: Proc. Int'l Conf. Machine Learning, 2004.

[53] D. Cai, S. Yu, J.-R. Wen, W.-Y. Ma, VIPS: a VIsion-based Page Segmentation Algorithm, Tech. Rep., Microsoft Research, 2003.

[54] R. Fergus, P. Perona, A. Zisserman, D. E. Science, A Visual Category filter for Google images, in: Proc. European Conf. on Computer Vision, 2004.

[55] A. Bergamo, L. Torresani, Exploiting weakly-labeled web images to improve object classification: a domain adaptation approach, in: Advances in Neural Information Processing Systems, 2010.

[56] V. Sindhwani, P. Niyogi, M. Belkin, A co-regularization approach to semi-supervised learning with multiple views, in: Proc. ICML Worshop on Learning with Multiple Views, 2005.

[57] M. Stikic, B. Schiele, Activity recognition from sparsely labeled data using multi-instance learning, in: Proc. Int'l Symposium on Location and Context Awareness, 2009.

[58] Y. Jia, C. Zhang, Instance-level Semi-supervised Multiple Instance Learning, in: Proc. AAAI Conf. on Artificial Intelligence, 2008.

[59] A. Vezhnevets, J. Buhmann, Towards weakly supervised semantic segmentation by means of multiple instance and multitask learning, in: Proc. IEEE Conf. on Computer Vision and Pattern Recognition, 2010.

[60] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, ImageNet: A Large-Scale Hierarchical Image Database, in: Proc. IEEE Conf. on Computer Vision and Pattern Recognition, 2009.

[61] P. Gehler, O. Chapelle, Deterministic annealing for multiple-instance learning, in: Proc. Int'l Conf. on AI & Statistics, 2007.

[62] Z. Zhou, Y. Sun, Y. Li, Multi-Instance Learning by Treating Instances As Non-I.I.D. Samples, in: Proc. Int'l Conf. Machine Learning, 2009.

[63] T. Deselaers, V. Ferrari, A Conditional Random Field for Multiple-Instance Learning, in: Proc. Int'l Conf. Machine Learning, 2010.

[64] H. Wang, Q. Yang, H. Zha, Adaptive p-posterior mixture-model kernels for multiple instance learning, in: Proc. Int'l Conf. Machine Learning, 2008.

[65] P. Wohlhart, M. Köstinger, M. P. Roth, H. Bischof, Multiple Instance Boosting for Face Recognition in Videos, in: Proc. DAGM Symposium, 2011.

[66] P. Felzenszwalb, R. Girshick, D. McAllester, D. Ramanan, Object detection with discriminatively trained part based models, IEEE Trans. on Pattern Analysis and Machine Intelligence (2009) 1627–1645.

[67] B. Alexe, T. Deselaers, V. Ferrari, What is an object?, in: Proc. IEEE Conf. on Computer Vision and Pattern Recognition, 2010.

[68] N. Bruce, D.B., J. Tsotsos, K., Saliency Based on Information Maximization, in: Advances in Neural Information Processing Systems, 2006.

[69] P. Arbeláez, M. Maire, C. Fowlkes, J. Malik, From contours to regions: An empirical evaluation, in: Proc. IEEE Conf. on Computer Vision and Pattern Recognition, 2009.

[70] I. Endres, D. Hoiem, Category independent object proposals, in: Proc. European Conf. on Computer Vision, 2010.

[71] M. Nguyen, L. Torresani, F. De la Torre, C. Rother, Weakly supervised discriminative localization and classification:a joint learning process, in: Proc. IEEE Int'l Conf. on Computer Vision, 2010.

[72] V. Ferrari, T. Tuytelaars, L. V. Gool, Object Detection by Contour Segment Networks, in: Proc. European Conf. on Computer Vision, 2006.

[73] M. Everingham, C. K. I. Van Gool, L.and Williams, J. Winn, A. Zisserman, The PASCAL Visual Object Classes Challenge, 2007.

[74] S. Vijayanarasimhan, K. Grauman, Large-scale live active learning: Training object detectors with crawled data and crowds, 2011.

[75] F. Chung, Spectral graph theory, 92, Amer Mathematical Society, 1997.

[76] I. Tsang, J. Kwok, Large-scale sparsified manifold regularization, in: Advances in Neural Information Processing Systems, 2007.

[77] J. Shewchuk, An Introduction to the Conjugate Gradient Method Without the Agonizing Pain, Tech. Rep., School of Computer Science, Carnegie Mellon University, 1994.

[78] P. Huber, Robust estimation of a location parameter, The Annals of Mathematical Statistics 35 (1) (1964) 73–101.

[79] S. Boyd, L. Vandenberghe, Convex Optimization, Cambridge University, NY, 2004.