Image Classification with Limited Training Data and Class Ambiguity

Maksim Lapin, M.Sc.

A dissertation submitted towards the degree
Doctor of Engineering (Dr.-Ing.)
of the Faculty of Mathematics and Computer Science
of Saarland University

Saarbrücken, June 2017

Day of Colloquium	22 May, 2017
Dean of the Faculty	Prof. Dr. Frank-Olaf Schreyer Saarland University, Germany

Examination Committee

Chair	Prof. Dr. Joachim Weickert
Reviewer, Advisor	Prof. Dr. Bernt Schiele
Reviewer, Advisor	Prof. Dr. Matthias Hein
Reviewer	Prof. Dr. Christoph H. Lampert
Academic Assistant	Dr. Mykhaylo Andriluka

Abstract

Modern image classification methods are based on supervised learning algorithms that require labeled training data. However, only a limited amount of annotated data may be available in certain applications due to scarcity of the data itself or high costs associated with human annotation. Introduction of additional information and structural constraints can help improve the performance of a learning algorithm. In this thesis, we study the framework of learning using privileged information and demonstrate its relation to learning with instance weights. We also consider multitask feature learning and develop an efficient dual optimization scheme that is particularly well suited to problems with high dimensional image descriptors.

Scaling annotation to a large number of image categories leads to the problem of class ambiguity where clear distinction between the classes is no longer possible. Many real world images are naturally multilabel yet the existing annotation might only contain a single label. In this thesis, we propose and analyze a number of loss functions that allow for a certain tolerance in top k predictions of a learner. Our results indicate consistent improvements over the standard loss functions that put more penalty on the first incorrect prediction compared to the proposed losses.

All proposed learning methods are complemented with efficient optimization schemes that are based on stochastic dual coordinate ascent for convex problems and on gradient descent for nonconvex formulations.

Zusammenfassung

Moderne Bildklassifizierungsmethoden basieren auf überwachten Lernalgorithmen, die annotierte Trainingsdaten erfordern. In bestimmten Anwendungen steht aufgrund der Knappheit der Daten selbst oder der hohen Kosten, die durch Annotationen durch Menschen entstehen, jedoch vielleicht nur eine begrenzte Anzahl von annotierten Daten zur Verfügung. Die Einführung zusätzlicher Informationen und struktureller Nebenbedingungen kann dazu beitragen, die Leistung eines Lernalgorithmus zu verbessern. In dieser Arbeit untersuchen wir das Lernen mit privilegierten Informationen und zeigen eine Beziehung zum Lernen mit gewichteten Beispielen. Wir betrachten auch das Lernen von Merkmalen für Multitask Klassifikation und entwickeln eine effiziente duale Optimierungsmethode, die sich besonders gut für Probleme mit hochdimensionalen Bilddeskriptoren eignet.

Skalierung von Annotationen zu einer großen Anzahl von Bildkategorien führt zum Problem der Klassen-Ambiguität, wo eine klare Unterscheidung zwischen den Klassen nicht mehr möglich ist. Viele natürliche Bilder sind Teil mehrer Klassen, aber die vorhandene Annotation könnte möglicherweise nur ein einziges Label enthalten. In dieser Arbeit schlagen wir eine Reihe von Verlustfunktionen vor, die eine gewisse Toleranz in Top k Vorhersagen eines Klassifikators ermöglichen, und analysieren diese. Unsere Ergebnisse zeigen konsequente Verbesserungen gegenüber den üblichen Verlustfunktionen, die die erste falsche Vorhersage stärker bestrafen.

Wir ergänzen alle vorgeschlagenen Lernmethoden durch effiziente Optimierungsalgorithmen, die auf dualem Koordinatenanstieg für konvexe bzw. auf Gradientenverfahren für nichtkonvexe Probleme basieren.

Contents

List of Figures

List of Tables

1 Introduction

One way to motivate our work is to imagine the immense impact of machine learning and computer vision on our daily life. While specific contributions of this thesis are discussed in § 1.2, let us briefly consider the cumulative long-term effect of the research in these fields. Automation of human labor is ubiquitous in modern society while handmade products are rare and becoming increasingly exotic. Automated machinery is employed in production of virtually all materials and tangible goods, it has alleviated the burden of hard physical tasks, and dramatically transformed the nature of manual work increasing the value of knowledge, creativity, and analytical skills. Yet, however mundane and routine, the intellectual work has been traditionally difficult, if not impossible, to automate.

Today, we observe a rapid advance of the frontier for automation enabled by artificial intelligence (AI). An increasing amount of tasks that require some form of perception, reasoning, or recognition can now be executed by machines. The key technology behind the recent progress in AI is machine learning, which is concerned with the study of algorithms that can "learn", i.e. extract useful patterns, from data. This is an exciting area of research riddled with hard scientific problems that will have a profound social and economic impact when solved.

1.1 Teaching Machines to See

A high level problem that we are interested in is how to teach a machine to see. In § 1.1.1, we elaborate on what it means to "teach a machine" and cover some of the basics of *machine learning*, while in § 1.1.2 we explain what it means for a machine to "see" with a short historical tour of image recognition in *computer vision*. Towards the end of each section, we also discuss the challenges of the respective field that we address in the present dissertation.

1.1.1 Machine Learning

A traditional way to address a challenge in computer science is to (i) analyze it, (ii) recognize an underlying computing problem, and (iii) develop an algorithm that solves it or finds a useful approximation. For example, we can efficiently sort a list of objects or compute a reasonably short path for a traveling salesman[1]. Many more seemingly hard mathematical, logical, and engineering problems can be approached that way. However, there are also numerous tasks that are routinely

[1] The traveling salesman problem (TSP) is an NP-hard problem, however, a large number of efficient heuristics and (exponential) exact algorithms are known. For example, the World TSP instance with over 1.9M cities has been solved to within 0.05% of the optimal tour, see http://www.math.uwaterloo.ca/tsp/world/.

performed by humans and often seem effortless to us, yet are notoriously difficult to program. As Moravec, (1988, p. 15) writes,

> . . . it is comparatively easy to make computers exhibit adult level performance on intelligence tests or playing checkers, and difficult or impossible to give them the skills of a one-year-old when it comes to perception and mobility.

Moving around in space, recognizing a voice, recognizing a face, judging people's intentions and motivations are all examples of some of the oldest human skills that we have perfected over time. Yet, as they are largely unconscious, our understanding of such skills is not elaborate enough to implement them in a well defined program.

Machine learning, or automated learning, attempts to bypass the need to explicitly enumerate and consider all potential minute details and tiniest variations of the possible inputs. Instead, it is focused on developing a framework for learning algorithms that enable computers to automatically extract knowledge from "experience", i.e. training data, and apply it in solving a given task on previously unseen, test data. In contrast to the classical programming, a learning algorithm is expected to change and adapt to the given input examples – a process known as training or learning. Moreover, a successful learner should abstract away the irrelevant details and generalize beyond the training examples, which allows it to perform well on unseen data and surpasses learning by memorization. The process of learning general principles from observations is known as *inductive inference*, and it is precisely the process that machine learning is trying to automate.

The decision which information in the input is relevant and which is irrelevant for the given task is the fundamental problem in machine learning. It turns out, that the incorporation of the *prior knowledge* that biases the learning algorithm is essential to the success of learning. Without the prior knowledge there would be no way to decide which correlations in the input signal are spurious and which correspond to a useful pattern. This is formally defined and proved in the statistical learning theory, particularly in a series of "no free lunch" theorems that show that there exists no universally superior learning algorithm (Devroye et al., 2013). If one learning rule performs well on a certain task, then there must be another task where an alternative algorithm performs better.

The development of tools and techniques for incorporating the prior knowledge and translating it into an inductive bias is one of the central themes in the theory of machine learning. A specific example of introducing the prior knowledge is through the use of *regularization*, which is also a way to control the complexity of a learned model. Regularization allows one to impose certain restrictions on the learner which limits its ability to fit the training data exactly. A common phenomenon is often observed by machine learning practitioners: a method works great on the training data, but the results on the test data are poor. This situation is known as *overfitting* and suggests that the learner has fitted the training data too well failing to abstract away the irrelevant details. The opposite situation known as *underfitting* is also possible, and happens when the learned model is too simplistic to obtain good performance even on the training data. Regularization provides a way to tradeoff between over- and underfitting, and is one of the most important areas of research in machine learning (Duda et al., 2012).

Supervised Learning. Machine learning is a broad field of computer science and is concerned with different types of learning. For example, one differentiates between supervised and unsupervised learning, batch and online learning, active learning, and reinforcement learning. In this thesis, we consider *supervised learning* in the batch regime, which means that we are given a set, i.e. a batch, of n training pairs $\{(x_i, y_i)\}_{i=1}^n$, where $x_i \in \mathcal{X}$ is a training example and $y_i \in \mathcal{Y}$ is its label, and the goal is to construct a function $f : \mathcal{X} \to \mathcal{Y}$ optimizing a certain learning objective. The sets $\mathcal{X}$ and $\mathcal{Y}$ are usually referred to as the *input space* and the *output space* respectively. We only work with discrete finite output spaces, which corresponds to classification (or categorization), and distinguish between binary, multiclass, and multilabel classification depending on the set $\mathcal{Y}$.

The study of learning objectives is the main topic of this dissertation. We focus on regularized empirical risk minimization defined as follows (Hein, 2016).

Definition 1.1. Let $\{(x_i, y_i)\}_{i=1}^n$ be a training sample, $\mathcal{F}$ a class of functions, $L(y, f(x))$ a loss function, and $\Omega : \mathcal{F} \to \mathbb{R}_+$ a regularization functional. The **regularized empirical risk minimization** (RERM) is defined as

$$f^* = \operatorname*{arg\,min}_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^n L(y_i, f(x_i)) + \lambda \Omega(f),$$

where $\lambda \in \mathbb{R}_+$ is called the regularization parameter.

Our goal is to find a classifier $f \in \mathcal{F}$ that predicts reasonably good labels for any input $x \in \mathcal{X}$, not necessarily from the training set. First, we establish a measure of goodness using a loss function L which quantifies the level of discrepancy between the ground truth label y and our predicted label $f(x)$. The expected loss on unseen data is known as *risk*. Next, we introduce the prior knowledge by selecting an appropriate class of functions $\mathcal{F}$ and a suitable regularization functional Ω. Finally, we are confronted with the tradeoff between fitting the training data well (the first term) and having a simple, well-behaved model (the second term). This tradeoff is controlled using the regularization parameter λ, which is tuned on a holdout set or using cross-validation (Friedman et al., 2001).

In this thesis, we mainly use ℓ_2 regularization, but consider ℓ_p norms and other regularizers in (Jawanpuria et al., 2015). We also study extensions of the objective above in Chapters 3 and 4, where additional loss and regularization terms are introduced.

The main focus of this dissertation is on the analysis of loss functions and derivation of efficient optimization techniques for the corresponding learning objectives. We consider a wide range of different losses, including the classical hinge, logistic, and the softmax loss, as well as introduce novel loss functions in Chapters 5 and 6.

Surrogate Loss. Machine learning is a field of science that shares certain traits with statistics, optimization, information theory, and the general field of AI. However, we are more concerned with the practical aspects such as efficient algorithmic implementation and scalability to large datasets. When we develop learning algorithms, they have to work on real world data and produce meaningful results. An elaborate model, however elegant in theory, is of no practical use unless it can be trained efficiently. To this end, we need to consider the computational complexity of learning.

From the optimization point of view, the difficulty of solving the RERM problem directly depends on the participating objects: the function class $\mathcal{F}$, the loss function L, and the regularizer Ω. Our choices here are primarily governed by what is "easy" to optimize, in the sense that they lead to a convex optimization problem (Boyd and Vandenberghe, 2004). For example, we mainly work with linear classifiers of the form $f(x) = \langle w, x\rangle + b$ and ℓ_2 regularization, as mentioned above.

A good choice of the loss function, however, also takes into account how the classifier is evaluated in the end. For example, a natural measure for classification is the error count, which is also known as the 0-1 loss. Unfortunately, that loss is a discrete nonconvex function that leads to NP-hard combinatorial problems (Ben-David et al., 2003), which are intractable for our purposes. Instead, we consider continuous convex surrogate loss functions that can be optimized efficiently.

In the following, we distinguish between the target performance measure, which may be a discrete function like the 0-1 loss, and the *surrogate loss* function, which is actually used in the learning objective. An important question is whether anything can be said about the target performance of a classifier trained with a surrogate loss. To answer that, we introduce the notion of the *Bayes classifier* as the optimal classifier that minimizes the expected (discrete) loss. If X and Y are the random variables corresponding to a pair of observations in binary classification with the label encoding given by ±1, then we can define the *regression function* $\eta(x) = \mathbb{E}[Y \,|\, X = x]$ and obtain the Bayes classifier for the 0-1 loss as $f_B^*(x) = \mathbf{sign}\,\eta(x)$. Somewhat surprisingly, if we ignore the restrictions of the function class $\mathcal{F}$ and optimize the expected loss pointwise,

$$f_L^*(x) = \arg\min_{\alpha\in\mathbb{R}} \mathbb{E}_{Y\,|\,X}[L(Y,\alpha) \,|\, X = x],$$

then under relatively mild assumptions on the loss L, we obtain the Bayes classifier: $\mathbf{sign}\, f_L^*(x) = f_B^*(x) = \mathbf{sign}\,\eta(x)$. Such loss functions are known in the statistical learning theory as *classification calibrated* (Bartlett et al., 2006), and we come back to them in Chapter 6, extending the notion of calibration to a different loss.

Curse of Dimensionality. Let us now consider the input space $\mathcal{X}$, also known as the *feature space*. As hinted by the form of the linear classifier above, we mainly assume that the examples x_i are (feature) vectors in a d-dimensional Euclidean space, i.e. $\mathcal{X} = \mathbb{R}^d$. While there is no intrinsic limitation on the admissible range for d, an important phenomenon called the *curse of dimensionality* is recognized in the theory of machine learning. It can be illustrated with the following example taken from (Hein, 2016). Let $\mathcal{X} = [0,1]^d$ be the unit cube in $\mathbb{R}^d$, and partition every dimension into k parts, we obtain k^d bins in the cube. If we use simple majority vote classifiers, each restricted to their own bin, then we need $n \geq k^d$ training examples to classify in every bin. If the points are uniformly distributed and we classify e.g. only 95% of the volume, then it can be shown that we still need an exponential number of training examples. Moreover, even if consider the classical nearest neighbor (NN) method, it can still be shown that the 1-NN rule fails on some distributions unless we have exponentially many data points in the training set (Shalev-Shwartz and Ben-David, 2014).

A similar problem can be associated with the output space $\mathcal{Y}$ when the latter is complex (Tsochantaridis et al., 2005). For example, consider multilabel learning, where the task is to predict a set of labels for every $x \in \mathcal{X}$. The cardinality of $\mathcal{Y}$ increases exponentially with the number of classes, $|\mathcal{Y}| = 2^m$, and it quickly becomes unreasonable to expect to see every combination of the labels in the training set. Even for $m = 30$ classes, there are $2^{30} > 10^9$ possible subsets $Y \subset \mathcal{Y}$.

The only practical way to confront the curse of dimensionality is to make assumptions about the data and incorporate them as prior knowledge.

Challenges Addressed

We have provided a brief overview of the field of machine learning and the issues that arise in applications. In this section, we summarize some of the most important challenges that we aim to address in the present thesis.

Introduction of Prior Knowledge. Controlling the tradeoff between over- and underfitting and overcoming the curse of dimensionality are some of the most important aspects of a successful learner. Classifiers are particularly susceptible to overfitting when the amount of training data is limited or the dimensionality of the space $\mathcal{X}$ is high. Prior knowledge becomes extremely valuable in these situations, and we will consider different approaches that incorporate various forms of knowledge into the learning problem.

Scalable Optimization. Algorithmic aspects play a major role in machine learning applications, and the importance of utilizing large amounts of data has been highlighted by many authors (Banko and Brill, 2001; Halevy et al., 2009). In this thesis, we aim to develop efficient learning algorithms that scale well with respect to all three main dimensions: the number of training examples n, the number of classes m, and the number of features d.

Analysis of Loss Functions. Two factors complicate the choice of a learning objective: diversity of available methods, loss functions, and regularizers, and the discrepancy between the surrogate loss and the target performance measure. While the no-free-lunch theorem suggests that it is impossible to tell *a priori* which loss works best for the given problem, we still gain useful insights and guiding statistics from an experimental evaluation on real world datasets.

1.1.2 Computer Vision

Let us now consider what it means for a machine to "see". We start with a quick retrospective look at the early vision algorithms and then fast-forward to the present days. This allows us to appreciate the challenges in the field as well as the tremendous progress over the last decades. We mainly focus on the recognition aspect of vision, but acknowledge that many more tasks are being addressed by the community and refer the interested reader to the books by Forsyth and Ponce, (2003) and Szeliski, (2010).

Parts and Shape. Computer vision is a broad interdisciplinary field of science that is generally concerned with the perception and understanding of the physical world around us. As Ballard and Brown, (1982) write (emphasis added),

> Computer vision is the construction of explicit, meaningful *descriptions* of physical objects from images. Image understanding is very different from image processing, which studies image-to-image transformations, not explicit description building. Descriptions are a prerequisite for recognizing, manipulating, and thinking about objects.

Differentiating itself from image processing, early computer vision attempted to directly construct the "descriptions" of objects and recover their 3D structure. The modeling of nonrigid objects naturally involved elastic arrangements of parts or used cylinders to represent the human body. From the perspective of machine learning, we recognize that such models incorporated strong prior knowledge assumptions and required no training in the modern sense[2]. In fact, the idea that an object can be decomposed into a constellation of parts has been particularly influential in the field. Following Burl et al., (1998) and Weber et al., (2000), an object is composed of parts that form a shape, where the shape describes the mutual position of the parts. The shape can be modeled explicitly in a probabilistic framework (Fergus et al., 2003), or implicitly by considering which local appearances are consistent with each other (Leibe and Schiele, 2003). The part based models have been successful in combination with a maximum margin classifier (Felzenszwalb et al., 2010), and remain popular in the present days.

While a lot has been achieved by careful development of sophisticated structural models, one has to realize important limitations of this approach. First, it is hard to give a precise definition of a part. One faces an interesting tradeoff between having parts that are too instance specific, which would not generalize well, and parts that are too generic, which would undermine the ability to discriminate the object classes or lead to unreasonably complex models. Second, many visual categories do not naturally decompose into parts, at least they may not decompose into parts that are semantically meaningful to us. Third, the shape model is typically restricted to a tree or a star-graph to enable efficient inference (Felzenszwalb and Huttenlocher, 2005). Finally, the main challenge remains to identify the parts in the image, which is, in essence, a recognition task on its own.

Invariant Features. An established approach to object recognition assumes the existence of certain invariant features, or properties, that are common to a given category and remain stable under a variety of transformations, including translation,

[2] Marr and Nishihara, (1978) write (emphasis added) "We view recognition as a gradual process proceeding *from the general to the specific*, that overlaps with, guides, and constrains the derivation of a description from the image". In contrast, inductive inference in modern learning systems proceeds from specific examples to the general concept.

(limited) rotation, scaling, and even certain appearance variations. This approach can be traced back to Pitts and McCulloch, (1947)[3] and the early computer vision work of Hu, (1962), which used 2D moment invariants to recognize characters.

The study of invariants in vision can be arranged into two large groups of approaches: the ones based on geometry (Besl and Jain, 1985; Ullman et al., 1996) and appearance (Murase and Nayar, 1995; Schmid and Mohr, 1997). A notable example from the first group is the concept of geometric invariance (Mundy, 2006), which studies the properties of an object that do not vary with viewpoint, e.g. the ratio of collinear segment lengths. Unfortunately, it was shown that there exist no viewpoint invariant features in the general case, there are only special-case invariants for restricted configurations of 3D points (Burns et al., 1992).

Geometric invariants reveal an important issue in practical recognition systems: sensitivity to noise (Meer et al., 1998). In hindsight, one could argue that handcrafted features tend to use the minimal number of measurements and, therefore, are lacking the redundancy to be robust against outliers. In the context of machine learning, the insensitivity of a learning algorithm to small changes in the input is known as stability, and was shown to be both necessary and sufficient for learning (Bousquet and Elisseeff, 2002). While regularization is an established technique to ensure classifier stability in the RERM framework, one has to appreciate the challenge of obtaining good features that maintain stability under various transformations and noise conditions.

In contrast to the geometry based approach, appearance methods enforce little or no geometric constraints. Early approaches include, for example, histogram matching (Schiele and Crowley, 1996; Swain and Ballard, 1991) and eigenspace matching (Turk and Pentland, 1991). Instead of constructing semantically interpretable parts and geometry-inspired models, the focus is shifted towards computing noise resistant holistic image descriptors. When a set of images of an object is obtained by varying pose and illumination in small increments and then projected onto the dominant eigenvectors, Murase and Nayar, (1995) observed that the corresponding points lie on a low dimensional manifold. Notably, small variations in the input now cause limited variations in the output and lead to a robust recognition system.

The assumption that data lies on a low dimensional manifold has been very popular in the machine learning community (Belkin and Niyogi, 2003; Hein and Maier, 2006). While it may be an adequate tool to handle viewpoint and illumination variations, real world images pose further challenges, such as scale change, partial visibility, occlusion, and background clutter. A multi-scale representation is often employed to reduce sensitivity to scale changes (Lindeberg, 2013), whereas local features provide robustness against partial occlusion (Schmid and Mohr, 1997).

Bag of Words. The use of dense local features sampled at a large number of locations has lead to efficient object recognition in cluttered real world scenes. A particularly successful feature generation method called the Scale Invariant Feature

[3] Interestingly, a contemporary goal in computer vision some 70 years later is still quite accurately summarized by Pitts and McCulloch, (1947): “We seek general methods for designing nervous (sic) nets which recognize figures in such a way as to produce the same output for every input belonging to the figure”.

Transform (SIFT) was developed by Lowe, (1999). The SIFT features demonstrate invariance to changes in illumination, scale, viewpoint, affine distortion, and additive noise. As such, they allow matching between different views of the same instance (Lowe, 2004) and enable object classification (Mikolajczyk et al., 2005).

Inspired by the bag of words representation for text categorization (Joachims, 1998), Csurka et al., (2004) and Sivic and Zisserman, (2003) introduced visual analogies of words and vocabularies that are computed from local descriptors. Following their methodology, an image is characterized by a histogram of visual word counts, which yields a global feature vector suitable for training a classifier. While technical specifics of each step in the recognition pipeline have changed significantly in the last decade (Chapter 2), the general idea of aggregating a large number of relatively simple local measurements into a more complex global representation has endured to the present day.

Extraction of invariant features and computation of the feature vector representing an image are important ingredients in the classification pipelines that we consider in Chapters 4–6. Although we focus on the learning objectives and mainly consider the input space fixed, the choice of an image descriptor and certain engineering decisions in a pre-processing phase often play a decisive role in achieving state of the art recognition performance. Furthermore, the adoption of deep learning methods (Krizhevsky et al., 2012) and end-to-end training, where features and the classifier are learned jointly, blurs the distinction between the feature extraction and classifier training steps.

Kernel Methods. The performance of machine learning algorithms depends heavily on the representation of the data they are given. The kernel methods (Lampert, 2009; Schölkopf and Smola, 2002), which we mainly employ in the thesis, require every image[4] $x \in \mathcal{X}$ to be embedded in a dot product space $\mathcal{H}$, also called the feature space, using a feature map $\Phi : \mathcal{X} \to \mathcal{H}$. Once we have a vectorial representation of images, we can focus on training classifiers and the analysis of different learning algorithms. However, the choice of the feature map remains largely unspecified. Moreover, the focus has been traditionally rather on designing the kernel, $k(x, x') = \langle \Phi(x), \Phi(x') \rangle$, which induces the feature map implicitly.

The freedom to define the notion of similarity between images using a kernel function, as well as the freedom to design the corresponding feature maps, has fueled research in the computer vision community for decades. We have already mentioned the study of invariant features and the emergence of the bag of words model as notable examples from that era. However, there is an important limitation to that approach. When the individual steps in the classification pipeline are designed, or hand crafted, separately using manual supervision at each stage, there is no guarantee that local decisions lead to an improvement in the overall recognition performance. Just like it is virtually impossible for humans without specialized tools to write down a good classifier for high dimensional data, it is difficult to design a good feature extraction method. Furthermore, manual design that relies on strong prior knowledge assumptions inevitably introduces a strong bias as well.

[4] In the following, we omit the explicit mention of images and use x, $\mathcal{X}$ to denote a feature vector and the feature space respectively.

Neural Networks. An alternative approach based on the concept of artificial neural networks, which can be dated back to the early works of McCulloch and Pitts, (1943) and Rosenblatt, (1958), has become a de facto standard in modern computer vision. The main idea relevant to our discussion is that of representation learning (Bengio et al., 2013). Instead of designing features, the focus is on designing algorithms that learn useful features from raw input, such as directly from image pixels[5]. As before, lower-level patterns are progressively combined into a higher-level representation. However, in contrast to the previous approach, the parameters of every processing module at each layer are now subject to joint training via error backpropagation (Rumelhart et al., 1986). As a result, the emerging features are tailored to the given task and generally outperform their hand-crafted counterparts.

A particularly successful idea in the context of neural networks for image recognition is based on the basic understanding of the topological structure of images and the introduction of local receptive fields (Hubel and Wiesel, 1959). The idea of sweeping a local feature extractor over the image corresponds to a convolution, which is at the heart of convolutional networks (ConvNets). The Neocognitron of Fukushima and Miyake, (1982) was, perhaps, the first ConvNet to demonstrate invariance to shifts in position and shape distortion, while LeCun et al., (1989) were the first to train the convolutional filters using backpropagation.

Semantic Interpretation. So far, we have mainly focused on the description of image content in terms of feature vectors which correspond to the elements of the input space $\mathcal{X}$. Next, we consider the issues associated with the output space $\mathcal{Y}$.

Image recognition aims at understanding the semantic content of images. More specifically, it aims at capturing semantic interpretations of images that are relevant to the application. Naturally, such interpretations form only a subset of all possible interpretations of the image, which is known as weak semantics (Smeulders et al., 2000). Therefore, one can already anticipate that image categorization is necessarily inexact in many real world applications. To highlight the disconnect between an automated image representation and possible human interpretations, Smeulders et al., (2000) introduced the concept of the *semantic gap*:

> The semantic gap is the lack of coincidence between the information that one can extract from the visual data and the interpretation that the same data have for a user in a given situation.

Striving to bridge the gap, a significant portion of research has been focused on the introduction of learning algorithms and the design of better image descriptors (Lew et al., 2006). For example, Li and Wang, (2003) developed automatic linguistic indexing of pictures, where a collection of concept classifiers was used to tag images with the relevant labels. Vogel and Schiele, (2007) introduced a semantic image representation based on the histogram of local concept occurrences. Lampert et al., (2009) established an approach to high-level image description based on attributes. Finally, Li et al., (2009) made significant strides towards semantic

[5] However, there is still a pre-processing step that usually involves rescaling and mean subtraction, but may be also followed by PCA/ZCA whitening (Krizhevsky, 2009).

scene understanding with a unified framework for classification, annotation, and segmentation of images.

Today, advances in training deep neural networks demonstrate impressive results on many visual recognition tasks (He et al., 2016). With the sufficient amount of annotated training data, the learned representations have been able to accommodate most of the tasks that have been formalized so far. While that observation has sparked interest in pursuing new tasks that have not been considered before, it is also important to revisit the general problem of image recognition and understand which limitations of the classical tasks can be addressed today.

The issue, of course, is that recognition in the broad sense is an ill-posed problem. When defining a task for a particular application, the main challenge is to capture the relevant image semantics in a convenient, compact form, and formulate the prediction goal precisely. One of the most successful and well studied approaches relies on a linguistic description of images in terms of keywords and captions, which, in particular, leads to the multiclass and multilabel classification problems considered in this dissertation.

Class Ambiguity. However natural, image tagging suffers from two well-known limitations: the annotation cost and incomplete labeling. Often, image datasets provide only a single label per image, which means there is just one "correct" associated category. Increasing the number of classes makes it more difficult to discriminate between them, while attempting to cut the annotation costs introduces label noise. Together, these issues lead to what we call the *class ambiguity* problem.

Figure 1.1 illustrates the problem on a modern dataset of scene images, where the task is to predict the ground truth label associated with the scene. One can appreciate that this exercise is difficult for humans, since there are multiple labels that could be reasonably associated with every picture. Moreover, the choice of *the* label often seems random as it does not always correspond to the most dominant or salient object in the image. Naturally, a human would need several guesses to enumerate suitable labels before naming the "right" one.

While class ambiguity could be conceivably reduced by investing into obtaining a better, richer annotation, we argue that a certain level of ambiguity is inevitable due to the fundamental difficulty of assigning a unique, or enumerating all possible semantic interpretations of an image. Therefore, the main challenge that we aim to address in the context of computer vision is to enable object and scene image categorization in the presence of class ambiguity in the ground truth annotation.

1.2 Contributions

In this section, we briefly discuss the main contributions of this thesis. Most of the learning methods that we consider are based on the soft margin support vector machine (SVM) of Cortes and Vapnik, (1995), multiclass SVM of Crammer and Singer, (2001), logistic regression and maximum entropy methods (Berger et al., 1996; Friedman et al., 2001), and their extensions.

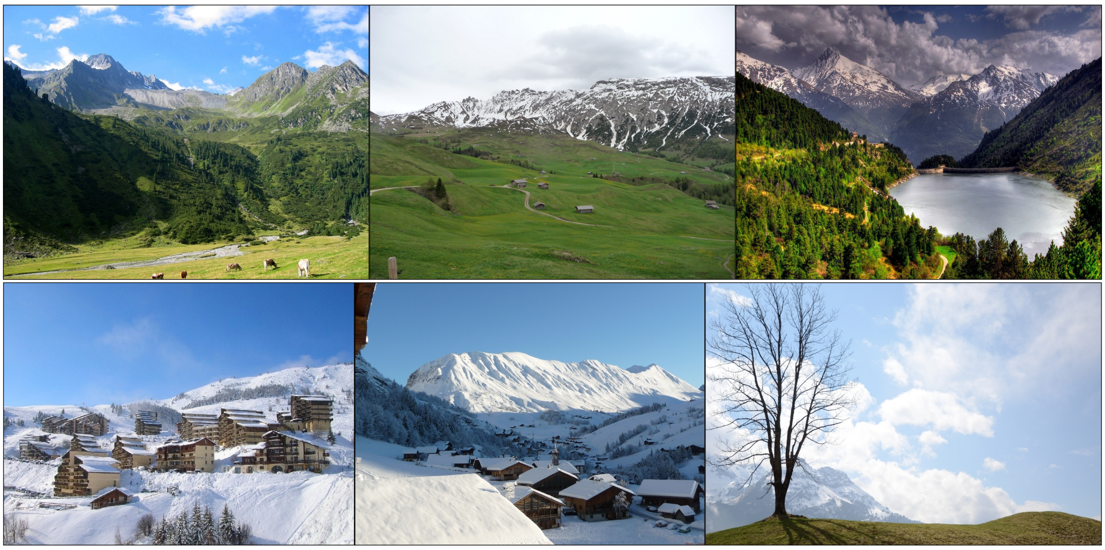


Figure 1.1.: Class ambiguity with a single label on the Places 205 (Zhou et al., 2014). **Labels:** Valley, Pasture, Mountain (top); Ski resort, Chalet, Sky (bottom). Note that multiple labels apply to each image and k guesses may be required to guess the ground truth label.

Analysis of Loss Functions and Learning Objectives

We analyze the following learning algorithms, the associated optimization problems, and the asymptotic property of classification calibration for the top-k error.

SVM+. We analyze uniqueness of SVM+ solutions and establish its close connection to the SVM with instance weights (WSVM). In particular, we show that any SVM+ solution can be reproduced by a WSVM with appropriately chosen weights. This observation enables the interpretation of privileged information in SVM+ as the guiding prior knowledge about which training examples are "easy" and which are "hard".

WSVM. We show how the weights for WSVM can be set from an SVM+ dual solution in such a way that WSVM reproduces the same solution. That demonstrates that the WSVM algorithm is at least as flexible as the SVM+. Furthermore, we reveal a constraint on the admissible SVM+ solutions which leads to the conclusion that not every WSVM solution can be constructed by SVM+, and we provide a specific counter example. Therefore, the WSVM algorithm is strictly more flexible and there may be interesting choices of the weights that are not covered by SVM+.

Top-k SVM. We introduce top-k SVM as a novel method to optimize for the top-k error performance metric, which allows k guesses instead of one. We elaborate on the connection between the proposed top-k hinge loss, a family of ranking losses by Usunier et al., (2009), and the general problem of learning to rank. Overall, we consider 4 versions of the top-k hinge loss: α/β and smooth/nonsmooth, where the α version is a tight convex upper bound on the top-k error, the β version belongs to the family of Usunier et al., (2009), and smoothing is done using Moreau-Yosida regularization.

Softmax and Top-k Entropy. We introduce the top-k entropy and the truncated top-k entropy loss functions as extensions of the classical softmax loss for the purpose of top-k error optimization.

Top-k Calibration. We introduce the notion of top-k calibration and analyze which of the multiclass methods, including the one-vs-all reduction schemes, are calibrated for the top-k error. In particular, we highlight that the softmax loss is uniformly top-k calibrated for all $k \geq 1$, which could explain its strong performance in top-k error for multiple values of k.

Optimization and Projection Algorithms

We develop efficient optimization schemes for the learning algorithms considered in the thesis. Most of the optimization algorithms are based on stochastic dual coordinate ascent (SDCA) method of Shalev-Shwartz and Zhang, (2013b).

Weight Learning. We develop a method to learn the instance weights for WSVM directly from data when there is a large validation sample available in addition to the training set. Our experiments demonstrate significant performance improvements if the split between the training and the validation sets favors the latter. These results prove the existence of nontrivial instance weights that are particularly helpful when the training data is limited.

Multitask Learning. We develop a multitask representation learning method with stochastic optimization in the dual space based on SDCA. All updates of the dual variables are computed in closed form, which makes the optimization scheme efficient. The approach is particularly attractive in the situations where the number of training examples is limited, but the dimensionality of the feature space is large. A linear mapping into a lower dimensional subspace that is jointly learned with the individual classifiers, provides additional regularization and enables the inter-class information sharing, which ultimately improves the recognition performance. The source code of our implementation is publicly available[6].

Multiclass, Top-k, and Multilabel Learning. We employ the SDCA framework for vector valued loss functions to develop algorithms for multiclass, top-k, and multilabel learning. Furthermore, we use Moreau-Yosida regularization to introduce smoothed formulations of multiclass, top-k, and multilabel SVMs. At the heart of our optimization schemes are efficient algorithms for computing the Euclidean and a biased projection onto the effective domain of the convex conjugate of the corresponding loss function. The source code of our implementation is publicly available[7].

[6] http://github.com/mlapin/cvpr14mtl

[7] http://github.com/mlapin/libsdca

Multitask Output Kernel Learning. Our work on efficient output kernel learning (Jawanpuria et al., 2015) is not included in this thesis, however the author contributed towards the implementation of the SDCA based method for learning the output kernel matrix with ℓ_p-norm regularizers. The source code of our implementation is publicly available[8].

The main ingredient in the SDCA framework is the update of the dual variables, which, for the vector valued loss functions, corresponds to a biased projection onto the effective domain of the conjugate loss. We develop efficient algorithms for computing the Euclidean and a biased projection onto the following convex sets.

Top-k Simplex (α) is as an interesting extension of the standard simplex, where every variable is upper bounded by $1/k$ of the total sum of all variables. It is the effective domain of the convex conjugate of the top-k hinge loss (α).

Top-k Cone is obtained by removing the upper bound on the sum of the variables in the top-k simplex (α), but keeping the bounds on each individual variable. The set is analogous to the positive orthant in the definition of the standard simplex, but has a more complicated structure.

Top-k Simplex (β) corresponds to the effective domain of the convex conjugate of the top-k hinge loss (β). The Euclidean (non-biased) projection reduces to the well-known continuous quadratic knapsack problem.

Bipartite Simplex emerges as the effective domain of the convex conjugate of the multilabel SVM loss of Crammer and Singer, (2003). We offer the term *bipartite simplex* as it appears in multilabel learning, where the labels are partitioned into positive and negative, and, therefore, correspond to a bipartite label ranking graph. Our novel projection algorithm based on the variable fixing scheme is particularly efficient compared to the prior work.

Entropic Projections onto the top-k simplex (α) arise in our optimization schemes with the softmax loss and the top-k entropy loss. In these cases, the presence of the entropy function complicates computation of SDCA updates and one usually resorts to an approximation. Instead, we propose to use the Lambert W function, which is evaluated efficiently, and develop an algorithm to directly compute the corresponding entropic projections.

Empirical Evaluation and Engineering Decisions

We perform extensive empirical evaluation of the proposed methods, compare them to prior work, and share our insights and observations.

Scalability. We demonstrate the efficiency of our learning methods and the proposed projection algorithms. In particular, we report the running times and present scalability plots with respect to the input dimension.

[8] http://www.ml.uni-saarland.de/code/FMTL-SDCA/FastMTL-SDCA.zip

OVA vs Multiclass. Although often comparable, the performance of direct multiclass methods is found in our experiments to be generally superior to the one-vs-all (OVA) scheme with binary classifiers. The differences are particularly noticeable for the OVA SVM, and in top-k error for $k > 1$.

Top-k Error Optimization. We show that our proposed top-k SVM outperforms competing ranking based methods in top-k error, especially for $k > 1$. Softmax loss and smooth multiclass SVM are shown to perform remarkably well in top-k error uniformly for all k, while further improvements are possible with the proposed top-k extensions.

Multiclass to Top-k to Multilabel. We explore the transition from multiclass to multilabel learning by comparing the performance on multilabel datasets between multiclass, top-k, and multilabel methods. The multiclass and top-k methods use only a single label of the most prominent object for training. As expected, the top-k methods outperform their multiclass counterparts in the presence of class ambiguity.

Finally, we confirm the previous observations regarding data augmentation and feature engineering decisions.

Data Augmentation. We demonstrate in a handwritten digit recognition experiment that data augmentation by horizontal and vertical translation helps and can be successfully combined with other forms of prior knowledge, such as the learned instance weights.

Feature Engineering. We explore a number of feature engineering decisions for scene classification using the Fisher Vector representation (Chapter 2). Our observations demonstrate the importance of careful feature design and complement similar findings of Sánchez et al., (2013).

1.3 Outline

The dissertation is arranged in two parts which are summarized below.

Part I consists of Chapters 3 and 4, and is primarily concerned with learning in the regime of limited training data. We explore two learning frameworks that have been developed to improve the predictive performance of learning algorithms by introducing additional prior knowledge, such as the privileged information and the task relatedness.

Part II consists of Chapters 5 and 6, and addresses the challenge of class ambiguity in modern large scale datasets. We note that multiple labels often apply to any given image, while the ground truth annotation contains only a single label. In this part, we explore learning using such singleton labels when the examples are multilabel in nature, and advocate for an adjusted performance metric, the top-k error, which allows k prediction attempts instead of one.

Next, we briefly summarize the content of each chapter.

Chapter 2 reviews related research and prior work with the goal of providing the context for the topics considered in this thesis. In particular, we are interested in establishing interdisciplinary links and highlighting the connections that exists between different research directions.

Chapter 3 considers the framework of learning using privileged information, which was introduced by Vapnik and Vashist, (2009), and explores its relation to learning with instance weights. We investigate the effect of correcting the loss on training data by adjusting the instance weights and observe performance improvements when the weights are learned on a large validation sample.

Chapter 4 presents an algorithm for multitask representation learning, where we examine if learning a representation that is shared across a large number of classes improves the classification of scene images. Moreover, we investigate scalability of multitask learning to high dimensional feature spaces and propose an efficient optimization scheme based on SDCA.

Chapter 5 discusses the problem of class ambiguity that arises in large scale datasets, and suggests that the top-k error is an appropriate target performance measure. We propose top-k multiclass SVM as a suitable learning algorithm for the top-k objective, and discuss an efficient algorithm for the Euclidean and a biased projection onto the top-k simplex. The latter enables top-k SVM optimization within the SDCA framework.

Chapter 6 extends the analysis of class ambiguity and top-k error optimization along multiple directions. In particular, we introduce smooth top-k SVM and top-k extensions of the softmax loss, analyze top-k calibration of multiclass methods, consider the transition from multiclass to multilabel learning, and propose smooth multilabel SVM. We discuss SDCA optimization of the considered methods, contribute novel projection algorithms, and perform an extensive empirical evaluation on multiclass and multilabel datasets.

Chapter 7 summarizes this dissertation, discusses the insights and conclusions, highlights certain limitations, and provides an outlook for further research.

Appendix A covers the basics of convex analysis and the necessary mathematical background. In particular, we recall the Lagrangian and Fenchel duality, define the convex conjugate, recap its properties, and give examples of convex functions that are used in the thesis.

Appendix B provides additional details are further results from Chapter 4.

Related Work 2

In this chapter, we place the contributions of this thesis in the broad context of related research. We aim to give a brief summary of the prior work, discuss the novelty of our contributions, and review how the field has advanced since then.

We start with a review of image classification pipelines in Section 2.1 with the focus on image representation. Section 2.2 surveys the literature on learning using privileged information, while Section 2.3 covers the optimization framework. Section 2.4 is concerned with the research on multitask learning, and finally Section 2.5 covers top-k and multilabel classification. Moreover, the last section establishes a connection to label ranking and discusses theoretical analysis of consistency and calibration of surrogate loss functions.

2.1 Image Classification

In this section, we consider three specific examples of the general image classification problem: handwritten digit recognition, visual object recognition, and scene categorization. The first problem, recognition of handwritten digits, is the most basic among the three in the sense that the classifiers can be trained directly on vectors of pixel values. It is the running example of Chapter 3, where we focus on learning using privileged information.

Recognition of visual objects and scene categories in real world images is a far more challenging task due to the challenges discussed in Chapter 1, such as the variability in appearance, viewpoint, and scale, the presence of background clutter, partial visibility and occlusion. Moreover, real world images are high dimensional (on the order of 10^5 and more), which rules out classifier training directly on pixel values due to the problems associated with the curse of dimensionality. Instead, we review approaches that exploit prior knowledge to build successful image representations: the bag of words (BOW) model, the Fisher vector (FV) encoding, and the convolutional neural networks (ConvNets). We consider object recognition in Chapters 5–6 and scene categorization in Chapters 4–6.

Handwritten Digit Recognition and the more general optical character recognition (OCR) have been extensively studied in the literature, see the reviews e.g. by Liu et al., (2003), Suen et al., (1992), and Trier et al., (1996). It was one of the early successful applications of neural networks (LeCun et al., 1989, 1998, 1995), until support vector machines (SVMs) became competitive on this task using the methods for incorporating prior knowledge (Decoste and Schölkopf, 2002; Schölkopf et al., 1996), such as engineering of invariant kernel functions and generation of artificially transformed examples, i.e. data augmentation. The latter was also explored in the context of convolutional neural networks, where Simard et al., (2003) introduced elastic distortions outperforming the previous SVM based approaches.

MNIST (LeCun et al., 1998) is the classical dataset for handwritten digit recognition which will become relevant in Chapters 3 and 4. The state of the art error rate on this dataset is well below 1% – Cireşan et al., (2012) and Wan et al., (2013) report respectively 0.23% and 0.21% error rate – which makes it a popular “toy” benchmark for novel learning methods and regularization techniques. In particular, it is relatively common to deviate from the established evaluation protocol on MNIST, which makes the performance numbers not directly comparable. In Chapter 3, we follow Vapnik and Vashist, (2009), who consider the problem of classifying the digits 5 and 8 from the MNIST dataset, which are additionally downsized from 28 × 28 to 10 × 10 pixels to make the problem more challenging. In Chapter 4, we follow the protocol of Kang et al., 2011, where a subset of the MNIST and USPS (Hull, 1994) examples are selected and preprocessed with PCA reducing the dimensionality to retain about 95% of the variance.

Bag of Words. The early work on appearance based object recognition, which may be considered the precursor to the bag of words model, utilized global image descriptors based on color and texture histograms (Niblack et al., 1993; Schiele and Crowley, 1996; Swain and Ballard, 1991). While the initial approaches were sensitive to the natural sources of appearance variability, the methods based on local invariant features increased the robustness to partial occlusion (Schmid and Mohr, 1997), scale changes (Fergus et al., 2003), and affine deformations (Lazebnik et al., 2004). Scale invariant feature transform (SIFT) of Lowe, (2004) and histograms of oriented gradients (HOG) of Dalal and Triggs, (2005) are notable examples of methods for local feature extraction that were particularly popular before the resurgence of convolutional neural networks.

The orderless bag of words (BOW) model originates in the domain of text document classification (Joachims, 1998; McCallum, Nigam, et al., 1998). Sivic and Zisserman, (2003) introduced the visual analogy of words and applied the obtained model to object matching in videos. Csurka et al., (2004) proposed the related bag of keypoints method for generic image categorization, while Sivic et al., (2005) further popularized the approach demonstrating a successful application of probabilistic topic modeling in the visual domain.

With the increased popularity of support vector machines in computer vision, a related research direction explored the design of invariant kernels based on local feature sets (Wallraven et al., 2003). In particular, Chapelle et al., (1999) considered a specialized form of radial basis function (RBF) kernel tailored to high dimensional histograms, while Barla et al., (2003) demonstrated the use of histogram intersection as an SVM kernel function for image classification. Subsequently, Grauman and Darrell, (2005) proposed the pyramid match kernel that approximates the partial matching between two feature sets. That idea was further developed by Lazebnik et al., (2006), who introduced the spatial pyramid image representation and demonstrated its superiority in recognizing natural scenes.

The kernel view of low-level image features, such as SIFT and HOG, was highlighted by Bo et al., (2010), who proposed match kernel descriptors based on gradient, color, and shape information. In (Bo et al., 2011), the approach was developed further to hierarchical feature learning, where the kernel descriptors are applied recursively. Their method is reminiscent of a three layer neural network and is one of many examples where one can recognize a connection between deep learning and approaches based on kernel methods (Cho and Saul, 2009; Perronnin and Larlus, 2015; Sydorov et al., 2014).

Finally, a parallel line of research investigated scalability of kernel SVMs to large image classification problems. Maji and Berg, (2009) introduced an explicit feature map approximating the histogram intersection kernel, which enabled significant reductions in training and test times compared to the corresponding kernel SVM. Vedaldi and Zisserman, (2012) generalized computation of explicit feature maps to a family of additive positive definite kernels, which includes the intersection, χ^2, and Hellinger's kernels. We use their method in Chapter 4 along with the Fisher vector representation discussed next.

Fisher Vector. The classical bag of words model computes a histogram of words by averaging the occurrence counts represented as one-hot vectors. This aspect of the BOW model, known as local feature encoding, seemed suboptimal and generated interest in finding better coding techniques. Soft assignment replaces the binary one-hot vector with a real-valued distribution over the closest words from the dictionary and models a neighborhood around the local feature (Perronnin et al., 2006; Van Gemert et al., 2010; Winn et al., 2005). While soft assignment typically outperforms the classical hard assignment corresponding to the mode of that distribution, the method does not make the most efficient use of the dictionary. Chatfield et al., (2011) observed that recognition performance increased with diminishing returns as they increased the size of the dictionary, but was likely not saturated even at 25K visual words in dictionaries for soft assignment methods. Sparse coding, in contrast, finds the closest subspace spanned by a linear combination of a few words from the dictionary which represent the local feature well (Boureau et al., 2010; Wang et al., 2010; Yang et al., 2009). Still, the use of large over-complete dictionaries has been found to be important for improved performance (Chatfield et al., 2011), leading to increased computational costs of the encoding step. Boureau et al., (2010), Wang et al., (2010), and Yang et al., (2009) also demonstrated performance improvements using max pooling instead of average pooling. Finally, Mairal et al., (2012) studied supervised dictionary learning and considered loss functions alternative to the ℓ_2 distance criterion.

Sánchez et al., (2013) demonstrated strong performance improvement over the previous methods using a feature encoding scheme based on the Fisher kernel. Combining the benefits of generative and discriminative approaches, the Fisher kernel (Jaakkola, Haussler, et al., 1999) characterizes an input vector by its deviation from the generative model. Specifically, it computes the gradient of the empirical log-likelihood with respect to the parameters of a generative model, such as a Gaussian mixture model. The approach is known as the Fisher vector (FV) encoding and was popularized in computer vision by Perronnin and Dance, (2007) and Perronnin et al., (2010). The FV encoding was instrumental to achieving state of the art performance in visual object and scene classification (Chatfield et al., 2011; Juneja et al., 2013), yet one of its major shortcomings is that the obtained descriptors are dense and high dimensional. To address the issue of high computational costs, Sánchez et al., (2013) compressed the FVs significantly reducing their dimensionality using the product quantization technique (Gray and Neuhoff, 1998; Jégou et al., 2010). While that reduces the memory footprint and the computational costs for training linear SVMs (solving the primal problem), it also has a negative effect on the recognition performance. Instead, we use dual optimization in Chapter 4 and work with Gram matrices that are computed directly from high dimensional Fisher vectors (up to 260K dimensions), which is well justified for small and moderately sized training samples.

Convolutional Neural Networks (ConvNets) have led to a series of breakthroughs in image classification starting with a leap in recognition accuracy achieved by Krizhevsky et al., (2012) on a large scale ImageNet dataset (Russakovsky et al., 2015). Originally introduced by LeCun et al., (1989) for the problem of handwritten character recognition, ConvNets have been successfully applied to various vision tasks, including categorization of textureless toys (LeCun et al., 2004), classification of house number digits (Sermanet et al., 2012), recognition of Chinese characters and traffic signs (Cireşan et al., 2012), and object recognition (Jarrett et al., 2009).

Improving upon the original architecture of Krizhevsky et al., (2012), Sermanet et al., (2014) and Zeiler and Fergus, (2014) utilized smaller receptive fields and used denser strides in convolutional layers. However, Simonyan and Zisserman, (2015) demonstrated using an architecture with very small (3×3) convolutional filters that depth may be an even more important factor in designing ConvNets – their VGGNet architectures show consistent increase in performance as the network becomes deeper, from 11 to 16 to 19 layers. A similar result advocating the use of deep architectures is due to Szegedy et al., (2015) whose GoogLeNet (Inception-v1) has 22 layers. More recently, He et al., (2016) proposed a residual network (ResNet) architecture with 152 (and even with over 1000) layers, which utilizes residual learning with shortcut connections.

Image representations learned by ConvNets on large datasets have been observed to transfer well (Oquab et al., 2014; Razavian et al., 2014). In our multiclass and multilabel image classification experiments, we employ a relatively simple image recognition pipeline following Simonyan and Zisserman, (2015), where feature vectors are extracted from a convolutional network, such as VGGNet or ResNet, and are then used to train a linear classifier with the different loss functions. The ConvNets that we use for object classification are pre-trained on the ImageNet dataset (Russakovsky et al., 2015), where there is a large number of object categories (1000), but relatively little variation in scale and location of the dominant object. For scene recognition, we also use a VGGNet-like architecture of Wang et al., (2015a) that was trained on the Places 205 dataset (Zhou et al., 2014).

Deep learning architectures in general and convolutional networks in particular are introduced in (Bengio, 2009; Goodfellow et al., 2016), while Schmidhuber, (2015) provides a historical survey of the relevant work.

2.2 Learning Using Privileged Information

In this section, we set the stage for Chapter 3 by reviewing research on learning using privileged information (LUPI). As we demonstrate a connection to learning with instance weights, we also discuss the literature on weighted learning.

LUPI Framework. Since the introduction of the new learning paradigm and the corresponding SVM+ algorithm originally by Vapnik, (2006) and later by Vapnik and Vashist, (2009) and Vapnik et al., (2009), there is a steady interest in the LUPI framework. Liang and Cherkassky, (2008) and Liang et al., (2009) studied the relation between SVM+ and multitask learning. Pechyony and Vapnik, (2010) considered a generalized version of the SVM+ algorithm and analyzed the corresponding risk bound. Pechyony and Vapnik, (2011) adapted the sequential minimal optimization (SMO) algorithm of Platt, (1999) for SVM+ training. Niu et al., (2012) used ℓ_1 regularizer instead of ℓ_2 in SVM+, Liu et al., (2013) developed a multiclass SVM algorithm exploiting privileged information, while Ji et al., (2012) adopted privileged information in training multitask multiclass SVMs. Fouad et al., (2012) applied the LUPI framework to metric learning, Chen et al., (2012) extended it to boosting algorithms, and Yang and Patras, (2013) trained regression forests for facial feature detection. Feyereisl and Aickelin, (2012) used the privileged information for data clustering and Levy and Wolf, (2013) proposed an SVM$\ominus$ method to compute similarity scores in video face recognition. Note, however, that the latter method is not related to the SVM$-$ algorithm we have in mind in § 3.4.5. In particular, SVM$\ominus$ reduces to SVM with a pre-processing step, similar to (Schölkopf et al., 1998), while in our case the optimization problem as well as the motivation are entirely different.

Sharmanska et al., (2013) applied the LUPI paradigm in learning to rank and proposed a Rank Transfer maximum margin method which utilizes privileged information to adjust the margins when ranking pairs of training examples. This line of work was developed further in (Sharmanska et al., 2014), where a Margin Transfer method is applied to multiclass object classification with three types of privileged information: attributes, bounding boxes, and image tags. Hernández-Lobato et al., (2014) proposed a Bayesian method based on the framework of Gaussian process classifiers where the privileged data enters the model of the noise term and influences the confidence of the classifier on every training example. Li et al., (2014b) exploited privileged information in multiple instance learning extending the method of Bunescu and Mooney, (2007), and Chen and Kamarainen, (2014) developed a learning to count method with back-propagated information.

Developing the LUPI framework further, Vapnik and Izmailov, (2015) considered a general concept of intelligent teacher who transfers knowledge to a student. While SVM+ is a particular instantiation of that approach, the key idea models an asymmetric interaction between a teacher, who has access to privileged information, and a student, who is learning a decision rule. Subsequently, Lopez-Paz et al., (2015) pointed out a similarity with the distillation approach to knowledge transfer of Hinton et al., (2015), and proposed a unified view of generalized distillation. Concurrently, Xu et al., (2015) proposed a novel formulation for metric learning with privileged information, Ren et al., (2015) considered multitask learning, while Wang and Ji, (2015) introduced the loss inequality and relationship preserving regularization techniques based on the use of privileged information.

Finally, Li et al., (2016) developed a fast optimization scheme for SVM+ based on dual coordinate ascent, Motiian et al., (2016) adapted the information bottleneck method of Slonim et al., (2006) to utilize privileged information, and You et al., (2017) introduced a privileged multilabel learning method.

Weighted Learning. As we demonstrate a relationship between SVM+ and SVM with instance weights in Chapter 3, we give here a brief overview of the research directions where learning with instance weights is a natural technique.

Cost-sensitive learning is one of the most prominent examples of weighted learning and is concerned with the situation where misclassification costs are nonuniform. While stratification, i.e. changing the frequency of classes in the training data in proportion to their cost, is one of the early attempts to obtain cost-sensitive classifiers (Breiman et al., 1984), the MetaCost method of Domingos, (1999) used nonuniform instance weights directly in the learning objective. Domingos, (1998) provides a brief overview of the early cost-sensitive learning approaches, while Elkan, (2001) and Sun et al., (2007) give a more recent outlook. In particular, Margineantu, (2002) developed a procedure for confidence based probability estimation, while Zadrozny et al., (2003) proposed a rejection sampling technique to obtain cost-sensitive classifiers. A closely related setting is that of learning with imbalanced data where certain classes are underrepresented. Chawla et al., (2004) and He and Garcia, (2009) provide a comprehensive survey of the literature complemented by López et al., (2013) and Sun et al., (2009).

Sample bias correction (Cortes et al., 2010; Dudík et al., 2005; Huang et al., 2007) and domain adaptation (Bickel et al., 2007; Shimodaira, 2000; Sugiyama et al., 2008) are further examples of challenges that can be addressed with importance weighting. Perhaps the most related in terms of the learning algorithm (SVM) and the interpretation of instance weights are the works on fuzzy SVM by Lin and Wang, (2002), where each data point has a fuzzy class membership represented by a weight between 0 and 1, weighted margin SVM of Wu and Srihari, (2004), where again each label has a confidence score between 0 and 1, and weighted SVM with an outlier detection pre-processing step of Yang et al., (2005), where a kernel-based clustering algorithm is used to generate instance weights.

2.3 Multitask, Top-k, and Multilabel Optimization

To facilitate experimental evaluation, we implement the necessary optimization routines for the learning methods proposed throughout Chapters 4–6. In this section, we provide a brief overview of the related literature on stochastic dual optimization, evaluation of projections and proximal maps, and discuss optimization of the logistic loss, which requires special care compared to the hinge loss.

SDCA. We mainly work with the stochastic dual coordinate ascent (SDCA) framework of Shalev-Shwartz and Zhang, (2013b) due to its ease of implementation, strong convergence guarantees, and the possibility to compute certificates of optimality with the duality gap. While Shalev-Shwartz and Zhang, (2013b) describe the general SDCA algorithm that we implement, their analysis is limited to scalar loss functions (both Lipschitz and smooth) with ℓ_2 regularization, which is only suitable for binary problems. A more recent work (Shalev-Shwartz and Zhang, 2014) extends the analysis to vector valued smooth and Lipschitz functions and general strongly convex regularizers, which is better suited to our multiclass and multilabel loss functions. A detailed comparison of recent coordinate descent algorithms is given by Fercoq and Richtárik, (2015) and Shalev-Shwartz and Zhang, (2014). Interesting extensions of the SDCA scheme include, for example, mini-batch optimization (Takác et al., 2013), importance sampling (Qu et al., 2015; Zhao and Zhang, 2015), and distributed optimization (Richtárik and Takác, 2016).

Euclidean Projection. Following Shalev-Shwartz and Zhang, (2014), the main step in the SDCA algorithm for vector valued loss functions performs an update of the dual variables by computing a projection or, more generally, the proximal operator of an appropriate function (Parikh and Boyd, 2014). The proximal operators that we consider can be equivalently expressed as instances of a continuous nonlinear resource allocation problem which has a long research history, see the surveys by Patriksson, (2008) and Patriksson and Strömberg, (2015). Most related to our setting is the Euclidean projection onto the unit simplex or the ℓ_1-ball in $\mathbb{R}^n$, which can be computed via breakpoint searching (Kiwiel, 2008a) and variable fixing (Kiwiel, 2008b; Michelot, 1986). The former can be done in $O(n \log n)$ time with a simple implementation based on sorting (Held et al., 1974), or in $O(n)$ time with an efficient median finding algorithm (Brucker, 1984; Kiwiel, 2007). In Chapters 5–6, we choose the variable fixing scheme which does not require sorting and is easy to implement. Although its complexity is $O(n^2)$ on pathological inputs with elements growing exponentially (Condat, 2014), the typical observed complexity in practice is linear and is competitive with breakpoint searching algorithms (Condat, 2014; Kiwiel, 2008b).

Projection algorithms have been also studied in the context of learning sparse models using projected gradient methods (Beck and Teboulle, 2009; Nesterov, 2014; Schmidt et al., 2011), since sparsity inducing norms often lead to problems with simple constraints, which are well addressed by optimization schemes involving projection subroutines (Bertsekas, 1982). In particular, efficient learning algorithms have been developed for the following norms: ℓ_2 by Shalev-Shwartz and Singer, (2006), $\ell_{2,1}$ by Liu et al., (2009), $\ell_{1,\infty}$ by Quattoni et al., (2009), and $\ell_{1,q}$ by Sra, (2011). Finally, Yu et al., (2012) considered regularization with an intersection of norm balls.

Logistic Loss and the Lambert W Function. While there are efficient projection algorithms for optimizing the SVM hinge loss and its variations, the situation is a bit more complicated for logistic regression, both binary and multiclass. There exists no analytical solution for an update with the logistic loss, and Shalev-Shwartz and Zhang, (2014) suggest a formula in the binary case which computes an approximate update in closed form. Multiclass logistic (softmax) loss is optimized in the SPAMS toolbox (Mairal et al., 2010) using the fast iterative shrinkage-thresholding algorithm (FISTA) of Beck and Teboulle, (2009). Alternative optimization methods are considered by Yu et al., (2011) who also propose a two-level coordinate descent method in the multiclass case. Different from these works, we propose to follow closely the same variable fixing scheme that is used for SVM training and use the Lambert W function (Corless et al., 1996) in the resulting entropic proximal map. Our runtime compares favorably with SPAMS, as we show in § 6.5.2.

The key ingredient in our approach is fast evaluation of the Lambert W function of the exponent, which is closely related to evaluation of the principal $W_0(z)$ branch of the multivalued W function. Early algorithms have been proposed by Barry et al., (1995) and Fitsch et al., (1973). Analytical approximations of the function along with a literature survey are given by Barry et al., (2000), while Chapeau-Blondeau and Monir, (2002) propose a rational function approximation scheme. Our implementation is based on the more recent works by Fukushima, (2013) and Veberič, (2012).

2.4 Multitask Learning

This section provides the context for Chapter 4 where we propose a multitask feature learning method for scene classification. We review some of the classical and most related papers on this topic, discuss the novelty of our approach, and explore more recent research directions.

MTL Foundations. In multitask learning (MTL), relevant information is shared among the related tasks during training with the goal of improving the prediction performance of one or each of those tasks compared to training them independently (Baxter et al., 2000; Thrun and Pratt, 2012). Caruana, (1997) demonstrated the concept of multitask learning by training a neural network using supervision from several related tasks which shared a common representation. The idea is particularly befitting deep neural networks where weight sharing occurs naturally in hidden layers. One of the early examples is a deep convolutional network of Collobert and Weston, (2008) for multitask feature learning in the domain of natural language processing.

On the theoretical side, Ben-David and Schuller, (2003) analyzed the notion of task relatedness and derived generalization error bounds for multitask learning, Evgeniou et al., (2005) extended the MTL framework to kernel methods, Ando and Zhang, (2005) developed a method for structured and semi-supervised multitask learning, Maurer, (2006) analyzed the Rademacher complexity of linear multitask learning, Pentina and Lampert, (2014) derived a PAC-Bayesian generalization bound for the expected loss on a future learning task in the lifelong learning scenario. The problem of learning the underlying structure between the tasks was investigated by Amit et al., (2007) and Kang et al., (2011). Moreover, convex formulations for multitask feature learning have been proposed by Amit et al., (2007), Argyriou et al., (2008), Chen et al., (2013), Jawanpuria and Nath, (2012), and Zhong and Kwok, (2012). Unrelated (orthogonal) tasks have been exploited by Romera-Paredes et al., (2012).

MTL-SDCA. Most related to our method in Chapter 4 is the work of Maurer et al., (2013) which investigates the application of sparse coding and supervised dictionary learning to multitask learning. In the optimization schemes that alternate between feature encoding and dictionary learning, the latter step is usually computationally more demanding. For regression problems, where the loss function is the squared Euclidean norm, there are efficient algorithms for learning the codebook, e.g. the K-SVD (Rubinstein et al., 2010, 2008). For classification problems with the hinge loss and the logistic regression loss, a common approach is a variation of (stochastic) gradient descent (Jenatton et al., 2011b; Mairal et al., 2012; Roux et al., 2012), which is not scalable to extremely high feature dimensions. We address this issue in Chapter 4, where we consider an MTL formulation and propose an efficient stochastic optimization algorithm optimizing the dual problem. Following a common approach (Ando and Zhang, 2005), we interpret OVA classifiers in a multiclass problem as individual related tasks and demonstrate superiority of our MTL method in scene classification compared to independent single task learning.

Deep Learning. The general concept of multitask learning is being successfully applied in various deep learning architectures. Zhang et al., (2014) developed a deep MTL method for facial landmark detection where auxiliary tasks helped improve the performance of the main task, Wang et al., (2015b) designed a deep network for surface normal estimation utilizing the auxiliary tasks of predicting a room layout and edge labels (convex, concave, occluding, no edge), Misra et al., (2016) investigated how to control the degree of sharing between the tasks in a deep ConvNet using cross-stich units, Dai et al., (2016) developed a network for instance-aware semantic segmentation by decomposing the main task into the subtasks of bounding box estimation, pixel-level mask segmentation, and mask categorization. Most recently, there are interesting attempts to learn a “universal” representation that would accommodate a larger selection of seemingly less related tasks spanning multiple datasets (Bilen and Vedaldi, 2017; Kokkinos, 2016).

Finally, we note that multitask learning is closely related to numerous other learning frameworks including, in particular, representation learning (Argyriou et al., 2007), transfer learning (Pan and Yang, 2010), lifelong learning (Thrun and Mitchell, 1995), and curriculum learning (Pentina et al., 2015).

2.5 Top-k and Multilabel Classification

We consider top-k and multilabel classification in Chapters 5–6, where the classifier is trained to produce a set of labels rather than a single label. Depending on the number of labels in the available annotation, we distinguish between top-k classification (1 label) and multilabel classification (many labels). Here, we draw connections to the general problem of learning to rank, and in particular to label ranking. We start with a brief review of ranking in the context of information search and retrieval, then we focus on label ranking and top-k classification, and finally conclude with a discussion of multilabel classification. Towards the end of the section, we also review related research on the theoretical analysis of surrogate losses, which provides the necessary context for our analysis of top-k calibration.

Learning to Rank. Ranking is a supervised learning problem that arises whenever the structure in the output space admits a (partial) order (Tsochantaridis et al., 2005). The classical example is ranking in information retrieval (IR), see e.g. (Liu, 2009) for a recent review. There, a feature vector $\Phi(q, d)$ is computed for every query q and every document d, and the task is to learn a model that ranks the relevant documents for the given query before the irrelevant ones. Three main approaches are recognized within that framework: the pointwise, the pairwise, and the listwise approach. Pointwise methods cast the problem of predicting document relevance as a regression (Cossock and Zhang, 2006) or a classification (Li et al., 2007) problem. Instead, the pairwise approach is focused on predicting the relative order between the pairs of documents (Burges et al., 2005; Freund et al., 2003; Joachims, 2002). Finally, the listwise methods attempt to optimize a given performance measure directly on the full list of documents (Taylor et al., 2008; Xu and Li, 2007; Yue et al., 2007), or propose a loss function on the predicted and the ground truth lists (Cao et al., 2007; Xia et al., 2008). More recent research is focused on the top of the ranked list (Agarwal, 2011; Boyd et al., 2012; Li et al., 2014a; Rakotomamonjy, 2012; Rudin, 2009). However, they are mainly interested in search and retrieval, where the number of relevant documents by far exceeds what users are willing to consider. That setting suggests a different trade-off for recall and precision compared to our setting with only a few relevant labels.

Ranking objectives have been also considered for training convolutional architectures (Gong et al., 2013), most notably with a loss on triplets (Wang et al., 2014a; Zhao et al., 2015), that considers both positive and negative examples, and in learning a hash function for multilabel image retrieval (Zhao et al., 2015).

Label Ranking. Different from ranking in IR, our main interest in this thesis is label ranking which generalizes the basic binary classification problem to multiclass, multilabel, and even hierarchical classification, see (Vembu and Gärtner, 2010) for a survey. A link between the two settings is established if we consider queries to be examples (e.g. images) and documents to be class labels. The main contrast, however, is in the employed loss functions and performance evaluation at test time (§ 6.2, page 110). To contrast the two, we note that (i) the actual ranking of labels is often used only to compute a partition, which is also reflected in the corresponding performance measures (see § 6.2.1); (ii) the number of relevant documents per query is usually much larger than the number of class labels per example, which suggests different trade-offs (e.g. precision vs. recall); (iii) feature vectors in classification are typically fixed and do not depend on the class label.

Most related to our work presented in Chapters 5–6 is a general family of convex loss functions for ranking and classification introduced by Usunier et al., (2009). One of the loss functions that we consider (top-k SVM$^{\beta}$) is a member of that family. Other examples are Wsabie of Weston et al., (2011) and Wsabie^{++} of Gupta et al., (2014), which learn a joint embedding model optimizing an approximation of a loss from (Usunier et al., 2009).

Top-k classification in our setting is directly related to label ranking as the task is to place the ground truth label in the set of top k labels as measured by their prediction scores. An alternative approach is suggested by McAuley et al., (2013) who use structured learning to aggregate the outputs of pre-trained OVA binary classifiers and directly predict a set of k labels, where the labels missing from the annotation are modeled with latent variables. That line of work is pursued further by Xu et al., (2016c). The task of predicting a set of items is also considered by Ross et al., (2013), who frame it as a problem of maximizing a submodular reward function. A probabilistic model for ranking and top-k classification is proposed by Swersky et al., (2012), while Guillaumin et al., (2009) and Mensink et al., (2013) use metric learning to train a nearest neighbor model. An interesting setting related to top-k classification is learning with positive and unlabeled data considered by Kanehira and Harada, (2016) and Plessis et al., (2014), where the absence of a label does not imply it is a negative label, and also learning with label noise (Frénay and Verleysen, 2014; Liu and Tao, 2016).

Multilabel Classification. Label ranking is closely related to multilabel classification, which we consider in Chapter 6, and to tag ranking (Wang et al., 2012). Madjarov et al., (2012) and Zhang and Zhou, (2014) provide a comprehensive review of the literature on multilabel learning. Recent works extend the traditional approaches along multiple directions. In particular, Zhang and Wu, (2015) propose multilabel learning with class specific features, Xu et al., (2016b) explore learning with a large (10^4) number of classes, Luo et al., (2013) and Zhu et al., (2016) consider multi-view multilabel learning, while Aggarwal et al., (2017) and Pham et al., (2017) study multi-instance multilabel learning. Finally, Xu et al., (2016a) explore truncated trace norm regularization for low rank predictors.

There is also substantial amount of work on multilabel classification using deep learning architectures. Gong et al., (2013) use approximate top-k ranking objectives in training convolutional networks for multilabel classification, Zhao et al., (2016) exploit object proposal generation, which is supervised with a localization loss, to produce rich image representations tailored to classifying images containing multiple objects, Wang et al., (2016a) augment a convolutional architecture with a recurrent neural network (RNN) which models labels' dependency and co-occurrence. However, as argued by Vinyals et al., (2015), RNNs might not be well suited to problems where there is no natural order either in the input or in the output. Rezatofighi et al., (2016) address that issue by deriving a loss for learning the parameters of the negative binomial distribution which models the label set cardinality and demonstrate promising results in multilabel classification.

Consistency and Calibration Chapter 6 provides theoretical analysis of surrogate loss functions and OVA schemes investigating their calibration with respect to the top-k error. In this section, we discuss the related research on the analysis of consistency and calibration.

Loss functions are central in machine learning as they provide the means to evaluate the prediction quality and guide the learning algorithm during training. In this thesis, we focus on classification, which is a discrete prediction problem where minimizing the expected 0-1 error is known to be computationally hard

(Ben-David et al., 2003). Instead, it is common to minimize a surrogate loss that leads to efficient learning algorithms. An important question, however, is whether the minimizers of the expected surrogate loss also minimize the expected error. Loss functions which have that property are called calibrated or consistent with respect to the given discrete error measure.

Consistency in binary classification is well understood (Bartlett et al., 2006; Steinwart, 2005, 2007; Zhang, 2004), and significant progress has been made in the analysis of methods for multiclass classification (Pires and Szepesvári, 2016; Ramaswamy and Agarwal, 2012; Tewari and Bartlett, 2007; Zhang, 2004), multilabel classification (Gao and Zhou, 2011; Koyejo et al., 2015), and learning to rank (Buffoni et al., 2011; Calauzenes et al., 2012; Cossock and Zhang, 2008; Duchi et al., 2010; Ramaswamy et al., 2013). In Chapter 6, we investigate calibration of a number of surrogate losses with respect to the top-k error, which generalizes previously established results for consistency of multiclass methods.

Reid and Williamson, (2010b), Vernet et al., (2011), and Williamson et al., (2016) relate classification calibration to the notion of proper losses used in probability estimation. In particular, classification calibration is shown to be a weaker analog of properness suitable for classification problems. Williamson, (2014) discuss the geometry of losses and characterize proper losses in terms of convexity of a so-called superprediction set S. Interestingly, they also propose a procedure to construct novel losses starting with the definition of a new set S and then deriving the loss. While their approach is technically different, it is close in spirit to the technique we discuss in Chapter 6, where novel losses are constructed starting from a modified effective domain of the conjugate loss.

Consistency of algorithms optimizing complex performance measures, such as the F-measure, has been studied by Agarwal, (2013), Dembczynski et al., (2011), Narasimhan et al., (2015, 2014), and Ye et al., (2012). Theoretical analysis beyond consistency involves derivation of generalization error bounds that measure model performance on unseen data. Recent advances for multiclass, multilabel, and structured prediction are reported in (Cortes et al., 2016; Kuznetsov et al., 2014; Lei et al., 2015; Van Erven et al., 2015; Xu et al., 2016a).

Part I

Learning with Limited Training Data

When the number of training examples is small, learning becomes particularly challenging. In this part, we explore two techniques that can be used to improve the predictive performance of learning algorithms in the small sample regime.

- In Chapter 3, we consider the framework of *learning using privileged information*, which was introduced by Vapnik and Vashist, (2009), and explore its relation to learning with instance weights. Furthermore, we investigate the effect of correcting the loss on training data by adjusting the instance weights and observe substantial performance improvements when the weights are learned on a large validation sample.
- In Chapter 4, we look at multitask learning and see if learning a representation that is shared across a large number of classes improves classification of scene images. Moreover, we investigate if multitask learning is scalable to high dimensional feature spaces and propose an efficient optimization scheme based on stochastic dual coordinate ascent (SDCA).

3 Learning Using Privileged Information: SVM+ and Weighted SVM

When the amount of training data is limited, learning algorithms have to rely on prior knowledge more than would be required otherwise. An additional information that is available at training time could be used to improve the predictive performance. The idea of utilizing such **privileged information** during training was explored by Vapnik and Vashist, (2009). They introduced a novel *learning using privileged information* (LUPI) framework and proposed an SVM+ learning algorithm that exploits the privileged features during training.

In this chapter, we relate the privileged information to importance weighting and show that the prior knowledge expressible with privileged features can also be encoded by weights associated with every training example. We show that weighted SVM can always replicate an SVM+ solution, while the converse is not true and we construct a counterexample highlighting the limitations of SVM+. Finally, we touch on the problem of choosing weights for weighted SVMs when privileged features are not available.

The material in this chapter is based on the following publication:

- M. Lapin, M. Hein, and B. Schiele (2014a). "Learning Using Privileged Information: SVM+ and Weighted SVM." in: *Neural Networks* 53.

3.1 Introduction

Classification is a well-studied problem in machine learning, however, learning still remains a challenging task when the amount of training data is limited. Hence, information available in addition to the training sample – *the prior knowledge* – is the crucial factor in achieving further performance improvement.

Prior knowledge comes in different forms and its incorporation into the learning problem depends on a particular setting as well as the algorithm. This chapter focuses on introducing prior knowledge into a support vector machine (SVM) for binary classification. Lauer and Bloch, (2008) provide a review of different ways to incorporate prior knowledge into SVMs and give a categorization of the reviewed methods based on the type of prior knowledge they assume; see also (Schölkopf and Smola, 2002). We will mainly consider the scenario where the additional information is about the *training data* rather than about the target function. A loosely related setting is the semi-supervised learning framework (Chapelle et al., 2006), where unlabeled data carries certain information about the marginal distribution in the input space.

Vapnik and Vashist, (2009) introduced the learning using privileged information (LUPI) paradigm which aims at improving predictive performance of learning algorithms and reducing the amount of required training data. The additional information in this framework comes in the form of privileged features, which are available at training time, but not at test time. These features are used to parametrize the upper bound on the loss function and, essentially, are used to estimate the loss of an optimal classifier on the given training example. Higher loss may be seen as an indication that a given point is likely to be an outlier, and, hence, should be treated differently than a non-outlier. This simple idea has been extensively explored in the literature as we discussed in Chapter 2. The additional information about which training examples are likely to be the outliers can be encoded via instance weights. Therefore, one can already anticipate a close relation between the LUPI framework and importance weighting which is discussed next.

In the weighted learning scenario, each training example comes with a non-negative weight which is used in the loss function to balance the cost of errors. A typical example where instance weights appear naturally is cost-sensitive learning (Elkan, 2001). If the representation of classes in the training sample is unbalanced or different misclassification errors incur different penalties, one can encode that knowledge in the form of instance weights. Assigning high weight to a data point suggests that the learning algorithm should classify that point correctly, possibly at the cost of misclassifying “less important” points. In this chapter, however, we do *not* make the cost-sensitive learning assumption, i.e., we do not assume that different errors incur different costs on the test set. Instead, we decouple importance weighting on the training and on the test sets, and we only focus on the training data. This allows us, in particular, to also assign a high weight to an outlier if that ultimately leads to a better model.

As mentioned above, there are different forms of prior knowledge that can be encoded differently. In this chapter, we show that instance weights can express *the same type of prior knowledge* that is encoded via privileged features. In particular, this allows one to interpret the effect of privileged features in terms of the incurred importance weights. Remarkably, the resulting weights do emphasize outliers, which also happen to be the support vectors in SVMs.

Our focus in this work is on the study of the SVM+ algorithm, which is an extension of the support vector machine to the LUPI framework (Vapnik and Vashist, 2009). Using basic tools of convex analysis, we investigate uniqueness of the SVM+ solution and its relation to solutions of the weighted SVM (WSVM). It turns out there is a simple connection between an SVM+ solution and WSVM instance weights, moreover, that relation can be used to better understand the SVM+ algorithm and to study its limitations. Having realized that instance weights in WSVMs can serve the same purpose as privileged features in SVM+, we turn to the problem of choosing weights when privileged features are not available.

Contributions

Below is a summary of contributions of this chapter.

- We show that any non-trivial SVM+ solution is unique (in the primal), which is a stronger result than the one available for SVM and WSVM, where the offset b may not be unique and requires an *ad hoc* selection procedure.

- By reformulating the SVM+ dual optimization problem, we reveal its close connection to the WSVM algorithm. In particular, we show that any SVM+ dual solution can be used to construct weights for the WSVM which, in turn, produce exactly the same primal solution up to the non-uniqueness of b. This implies that WSVM with the appropriate weights can effectively mimic the SVM+ algorithm.

- We also study whether it is possible to go in the opposite direction which would imply that the two algorithms are equivalent. We give the necessary and sufficient condition for such an equivalence to hold and reveal that the SVM+ solutions are a strict subset of the WSVM solutions. We construct a simple counterexample where a WSVM solution cannot be found by SVM+ regardless of the privileged features and the values of the hyperparameters. This implies that the WSVM algorithm is strictly more general than SVM+ and that not every WSVM solution can be constructed by SVM+.

- Finally, we turn to the problem of choosing weights in the absence of privileged features. We show that the weights can be *learned* directly from data by minimizing an estimate of risk similar to standard procedures of hyperparameter tuning and model selection via cross-validation.

  If a large validation set is available, we show that WSVM with the learned weights outperforms both the SVM and the SVM+ algorithms. This highlights the potential of weighted learning and should motivate further work on the choice of weights when the amount of validation data is limited.

The rest of the chapter is organized as follows.

- In § 3.2, we introduce the SVM+ and the weighted SVM (WSVM) algorithms as well as discuss our notation.

- In § 3.3, we study uniqueness of SVM+ and WSVM solutions. Surprisingly, we discover that SVM+ solutions are unique unlike the SVM or WSVM solutions. We also introduce the notion of *equivalent weights* that all lead to the same WSVM solution.

- In § 3.4, we present our main findings: (i) Theorem 3.3 shows that any SVM+ solution is also a WSVM solution with appropriately chosen weights; (ii) Theorem 3.4 gives the necessary and sufficient condition for equivalence between the SVM+ and WSVM problems; (iii) § 3.4.4 presents a counterexample where a WSVM solution cannot be found by SVM+, no matter which privileged features are used; (iv) § 3.4.5 discusses whether it is possible to complement SVM+ with an SVM– algorithm.

- In § 3.5, we consider the problem of choosing the instance weights for WSVM. In particular, a weight learning method is proposed in § 3.5.3, which exploits a validation sample to select the instance weights automatically.

- In § 3.6, we present our experimental results on synthetic data as well as on a number of publicly available data sets from the UCI repository.

3.2 SVM+ and Weighted SVM

In this section, we introduce our notation and describe the SVM+ and WSVM learning algorithms. Our technical contributions mainly rely on basic results from convex analysis which we recall in § A. Specifically, we employ Lagrangian duality and the Karush-Kuhn-Tucker (KKT) conditions to study the SVM+ algorithm. The KKT conditions for SVM+ and WSVM problems can be found in § A.1.3.

3.2.1 Binary Classification

We consider binary classification with a feature space $\mathcal{X}$ and the label set $\mathcal{Y} = \{-1, 1\}$. Let $S = \{(x_i, y_i)\}_{i=1}^n$ be a training sample drawn i.i.d. from an unknown distribution $\mathbb{P}$ on $\mathcal{X} \times \mathcal{Y}$, and L be a convex margin-based loss function $L : \mathbb{R} \to \mathbb{R}_+$, e.g. the hinge loss $L(yf(x)) = \max\{0, 1 - yf(x)\}$. The task is to learn a prediction function $f : \mathcal{X} \to \mathbb{R}$ that classifies a given example x via $\mathbf{sign}\, f(x)$ and minimizes the expected loss, also called risk, $R(f) \triangleq \mathbb{E}\, L(Yf(X))$.

We use $\tilde{\mathcal{X}}$ to denote the space of *privileged information* and let $\{\tilde{x}_i\}_{i=1}^n$ be the privileged features supplied along with the training sample S. The privileged features are used in SVM+ to "correct" the loss on the given training examples, therefore, the space $\tilde{\mathcal{X}}$ is also called the *correcting space*. The $\star$ symbol is reserved to indicate optimal points associated with an optimization problem.

The derivations are generally easier with Euclidean spaces. However, we would like to highlight that our results generalize beyond linear classification. To that end, we recall that in the nonlinear setting the input data is mapped into a feature space endowed with an inner product. In our example, the decision space $\mathcal{X}$ is mapped into a feature space $\mathcal{Z}$ via a *feature map* Φ:

$$\mathcal{X} \ni x \mapsto z \triangleq \Phi(x) \in \mathcal{Z},$$

and the correcting space $\tilde{\mathcal{X}}$ is mapped into $\tilde{\mathcal{Z}}$ via $\tilde{\Phi}$:

$$\tilde{\mathcal{X}} \ni \tilde{x} \mapsto \tilde{z} \triangleq \tilde{\Phi}(\tilde{x}) \in \tilde{\mathcal{Z}}.$$

It is known (Schölkopf et al., 2001) that every inner product corresponds to a *positive definite kernel* function[1] k as follows:

$$\langle z_i, z_j \rangle_{\mathcal{Z}} = \langle \Phi(x_i), \Phi(x_j) \rangle_{\mathcal{Z}} = k(x_i, x_j).$$

[1] A function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ which for all $n \in \mathbb{N}$, $x_1, \ldots, x_n \in \mathcal{X}$ gives rise to a positive definite kernel matrix K is called a **positive definite kernel**.

Note that the same *kernel trick* applies to the privileged space $\tilde{\mathcal{X}}$ as well, which allows us to formulate algorithms with general kernels in mind. Since the corresponding space should be clear from the context, we omit the subscripts when dealing with inner products and the induced norms.

We let $y = (y_1, \dots, y_n)^\top$ and $Y = \mathbf{diag}(y)$. The kernel matrices K and $\tilde{K}$ are defined entrywise via $K_{ij} = k(x_i, x_j)$ and $\tilde{K}_{ij} = \tilde{k}(\tilde{x}_i, \tilde{x}_j)$ for all $i, j = 1, \dots, n$. The null space and the column space of a matrix A are denoted $\mathcal{N}(A)$ and $\mathcal{R}(A)$ correspondingly. The orthogonal complement of a is $a^\perp$, and $\mathbf{0}$ (respectively $\mathbf{1}$) is the vector of all zeros (ones).

3.2.2 Support Vector Machine with Privileged Information (SVM+)

In the framework of learning using privileged information (LUPI), the decision space $\mathcal{X}$ is augmented with a correcting space $\tilde{\mathcal{X}}$ of privileged features $\tilde{x}$ that are available *at training time only* and are essentially used to estimate the loss $L(y_i f^\star(x_i))$ of an optimal classifier $f^\star \triangleq \arg\min_{f \in \mathcal{H}} L(f)$ on the given training sample. The SVM+ algorithm (Pechyony and Vapnik, 2011) is a generalization of the support vector machine that implements the LUPI paradigm. The slack variables ξ_i are parametrized as a function of privileged features:

$$\xi_i(\tilde{w}, \tilde{b}) \triangleq \langle \tilde{w}, \tilde{z}_i \rangle + \tilde{b},$$

where $(\tilde{w}, \tilde{b})$ are the additional parameters to be learned. The following optimization problem defines the SVM+ algorithm.

$$\begin{aligned} \min_{w,b,\tilde{w},\tilde{b}} \quad & \frac{1}{2}(\|w\|^2 + \gamma \|\tilde{w}\|^2) + C \sum_{i=1}^{n} \xi_i(\tilde{w}, \tilde{b}) \\ \text{s.t.} \quad & y_i(\langle w, z_i \rangle + b) \geq 1 - \xi_i(\tilde{w}, \tilde{b}) \\ & \xi_i(\tilde{w}, \tilde{b}) \geq 0 \end{aligned} \tag{3.1}$$

Note that there are two hyper-parameters, γ and C, that control the trade-off between the three terms of the objective, where the second term limits the capacity of the set of correcting functions $\xi_i(\tilde{w}, \tilde{b})$.

3.2.3 Support Vector Machine with Instance Weights (WSVM)

The weighted support vector machine (WSVM) is a well-known generalization of the standard SVM. Each instance (x_i, y_i) is assigned an *importance weight* $c_i \in \mathbb{R}_+$ and in place of the standard empirical risk estimator $\hat{R}(f) \triangleq n^{-1} \sum_{i=1}^{n} L(y_i f(x_i))$ its weighted version is employed:

$$\hat{R}_{\mathrm{w}}(f) \triangleq \sum_{i=1}^{n} c_i L(y_i f(x_i)).$$

The WSVM optimization problem is given below.

$$\begin{aligned} \min_{w,b,\xi} \quad & \frac{1}{2}\|w\|^2 + \sum_{i=1}^{n} c_i \xi_i \\ \text{s.t.} \quad & y_i(\langle w, z_i \rangle + b) \geq 1 - \xi_i, \\ & \xi_i \geq 0. \end{aligned} \tag{3.2}$$

At first glance, it may appear that the two generalizations of SVM are unrelated. As will become clear in the following, however, there is a relation between the two and the solution space of WSVMs includes SVM+ solutions. This is not very surprising as soon as one realizes that re-weighting allows to alter the loss function to a large extent and, in particular, one can mimic the effect of privileged features. The close relationship can already be seen when comparing the dual problems.

3.2.4 SVM+ and WSVM Lagrange Dual Problems

In this section, we derive the the Lagrange dual problems for the SVM+ and WSVM formulations (3.1) and (3.2). Technical details can be found in Appendix A.1 (page 179) and in (Schölkopf and Smola, 2002; Vapnik et al., 2009).

Let α and β be the Lagrange dual variables of the SVM+ or the WSVM problem corresponding respectively to the first and the second inequality constraints. Define $\tilde{\alpha} \triangleq \alpha + \beta - C\mathbf{1}$, and note that β can be eliminated using $\beta \geq 0$, which leads to the constraint $\alpha_i \leq C + \tilde{\alpha}_i$ for all $i = 1, \ldots, n$. Let

$$F(\alpha) \triangleq (1/2)\, \alpha^\top Y K Y \alpha - \mathbf{1}^\top \alpha, \qquad \tilde{F}(\tilde{\alpha}) \triangleq (1/2)\, \tilde{\alpha}^\top \tilde{K} \tilde{\alpha}.$$

It is not hard to see that the following optimization problem is equivalent to the dual of the SVM+ problem (3.1).

$$\begin{aligned} \min_{\alpha,\tilde{\alpha}} \quad & F(\alpha) + \frac{1}{\gamma}\tilde{F}(\tilde{\alpha}) \\ \text{s.t.} \quad & y^\top \alpha = 0,\ \mathbf{1}^\top \tilde{\alpha} = 0,\ 0 \leq \alpha_i \leq C + \tilde{\alpha}_i. \end{aligned} \tag{3.3}$$

Likewise, the problem below is equivalent to the dual of the WSVM problem (3.2).

$$\begin{aligned} \min_{\alpha} \quad & F(\alpha) \\ \text{s.t.} \quad & y^\top \alpha = 0,\ 0 \leq \alpha_i \leq c_i. \end{aligned} \tag{3.4}$$

Note that the constraint $\alpha_i \leq C + \tilde{\alpha}_i$ is the crucial part of the SVM+ problem as it introduces a coupling between the decision space $\mathcal{X}$ and the correcting space $\tilde{\mathcal{X}}$. Recall from the representer theorem (Schölkopf et al., 2001) that an SVM solution has the form $f = \sum_{i=1}^{n} \alpha_i y_i k(x_i, \cdot)$. Correcting features thus control the maximum influence a data point (x_i, y_i) can have on the resulting classifier, just like the weights in WSVMs.

3.3 Uniqueness Results

The connection between SVM+ and WSVM explored in Section 3.4 relies on the analysis of uniqueness of their solutions. Effectively, the statements can only be made with respect to the classes of equivalent solutions and equivalent weights, hence, it is imperative to first obtain a better understanding of different sources of non-uniqueness in these optimization problems.

In this section, we show that every non-trivial SVM+ solution is unique, unlike WSVM solutions that may have a non-unique offset b. Furthermore, we describe a set of equivalent weights that yield the same WSVM solutions. The latter will be used to prove equivalence between the SVM+ and the WSVM algorithms under additional constraints.

3.3.1 Uniqueness of WSVM and SVM+ Solutions

We begin with a known result due to Burges and Crisp, (1999) that characterizes uniqueness of the weighted SVM solution. Essentially, it states that if there is an equilibrium between instance weights of support vectors, then the separating hyperplane can be shifted within a certain range without altering the total cost in the WSVM problem. In that case, a WSVM solver has to rely on some additional information or an *ad hoc* heuristic to choose a value for b in the allowed range.

Theorem 3.1 (Burges and Crisp, 1999)**.** Define the following index sets:

$$\mathcal{I}_{\pm} \triangleq \{i : y_i \gtrless 0\}, \qquad \mathcal{I}_0 \triangleq \{i : y_i f(x_i) < 1\}, \qquad \mathcal{I}_1 \triangleq \{i : y_i f(x_i) \leq 1\}.$$

The solution to the problem (3.2) is unique in w. It is not unique in b and ξ iff one of the following two conditions holds:

$$\sum_{i \in \mathcal{I}_- \cap \mathcal{I}_0} c_i = \sum_{i \in \mathcal{I}_+ \cap \mathcal{I}_1} c_i, \qquad \sum_{i \in \mathcal{I}_+ \cap \mathcal{I}_0} c_i = \sum_{i \in \mathcal{I}_- \cap \mathcal{I}_1} c_i.$$

Note that in practice it may happen that one of the two conditions holds and the WSVM problem (3.2) does not have a unique solution. This is not the case for the SVM+ as shown next.

Theorem 3.2. For any $C > 0$, $\gamma > 0$, the solution to the problem (3.1) is unique in $(w, \tilde{w}, \tilde{b})$. If there is a support vector, then b is unique as well, otherwise:

$$\max_{i \in \mathcal{I}_+} \left\{1 - \langle \tilde{w}, \tilde{z}_i \rangle - \tilde{b}\right\} \leq b \leq \min_{i \in \mathcal{I}_-} \left\{\langle \tilde{w}, \tilde{z}_i \rangle + \tilde{b} - 1\right\}.$$

Proof. Following Burges and Crisp, (1999), let F be the objective function:

$$F = \frac{1}{2}\left\|w\right\|^2 + \frac{\gamma}{2}\left\|\tilde{w}\right\|^2 + C\sum_{i=1}^{n}(\langle \tilde{w}, \tilde{z}_i \rangle + \tilde{b}),$$

and define $u \triangleq (w, \tilde{w}, \tilde{b})^\top$. Suppose u_1 and u_2 are two solutions, then, since the problem is convex, a family of solutions is given by $u_t = (1-t)u_1 + tu_2$, $t \in [0, 1]$, and $F(u_1) = F(u_2) = F(u_t)$. Expanding $F(u_t) - F(u_1) = 0$ and differentiating with respect to t yields:

$$\begin{gathered}(t-1)\|w_1\|^2 + (1-2t)\langle w_1, w_2\rangle + t\|w_2\|^2 \\ +\gamma\left[(t-1)\|\tilde{w}_1\|^2 + (1-2t)\langle \tilde{w}_1, \tilde{w}_2\rangle + t\|\tilde{w}_2\|^2\right] \\ +tC\sum_{i=1}^{n}\left(\langle \tilde{w}_2 - \tilde{w}_1, \tilde{z}_i\rangle + \tilde{b}_2 - \tilde{b}_1\right) = 0, \\ \|w_1 - w_2\|^2 + \gamma\|\tilde{w}_1 - \tilde{w}_2\|^2 = 0.\end{gathered}$$

Since $\gamma > 0$ it follows that $w_1 = w_2$ and $\tilde{w}_1 = \tilde{w}_2$. Plugging that into the first equation yields $\tilde{b}_2 = \tilde{b}_1$. Uniqueness of b now follows from the complementary slackness condition (see page 180). If all $\alpha_i = 0$, i.e. there are no support vectors, then $w = \mathbf{0}$ and the result follows from the KKT conditions (A.8) (page 182). □

This result is interesting on its own, since it shows that the SVM+ is formulated in a way that privileged features always give enough information to choose *the* unique solution (if there are no support vectors, then the constant classifier can be given by $b = \pm 1$ depending on the class balance).

Uniqueness of the Dual Solution

Next, we consider uniqueness of dual solutions, which becomes relevant when the SVM+ or WSVM algorithms are implemented in the dual. These results are included for completeness and do not play a major role in the rest of this chapter.

Proposition 3.1. If $(\alpha_1, \tilde{\alpha}_1)$ and $(\alpha_2, \tilde{\alpha}_2)$ are two solutions to the SVM+ dual optimization problem (3.3), then

$$\begin{aligned}(\alpha_1 - \alpha_2) &\in \mathcal{N}(YKY) \cap \mathbf{1}^{\perp} \cap y^{\perp}, \\ (\tilde{\alpha}_1 - \tilde{\alpha}_2) &\in \mathcal{N}(\tilde{K}) \cap \mathbf{1}^{\perp}.\end{aligned}$$

If α_1 and α_2 are two solutions to the WSVM dual problem (3.4), then

$$(\alpha_1 - \alpha_2) \in \mathcal{N}(YKY) \cap \mathbf{1}^{\perp} \cap y^{\perp}.$$

Proof. The proof employs the same method as in the proof of Theorem 3.2 and we only provide the part concerning the WSVM problem.

Let $K' = YKY$ and consider a family of solutions $\alpha_t = (1-t)\alpha_1 + t\alpha_2$, $t \in [0,1]$. Note that $(\alpha_1 - \alpha_2) \in y^{\perp}$ follows directly from the optimization constraints. Expanding $F(\alpha_t) - F(\alpha_1) = 0$ and differentiating with respect to t yields:

$$\begin{gathered}(t-1)\alpha_1^{\top}K'\alpha_1 + (1-2t)\alpha_1^{\top}K'\alpha_2 + t\alpha_2^{\top}K'\alpha_2 + \mathbf{1}^{\top}(\alpha_1 - \alpha_2) = 0, \\ (\alpha_1 - \alpha_2)^{\top}K'(\alpha_1 - \alpha_2) = 0.\end{gathered}$$

It follows that $(\alpha_1 - \alpha_2) \in \mathcal{N}(K')$. Let $\alpha_1 = \alpha_2 + v$, $v \in \mathcal{N}(K')$, then from the first equation $\mathbf{1}^{\top}v = 0$, which completes the proof. □

Corollary 3.1. If K has full rank, then solution to the problem (3.4) is unique. If K and $\tilde{K}$ have full rank, then solution to the problem (3.3) is unique.

Perhaps it is not surprising that when the Gram matrix K (respectively $\tilde{K}$) is full rank, then the dual solution is unique both for SVM+ and WSVM.

3.3.2 Equivalent Weights

Apart from the conditions discussed in the previous section, another source of non-uniqueness is that any given WSVM solution corresponds, in general, to multiple weight vectors c. In this section, we give a characterization of all such vectors.

Definition 3.1. A family of equivalent weights $\mathcal{W}$ is defined for a given WSVM solution $(w^\star, b^\star, \xi^\star, \alpha^\star, \beta^\star)$ as

$$\mathcal{W} \triangleq \{\mu + \nu \,|\, \mu \in \mathcal{U},\ \nu \in \mathcal{V}\},$$

where we define the two subspaces $\mathcal{U}$ and $\mathcal{V}$ as

$$\mathcal{U} \triangleq \{\mu \in \mathbb{R}^n_+ \,|\, \mathbf{1}^\top(\mu - \alpha^\star) = 0,\ \textstyle\sum_i \mu_i y_i z_i = w^\star,\ \mu^\top y = 0,\ \mu_i(\xi_i^\star - h_i) = 0\ \forall i\},$$
$$\mathcal{V} \triangleq \{\nu \in \mathbb{R}^n_+ \,|\, \nu_i \xi_i^\star = 0\ \forall i\},$$

with $h_i \triangleq \max\{0, 1 - y_i(\langle w^\star, z_i\rangle + b^\star)\}$ being the hinge loss at (x_i, y_i) for all i.

The following simple statement shows that the set $\mathcal{W}$ defined above contains *all* weights that correspond to a given WSVM solution.

Proposition 3.2. Let $(w^\star, b^\star, \xi^\star)$ and $(\alpha^\star, \beta^\star)$ be primal and dual optimal points for the WSVM problem (3.2). The point $(w^\star, b^\star, \xi^\star)$ is primal optimal for any weight vector $c \in \mathcal{W}$, and all such weights are contained in $\mathcal{W}$.

Proof. The proof consists in a straightforward application of the KKT conditions (A.5) (page 181). The additional constraint $\mathbf{1}^\top(\mu - \alpha^\star) = 0$ follows from Proposition 3.1 since it must hold that $(\mu - \alpha^\star) \in \mathbf{1}^\perp$. □

By definition, the set $\mathcal{V}$ suggests that there is freedom to assign any nonnegative weight on the points (x_i, y_i) where the optimal slack variable $\xi_i^\star$ is zero, which implies that the optimal loss on such examples is zero. Another way to express this fact is given below.

Corollary 3.2. There exists a vector $c' \in \mathcal{W}$ such that $c' = \alpha' = \alpha^\star$ and $\beta' = \mathbf{0}$.

It is not surprising that *a posteriori* all weight could be concentrated on support vectors as suggested by Corollary 3.2. As will become clear in the following, this is close to what the SVM+ algorithm is constrained to do.

3.4 SVM+ and WSVM Relationship

In this section, we reveal a connection that exists between the SVM+ and WSVM algorithms. We also present our main theoretical results, in particular, the conditions under which SVM+ and WSVM are equivalent. The content of this section is summarized below.

- In § 3.4.1, we show that it is always possible to find weights from an SVM+ solution such that WSVM constructs the same solution.
- In § 3.4.2, we discuss when it is possible to go in the opposite direction and reveal a fundamental constraint of the SVM+ algorithm. Together with the previous result, it implies that the set of SVM+ solutions is a proper subset of all WSVM solutions.
- In § 3.4.3, we state the necessary and sufficient condition for the equivalence between SVM+ and WSVM. The condition is given in terms of the WSVM instance weights and the optimal loss on training data. It is our attempt to provide a description of the subset of all WSVM solutions that coincides with the set of all SVM+ solutions.
- In § 3.4.4, we present an illustrative counterexample violating that condition.
- In § 3.4.5, we give our preliminary ideas about existence of the complement of SVM+ which we call SVM–.

3.4.1 SVM+ Solutions Are Also WSVM Solutions

The following theorem shows that any SVM+ solution is also a solution to the WSVM problem with appropriately chosen weights. Moreover, such a choice of weights can always be given by the SVM+ dual variables.

Theorem 3.3. Let $(w^\star, b^\star, \tilde{w}^\star, \tilde{b}^\star)$ and $(\alpha^\star, \beta^\star)$ be primal and dual optimal points for the SVM+ problem (3.1). There exists a choice of $\xi^\star$ and instance weights c, namely $c = \alpha^\star + \beta^\star$, such that $(w^\star, b^\star, \xi^\star)$ and $(\alpha^\star, \beta^\star)$ are primal and dual optimal points for the WSVM problem (3.2) with weights c.

Proof. Given any fixed feasible $\tilde{\alpha}$, the SVM+ problem (3.3) is equivalent to the WSVM problem (3.4) with $c = \tilde{\alpha} + C\mathbf{1}$. In particular, if $(\alpha^\star, \tilde{\alpha}^\star)$ is a solution to (3.3), then $\alpha^\star$ is a solution to (3.4) with $c = \tilde{\alpha}^\star + C\mathbf{1} = \alpha^\star + \beta^\star$. Let $\xi_i^\star = \langle \tilde{w}^\star, \tilde{z}_i \rangle + \tilde{b}^\star$, then the points $(w^\star, b^\star, \xi^\star)$ and $(\alpha^\star, \beta^\star)$ verify the KKT conditions (A.5) (page 181) for the WSVM problem (3.2). □

Pechyony and Vapnik, (2010) argue that a good choice of privileged features leads to improved predictive performance of the SVM+ algorithm. Therefore, a direct corollary of Theorem 3.3 above is that a good choice of weights leads to improved performance of WSVM. We verify that claim empirically in a set of experiments (§ 3.6, page 55) where the weights are learned using a large validation sample. Although idealized, that experiment is close in spirit to the Oracle SVM setting of Vapnik and Vashist, (2009).

Figure 3.1 shows a toy binary classification example in $\mathbb{R}$. The optimal decision boundary is at $x_0 = 3.5$, and we let the points $x_2 = 2$ and $x_5 = 5$ be the support vectors on the margin. From that, we can compute the optimal slack variables for

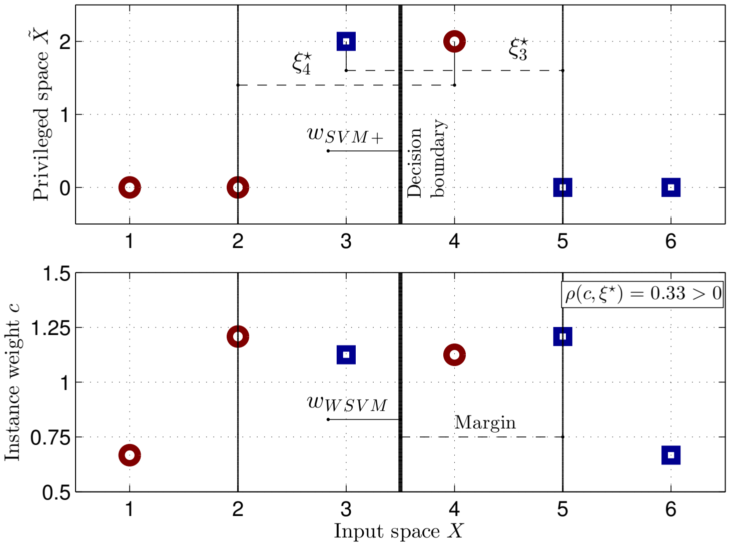


Figure 3.1.: An example of equivalence between SVM+ (top) and WSVM (bottom). The privileged features coincide with the optimal slack variables $\xi_i^\star$, as motivated by the LUPI paradigm, and instance weights c_i are given by the sum of SVM+ dual variables (Theorem 3.3). Note that since the WSVM solution replicates an SVM+ solution, the weighted average loss is greater than the non-weighted one, i.e. $\rho(c, \xi^\star) \geq 0$ (Theorem 3.4).

the misclassified points x_3 and x_4 as follows: $\xi_3^\star = \xi_4^\star = 2$, and the remaining $\xi_i^\star$ are at zero. Using the optimal slack variables as privileged features, $\tilde{x}_i = \xi_i^\star$, we obtain the SVM+ solution at the top of Figure 3.1.

Next, we use the SVM+ solution to compute instance weights as $c = \alpha^\star + \beta^\star$ and run the WSVM algorithm which finds exactly the same solution shown at the bottom. Note that the outliers (points x_3 and x_4) receive relatively high weight, so that the weighted average loss is greater than the non-weighted one. We investigate that phenomenon in § 3.4.3 where further details are provided.

3.4.2 Which WSVM Solutions Are SVM+ Solutions?

We now consider the opposite direction and characterize the SVM+ solutions in terms of the induced instance weights. The following Lemma 3.1 highlights the bias of the SVM+ algorithm as it establishes that every solution must satisfy a certain relation between the dual variables (respectively the weights) and the loss on the training sample. This is the key to showing that the SVM+ and the WSVM algorithms are not equivalent, and that the latter is strictly more generic as it does not impose that additional constraint.

Lemma 3.1. Let $C > 0$, $\gamma \geq 0$ be the SVM+ regularization parameters, and let $(w^\star, b^\star, \tilde{w}^\star, \tilde{b}^\star)$ and $(\alpha^\star, \beta^\star)$ be the corresponding primal and dual optimal points for the SVM+ problem (3.1). The following inequality holds:

$$\frac{\sum_{i=1}^{n}(\alpha_i^\star + \beta_i^\star)h_i}{\sum_{i=1}^{n}(\alpha_i^\star + \beta_i^\star)} \geq \frac{1}{n}\sum_{i=1}^{n} h_i, \tag{3.5}$$

where $h_i \triangleq \max\{0, 1 - y_i(\langle w^\star, z_i\rangle + b^\star)\}$ is the hinge loss at a point (x_i, y_i) for all $i = 1, \ldots, n$. If $\gamma = 0$, then (3.5) is satisfied with equality.

Proof. It follows from the KKT conditions (A.8) (page 182) that

$$\langle \tilde{w}^\star, \tilde{z}_i\rangle + \tilde{b}^\star = h_i + \delta_i,$$

with $\delta_i \geq 0$ for all $i = 1, \ldots, n$, and

$$(\alpha_i^\star + \beta_i^\star)\delta_i = 0,$$
$$\alpha_i^\star > 0 \vee \beta_i^\star > 0 \Rightarrow \delta_i = 0.$$

Multiplying by $(\alpha_i^\star + \beta_i^\star - C)$ and summing up yields

$$\gamma \langle \tilde{w}^\star, \tilde{w}^\star\rangle = \sum_{i=1}^{n}(\alpha_i^\star + \beta_i^\star)h_i - C\sum_{i=1}^{n}(h_i + \delta_i).$$

Note that $C = \frac{1}{n}\sum_{i=1}^{n}(\alpha_i^\star + \beta_i^\star) > 0$, hence

$$\gamma \langle \tilde{w}^\star, \tilde{w}^\star\rangle = \sum_{i=1}^{n}(\alpha_i^\star + \beta_i^\star)h_i - \frac{1}{n}\sum_{i=1}^{n}(\alpha_i^\star + \beta_i^\star)\sum_{i=1}^{n}(h_i + \delta_i).$$

Since $\gamma \langle \tilde{w}^\star, \tilde{w}^\star\rangle \geq 0$, it must hold that

$$\begin{aligned}\sum_{i=1}^{n}(\alpha_i^\star + \beta_i^\star)h_i &\geq \frac{1}{n}\sum_{i=1}^{n}(\alpha_i^\star + \beta_i^\star)\sum_{i=1}^{n}(h_i + \delta_i)\\ &\geq \frac{1}{n}\sum_{i=1}^{n}(\alpha_i^\star + \beta_i^\star)\sum_{i=1}^{n} h_i. \end{aligned}\tag{3.6}$$

Division by $\sum_{i=1}^{n}(\alpha_i^\star + \beta_i^\star)$ completes the proof. □

Taking into account that the corresponding weights in WSVM are given by the sum of the SVM+ dual variables, the above inequality can be re-written in a more compact form.

Corollary 3.3 (The Necessary Condition)**.** Let $(w^\star, b^\star, \tilde{w}^\star, \tilde{b}^\star)$ and $(\alpha^\star, \beta^\star)$ be primal and dual optimal points for the SVM+ problem (3.1). Define instance weights $c = \alpha^\star + \beta^\star$, and let $(w^\star, b^\star, \xi^\star)$ and $(\alpha^\star, \beta^\star)$ be primal and dual optimal points for the WSVM problem (3.2) with weights c. Then the following holds:

$$\langle c - \bar{c}\mathbf{1}, \xi^\star\rangle \geq 0,$$

where $\xi_i^\star = \max\{0, 1 - y_i(\langle w^\star, z_i \rangle + b^\star)\}$ and $\bar{c} \triangleq (1/n) \sum_{i=1}^n c_i$.

Proof. Follows from Theorem 3.3 and Lemma 3.1. □

Note that this result suggests a simple way to interpret the effect of privileged features – they impose a re-weighting of the input training data. Moreover, at the end of training more emphasis will be on points with positive loss and less on easy points, in particular, the non-support vectors may end up with zero weight.

SVM+ Reduction to Standard SVM

In this section, we slightly deviate from the main subject to show that when there is an equality in the previous lemma, then SVM+ reduces to the standard SVM. For simplicity, we only state this result for $\tilde{x}_i \in \mathbb{R}^d$.

Proposition 3.3. Assume the setting of Lemma 3.1 and let (3.5) be satisfied with equality, then

$$\langle \tilde{w}^\star, \tilde{x}_i \rangle + \tilde{b}^\star = h_i, \quad i = 1, \dots, n.$$

Furthermore, the following holds:

1. If $\gamma > 0$, then $\tilde{w}^\star = \mathbf{0}$ and $\tilde{b}^\star = h_i$ for all $i = 1, \dots, n$, i.e. the loss on all data points is the same. Hard margin SVM is a special case with $\tilde{b}^\star = 0$.

2. If $\gamma = 0$, then $\tilde{X}\tilde{\alpha}^\star = \mathbf{0}$, where $\tilde{X}$ is the matrix obtained by stacking all $\tilde{x}_i$. If additionally $\mathbf{rank}(\tilde{X}) = n$, then $\alpha_i^\star + \beta_i^\star = C$ for all $i = 1, \dots, n$ and any vector in $\mathbb{R}^n$ can be represented via $\langle \tilde{w}^\star, \tilde{x}_i \rangle + \tilde{b}^\star$, hence the soft margin SVM is recovered with $\xi_i^\star = \langle \tilde{w}^\star, \tilde{x}_i \rangle + \tilde{b}^\star$.

Proof. It follows from (3.6) that $\delta_i = 0$ and $\langle \tilde{w}^\star, \tilde{x}_i \rangle + \tilde{b}^\star = h_i$ for $i = 1, \dots, n$. If $\gamma > 0$, then $\gamma \langle \tilde{w}^\star, \tilde{w}^\star \rangle = 0$ implies $\tilde{w}^\star = \mathbf{0}$ and thus $\tilde{b}^\star = h_i$ for all i.

If $\gamma = 0$, then the KKT conditions (A.8) (page 182) imply $\tilde{X}\tilde{\alpha}^\star = \mathbf{0}$, where $\tilde{\alpha}^\star = \alpha^\star + \beta^\star - C\mathbf{1}$, as before. If $\mathbf{rank}(\tilde{X}) = n$, then $\tilde{X}\tilde{\alpha}^\star = \mathbf{0}$ yields $\tilde{\alpha}^\star = \mathbf{0}$, and so $\alpha_i^\star + \beta_i^\star = C$ for $i = 1, \dots, n$. Since $(\tilde{x}_i)_{i=1}^n$ forms a basis in $\mathbb{R}^n$ and there is no penalty on $\|\tilde{w}\|^2$ in the objective function, then SVM+ does not impose any additional constraints compared to the soft margin SVM. The primal and dual optimal points for SVM+ are thus also optimal for SVM with $\xi_i^\star = \langle \tilde{w}^\star, \tilde{x}_i \rangle + \tilde{b}^\star$. □

3.4.3 SVM+ and WSVM Equivalence

We now state the main result of this chapter which gives the necessary and sufficient condition for the equivalence between SVM+ and WSVM.

Theorem 3.4. Let $(w^\star, b^\star, \xi^\star)$ and $(\alpha_0^\star, \beta_0^\star)$ be primal and dual optimal points for the WSVM problem (3.2) with instance weights $c_0 \in \mathbb{R}_+^n$, not all zero. There exists a choice of parameters C, γ, and correcting features $\{\tilde{x}_i\}_{i=1}^n$ such that $(w^\star, b^\star)$ is optimal for the SVM+ problem iff:

$$\exists\, c \in \mathcal{W} \;:\; \rho(c, \xi^\star) \triangleq \langle c - \bar{c}\mathbf{1}, \xi^\star \rangle \geq 0, \tag{3.7}$$

where $\bar{c} \triangleq (1/n)\sum_{i=1}^n c_i$. If $\rho(c,\xi^\star) \geq 0$, one such possible choice is as follows:

$$C = \bar{c}, \qquad \gamma = \rho(c,\xi^\star), \qquad \tilde{x}_i = \xi_i^\star - \tilde{b}^\star,\ \forall i \tag{3.8}$$

moreover, the optimal $\tilde{w}^\star$ and $\tilde{b}^\star$ in that case are:

$$\tilde{w}^\star = 1, \qquad \tilde{b}^\star = \langle c,\xi^\star\rangle / \langle c,\mathbf{1}\rangle . \tag{3.9}$$

Proof. (3.7) **is necessary.** Assume primal and dual optimal points for the SVM+ problem (3.1) are $(w^\star, b^\star, \tilde{w}^\star, \tilde{b}^\star)$ and $(\alpha^\star, \beta^\star)$, and let $c = \alpha^\star + \beta^\star$ (note that $(\alpha^\star, \beta^\star)$ and $(\alpha_0^\star, \beta_0^\star)$ may be different). Theorem 3.3 states that there exists a $\xi_0^\star$ such that $(w^\star, b^\star, \xi_0^\star)$ and $(\alpha^\star, \beta^\star)$ are primal and dual optimal for the WSVM problem with the weights c. We need to show that $\xi_0^\star = \xi^\star$. This follows directly from the KKT conditions (A.5) (page 181) when all $c_{0,i} > 0$ and $c_i > 0$, since $h_i = \max\{0, 1 - y_i(\langle w^\star, x_i\rangle + b^\star)\}$ are the same for both problems. If some of the weights are zero, then the corresponding $\xi_i^\star$ is not uniquely defined (it is unbounded from above) and we assume that the algorithm returns the value at the lower bound, i.e. $\xi_i^\star = h_i$. Now, given that $\xi_0^\star = \xi^\star$, we have that $c \in \mathcal{W}$ by Proposition 3.2 and $\rho(c,\xi^\star) \geq 0$ by Corollary 3.3.

(3.7) **is sufficient.** First, consider the case $\rho(c,\xi^\star) > 0$, and let $(w^\star, b^\star, \xi^\star)$ and $(\alpha^\star, \beta^\star)$ be primal and dual optimal points for the WSVM problem with the weights c. We now construct the privileged features $\{\tilde{x}_i\}_{i=1}^n$ and provide $C > 0$, $\gamma > 0$, $\tilde{w}^\star$, and $\tilde{b}^\star$ such that $(w^\star, b^\star, \tilde{w}^\star, \tilde{b}^\star)$ and $(\alpha^\star, \beta^\star)$ are primal and dual optimal for the corresponding SVM+ problem.

It is sufficient to find one dimensional correcting features that additionally satisfy $\sum_{i=1}^n c_i \tilde{x}_i = 0$. The KKT conditions in this case imply that

$$\tilde{w}^\star = -\frac{C}{\gamma}\sum_{i=1}^n \tilde{x}_i, \qquad C = \frac{1}{n}\sum_{i=1}^n c_i = \bar{c}. \tag{3.10}$$

We require for all $i = 1, \dots, n$ that

$$\tilde{w}^\star \tilde{x}_i + \tilde{b}^\star = \max\{0, 1 - y_i(\langle w^\star, x_i\rangle + b^\star)\} = \xi_i^\star. \tag{3.11}$$

Multiplying both sides by c_i and summing up yields

$$\tilde{b}^\star = \langle c,\xi^\star\rangle / \langle c,\mathbf{1}\rangle .$$

Plugging (3.10) into (3.11) and solving for $\tilde{x}_i$ one gets:

$$\tilde{x}_i = \pm(\tilde{b}^\star - \xi_i^\star)\sqrt{\gamma/\rho(c,\xi^\star)}. \tag{3.12}$$

Choosing $\gamma = \rho(c,\xi^\star)$ and the plus sign in (3.12) for convenience, we have that (3.10) leads to $\tilde{w}^\star = \rho(c,\xi^\star)/\gamma = 1$.

Now, consider $\rho(c, \xi^\star) = 0$. Let $\tilde{X} = [\tilde{x}_1 \cdots \tilde{x}_n]$ be the matrix obtained by stacking $\{\tilde{x}_i\}$, and let $\gamma = 0$. Proposition 3.3 and the KKT conditions imply:

$$C = \bar{c}, \qquad \tilde{X}(c - \bar{c}\mathbf{1}) = \mathbf{0}, \qquad \tilde{X}^\top \tilde{w}^\star + \tilde{b}^\star \mathbf{1} = \xi^\star.$$

Hence, the matrix $\tilde{X}$ must satisfy

$$(c - \bar{c}\mathbf{1}) \in \mathcal{N}(\tilde{X}), \qquad (\xi^\star - \tilde{b}^\star \mathbf{1}) \in \mathcal{R}(\tilde{X}^\top).$$

The above requirements translate to

$$\langle c - \bar{c}, \xi^\star - \tilde{b}^\star \mathbf{1} \rangle = 0,$$

which holds for (3.8), (3.9), and $\rho(c, \xi^\star) = 0$. □

Let us make a few remarks. First, the condition (3.7) can be rewritten in terms of the averages as

$$\sum_{i=1}^{n} \omega_i \xi_i^\star \geq \frac{1}{n} \sum_{i=1}^{n} \xi_i^\star, \tag{3.13}$$

where $\omega_i \triangleq c_i / \sum_{i=1}^n c_i$ is the normalized weight. Hence, any SVM+ solution has an equivalent WSVM setting that puts *more weight on hard examples*, i.e. the points with higher loss.

Further, it is clear from the definition of equivalent weights (Definition 3.1) that the weight of points with $y_i f(x_i) > 1$ can be changed arbitrarily without altering the f since in that case $\xi_i^\star = 0$, $\alpha_i^\star = 0$ and $\beta_i^\star = c_i$, i.e. these points are not support vectors and they have no influence on the final classifier. Hence, their weight – the upper bound on the influence – does not matter.

This reasoning leads us to a condition that is much easier to check in practice than the one in Theorem 3.4. Note that condition (3.7) involves the set of equivalent weights and it is possible to check it directly using the definition of $\mathcal{W}$ as will be discussed below. However, if the kernel matrix is non-singular, as is often the case with the Gaussian kernel, then one can simply take $c = \alpha^\star$ and check (3.7) for that particular weight vector only.

Proposition 3.4. Let $(w^\star, b^\star, \xi^\star)$ and $(\alpha^\star, \beta^\star)$ be primal and dual optimal points for the WSVM problem (3.2) with instance weights $c \in \mathbb{R}^n_+$, not all zero. If

$$\mathcal{N}(YKY) \cap \mathbf{1}^\perp \cap y^\perp = \{\mathbf{0}\},$$

then there exists a choice of C, γ, and privileged features $\{\tilde{x}_i\}_{i=1}^n$, such that $(w^\star, b^\star)$ is primal optimal for the SVM+ problem (3.1) iff:

$$\rho(\alpha^\star, \xi^\star) = \xi^{\star\top} \Big(\mathbf{I} - (1/n)\mathbf{1}\mathbf{1}^\top\Big) \alpha^\star \geq 0. \tag{3.14}$$

Proof. Sufficiency follows directly from Theorem 3.4 since $c = \alpha^\star$ is a valid choice of weights (see Definition 3.1). For necessity, note that $\alpha^\star$ is unique by Proposition 3.1

and all the weights in $\mathcal{W}$ are of the form $c = \alpha^\star + \beta$, for $\beta \in \mathcal{V}$. The maximum in (3.7) corresponds to

$$\max_{\beta \geq 0} \sum_{i=1}^{n} \xi_i^\star \beta_i - (1/n) \sum_{i=1}^{n} \xi_i^\star \sum_{i=1}^{n} \beta_i,$$

which is attained at $\beta = \mathbf{0}$ since $\xi_i^\star \beta_i = 0$ for all $i = 1, \ldots, n$. □

Intuitively, the SVM+ algorithm maximizes the margin $2 \|w\|^{-1}$ by minimizing $F(\alpha)$, as in the standard SVM, and also gradually shifts focus to hard examples by minimizing $\tilde{F}(\tilde{\alpha})$. As long as there are sufficiently many points on the "correct" side of the margin, (3.13) can be achieved by reducing the weight of such non-support vectors, and so the SVM+ solution space is often as rich as that of the WSVM. In general, however, (3.13) may not be attainable without altering the f as demonstrated by the counter example below.

3.4.4 WSVM Solution Not Found by SVM+

We now consider the case when misclassified training points have low weight, i.e. $\rho(c, \xi^\star) < 0$, and construct an instance where SVM+ fails to find the corresponding WSVM solution. Consider the following example (Figure 3.2):

$$S = \{(1, +1),\ (2, -1),\ (3, +1)\}, \qquad c = (4, 6, 2)^\top.$$

The corresponding primal and dual optimal points are

$$\begin{aligned} w^\star &= -2, & \xi^\star &= (0, 0, 4)^\top, & \alpha^\star &= (4, 6, 2)^\top, \\ b^\star &= 3, & & & \beta^\star &= (0, 0, 0)^\top. \end{aligned}$$

Since $\rho(c, \xi^\star) = -\frac{2}{3} < 0$, this solution does not correspond to any of the SVM+ solutions (Lemma 3.1). Note that one can easily verify that $\mathcal{N}(YKY) \cap \mathbf{1}^\perp \cap y^\perp$ contains only $\mathbf{0}$, hence, Proposition 3.4 already completes the claim.

Similarly, one can show using Definition 3.1 that $\mathcal{U} = \{\alpha^\star\}$ and that other equivalent weights can only increase the weight of points 1 and 2, which would only decrease $\rho(c, \xi^\star)$. Therefore, there are no instance weights $c' \in \mathcal{W}$ for which $\rho(c', \xi^\star) \geq 0$ and, by Theorem 3.4, there is no correcting space that would make $(w^\star, b^\star) = (-2, 3)$ an SVM+ solution.

Figure 3.2 shows the learned WSVM and SVM+ models, where we use

$$\tilde{x}_i = \xi_i^\star - \langle c, \xi^\star \rangle \, / \, \langle c, \mathbf{1} \rangle\,, \qquad C = \bar{c}, \qquad \gamma = 1.$$

A different choice of C and γ could make SVM+ return, for example, a constant classifier, which is the solution of the standard SVM on that data, but there is no setting that would make it return $(w^\star, b^\star) = (-2, 3)$.

Note that in this example an even stronger result can be shown: SVM+ cannot reproduce the same *type* of dichotomy, i.e. even if we allowed it to return a line with *any* negative slope going through the same point, the SVM+ would still fail.

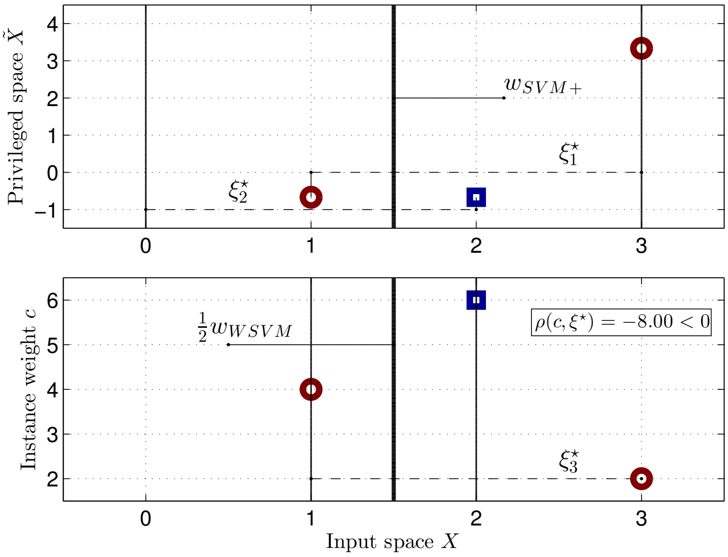


Figure 3.2.: An example of a WSVM solution (bottom) that cannot be found by SVM+. The instance weights c_i are chosen in a way to avoid a zero norm constant classifier ($f = +1$). The resulting weighted average loss is less than the non-weighted one, hence the SVM+ cannot find this solution. Computing the privileged features as in (3.8) leads to an SVM+ solution with the opposite prediction and a higher value of the weighted average loss.

This shows that there are settings where WSVM performs significantly better than SVM+ due to a fundamental constraint of the latter.

3.4.5 Is There an SVM−?

We have seen that SVM+ has a more constrained solution space than WSVM. Lemma 3.1 gives an exact characterization of that constraint in terms of the relation between the SVM+ dual variables and the optimal loss on the training sample. The WSVM solution space can thus be partitioned into solutions that can be found by SVM+ and the rest. We are now interested if there is a modification to the SVM+ algorithm that would yield solutions from that second part.

Theorem 3.4 suggests that $\gamma = \rho(c,\xi^\star) \geq 0$, so, intuitively, if we now require $\rho(c,\xi^\star) < 0$, the corresponding γ has to be with a minus:

$$\begin{aligned} \min_{w,b,\tilde{w},\tilde{b}} \quad & \frac{1}{2}(\|w\|^2 - \gamma\,\|\tilde{w}\|^2) + C\sum_{i=1}^{n}\xi_i(\tilde{w},\tilde{b}) \\ \text{s.t.} \quad & y_i(\langle w, z_i\rangle + b) \geq 1 - \xi_i(\tilde{w},\tilde{b}) \\ & \xi_i(\tilde{w},\tilde{b}) \geq 0 \end{aligned} \tag{3.15}$$

This problem is clearly non-convex as the objective is now a difference of convex functions. If there was a finite (local) minimizer $(w^\star, b^\star, \tilde{w}^\star, \tilde{b}^\star)$, the KKT conditions would still hold (Borwein and Lewis, 2000, Theorem 2.3.8) for a Lagrange multiplier vector $(\alpha^\star, \beta^\star)$, and one could show a result similar to Lemma 3.1, but with the reverse inequality.

Unfortunately, however, the problem (3.15) is unbounded from below, which is easy to see: the quadratic term $\|\tilde{w}\|^2$ grows faster than the linear term $\xi_i(\tilde{w}, \tilde{b})$ and the feasible set is unbounded. This shows that it is not trivial to modify the SVM+ algorithm to obtain solutions from its complement, and it is an open question if such a modification (with non-degenerate solutions) exists at all.

The phenomenon we observe here is that some of the WSVM solutions, namely the ones where $\rho(c, \xi^\star) \geq 0$, can be computed easily within the LUPI framework, while others ($\rho(c, \xi^\star) < 0$) may be completely out of reach. What are the implications of this observation in terms of learning a classifier?

Consider any training sample S of size n for a problem $\mathbb{P}$. Let $f_{c,S}$ be a classifier constructed by WSVM with weights c, and let $\xi^\star_{c,S}$ be the corresponding loss vector. The set of all instance weights $\mathbb{R}^n_+$ is partitioned into two subsets, $\mathcal{W}_+$ and $\mathcal{W}_-$, depending on the sign of $\rho(c, \xi^\star_{c,S})$. Define the "best" weight vectors in each of the two classes as $c_\pm = \arg\min_{c \in \mathcal{W}_\pm} L(f_{c,S})$. If $L(f_{c_-,S}) < L(f_{c_+,S})$, then the best classifier corresponds to the weights that are out of reach for the SVM+, hence, there are no privileged features that will yield an SVM+ classifier as good as $f_{c_-,S}$.

This reasoning motivated us to consider schemes for *learning* the weights that are unrelated to SVM+ and are not restricted by the above constraint.

3.5 How to Choose the Weights

Recall that we are interested in ways of incorporating prior knowledge about the training data. In the SVM+ approach, the role of additional information is played by the privileged features which are used to estimate the loss on the training sample. The same effect, as we have established, can be achieved by importance weighting in WSVM. Taking into account vast amount of work on weighted learning, it seems that re-weighting of misclassification costs is a very powerful method of incorporating prior knowledge. We would like to stress, however, that a critical difference to, for example, the cost-sensitive learning is that we are ultimately interested in minimizing the non-weighted expected loss and *the weights are only used to impose a bias on the learning algorithm.*

We also note that even though SVM+ solutions are contained within WSVM solutions, there is no implication that any of the two algorithms is "better". If privileged features are available, then SVM+ is a reasonable choice. On the other hand, if there are no privileged features or if one has concerns outlined at the end of § 3.4.5, then one may want to consider a more general WSVM method with some problem specific scheme for computing the instance weights.

In the following, we investigate two approaches that make different assumptions about what is additionally available to the learning algorithm at training time. The methods operate in a somewhat idealized setting and are mainly aimed at motivating further research on how to choose the weights. They may be thought of as the empirical counterparts of a more theoretical discussion involving the Oracle SVM in (Vapnik and Vashist, 2009). In particular, the weight learning method of

§ 3.5.3 can be thought of as a way of extracting additional information about the given problem from a validation sample, which is used as a reference.

3.5.1 Why Instance Weighting Is Important?

Let us first motivate the importance of instance weighting with two examples.

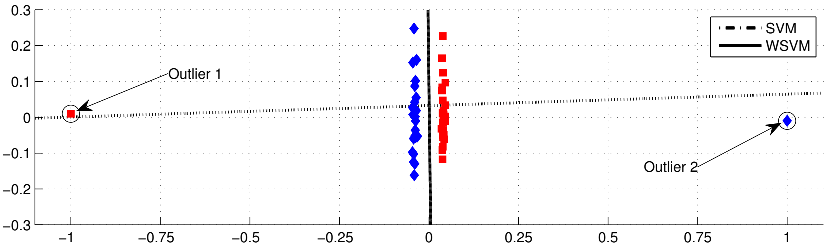


Figure 3.3.: Instance weighting helps against outliers on a toy problem in 2D. The two outliers in the training set have significant negative effect on the SVM model, which has a near chance level performance as a result (horizontal dashed line). In contrast, assigning zero weight to the outliers allows WSVM to recover a near optimal solution (vertical solid line).

Consider the toy problem shown in Figure 3.3. The data comes from two linearly separable blobs, so it is possible to achieve zero test error on them. However, the training sample is contaminated with two outliers that lie extremely far from the optimal decision boundary. Since SVM uses a surrogate loss and not the discrete 0-1 loss, the cost of a point is higher the further the point is from the separating hyperplane. Hence, SVM prefers to keep the two outliers close to the decision boundary, which leads to a near chance level performance on this data set.

Instance weighting, on the other hand, allows one to alter the cost of each point. In particular, if the two outliers are assigned zero weight, then WSVM is able to find a near optimal classifier as illustrated by the vertical solid line.

The second toy problem shown in Figure 3.4 suggests that instance weights can improve the predictive performance even in the nonlinear case, where the problem of extreme outliers is less likely to happen. As before, the issue evolves around the points that lie either too close to or even on the wrong side of the true decision boundary. We use the standard *Nadaraya-Watson estimator*,

$$\eta(x) = \frac{\sum_{i=1}^{n} K_h(x - x_i) y_i}{\sum_{i=1}^{n} K_h(x - x_i)},$$

with the Gaussian kernel and the bandwidth parameter h tuned on the validation set, to obtain an estimate of the conditional probability $\mathbb{P}(Y = 1 \,|\, X = x)$. Our estimate is shown as a heatmap in the background and is used to compute instance weights (reflected by the size of points) as

$$c_i \propto \mathbb{P}(Y = y_i \,|\, X = x_i).$$

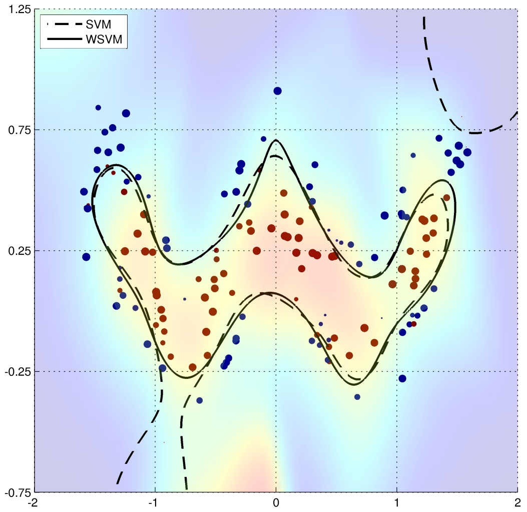


Figure 3.4.: Instance weighting leads to a more stable estimate of the decision boundary in a nonlinear 2D toy problem. The size of a data point corresponds to its weight, which is computed from an estimate of $\mathbb{P}(Y = 1 \,|\, X = x)$ shown in background. WSVM (solid line) is less influenced by outliers than SVM (dashed line) since the outliers are downweighted.

Note that the outliers are downweighted and have less influence on the WSVM decision boundary (solid line) compared to SVM (dashed line). We provide further details on this experiment, including the classification results, in § 3.6.2 (page 56).

3.5.2 Access to an Estimate Of $\mathbb{P}(Y = 1 \,|\, X = x)$

Having full access to the conditional probability $\mathbb{P}(Y = 1 \,|\, X = x)$ is clearly a hypothetical scenario, since in this case the classification problem is solved. However, it is interesting to see how this type of information could be used in construction of good weights. First, note that if $\mathbb{P}(Y = y_i \,|\, X = x_i)$ is available at least for the training sample $S = \{(x_i, y_i)\}$, then one can directly compute the conditional expectation and employ the following estimator:

$$R'(f) \triangleq \frac{1}{n}\sum_{i=1}^{n}\Big[L(f(x_i))\,\mathbb{P}(Y = 1 \,|\, X = x_i) + L(-f(x_i))\,\mathbb{P}(Y = -1 \,|\, X = x_i)\Big],$$

which is an unbiased estimator of $R(f)$:

$$\begin{aligned}\mathbb{E}\,R'(f) &= \mathbb{E}_X\left[L(f(X))\,\mathbb{P}(Y = 1 \,|\, X) + L(-f(X))\,\mathbb{P}(Y = -1 \,|\, X)\right]\\ &= \mathbb{E}_X\,\mathbb{E}_{Y\,|\,X}\left[L(Yf(X)) \,|\, X\right] = \mathbb{E}_{X,Y}\,L(Yf(X)) = R(f).\end{aligned}$$

The property of being biased or not is of asymptotic nature and is arguably of lesser interest in the small sample regime. Following this line of argument, we consider a conservative *weighted risk estimator* given by:

$$\hat{R}_{\mathrm{w}}(f) \triangleq \frac{1}{n}\sum_{i=1}^{n} w(x_i, y_i) L(y_i f(x_i)), \tag{3.16}$$

$$w(x_i, y_i) \triangleq \mathbb{P}(Y = y_i \,|\, X = x_i). \tag{3.17}$$

It is not hard to check that $\hat{R}_{\mathrm{w}}(f)$ is biased:

$$\begin{aligned} \mathbb{E}\,\hat{R}_{\mathrm{w}}(f) &= \mathbb{E}_X\,\mathbb{E}_{Y\,|\,X}\left[w(X,Y)L(Yf(X))\,|\,X\right] \\ &= \mathbb{E}_X\left[L(f(X))\,\mathbb{P}(Y=1\,|\,X)^2 + L(-f(X))\,\mathbb{P}(Y=-1\,|\,X)^2\right] \\ &\le \mathbb{E}_X\left[L(f(X))\,\mathbb{P}(Y=1\,|\,X) + L(-f(X))\,\mathbb{P}(Y=-1\,|\,X)\right] \\ &= \mathbb{E}_X\,\mathbb{E}_{Y\,|\,X}\left[L(Yf(X))\,|\,X\right] = R(f). \end{aligned}$$

More precisely, $\hat{R}_{\mathrm{w}}(f)$ is *conservative* in the sense that the points far from the decision boundary are weighted more, while the points with $\mathbb{P}(Y=1\,|\,X) \approx 0.5$ receive relatively low weight. This behavior is due to the $p \mapsto p^2$ transform being monotonically increasing and strictly convex on $[0,1]$. The monotonicity also ensures the following important property of the obtained weighted estimator when L is the discrete 0-1 loss:

$$\arg\min_f \mathbb{E}\,\hat{R}_{\mathrm{w}}(f) = f^* = \arg\min_f R(f),$$

that is, $\hat{R}_{\mathrm{w}}(f)$ is minimized by the Bayes optimal classifier and, therefore, the learning problem is not altered.

If the bias of $\hat{R}_{\mathrm{w}}(f)$ is a concern, one can tune the weights as the size of the training sample increases. To this end, we consider the following generalization of the weight function in (3.17):

$$c_\tau(x_i, y_i) \triangleq w^\tau(x_i, y_i), \tag{3.18}$$

where $\tau \in [0, \infty)$ is tuned along with the standard regularization parameter. Note that SVM is recovered when the weights are given by $c_0(x_i, y_i) \equiv 1$.

When $\mathbb{P}(Y=1\,|\,X)$ is estimated from a training sample, then WSVM with the weights given by (3.18) will mainly serve as a baseline for the method introduced in the following section. However, it is conceivable that an estimate of the conditional probability could be available from a different source, e.g. from annotations provided by humans. That setting is evaluated later in § 3.6.4 (page 58).

3.5.3 Learning the Weights

Given a training sample S, the weights in WSVM parametrize the set of hypotheses that the algorithm can choose from. As we show next, the weights can also be *learned* along with the classifier f, which will depend on the weights implicitly:

$$c^\star = \arg\min_{c\in\mathbb{R}^n_+} \mathbb{E}\, L(Y f^\star_c(X)), \tag{3.19}$$

$$f^\star_c = \arg\min_{f\in\mathcal{H}} \frac{1}{2}\left\|f\right\|^2 + \sum_{i=1}^{n} c_i L(y_i f(x_i)). \tag{3.20}$$

The optimization problem (3.19), as formulated above, cannot be solved in practice since the underlying probability distribution is unknown. Instead, we replace the true risk $R(f)$ in (3.19) with an empirical risk estimator. To avoid overfitting, however, we require independent data samples in (3.19) and in (3.20). Specifically, we assume that a validation sample S' is available at training time and the weights $c^\star$ are learned as follows:

$$c^\star = \arg\min_{c\in\mathbb{R}^n_+} \sum_{i=1}^{N} L(y'_i f^\star_c(x'_i)).$$

Our idea is inspired by the method of Chapelle et al., (2002) who suggested to tune hyper-parameters of SVM with the squared hinge loss by minimizing certain estimates of the generalization error using gradient descent. The squared hinge loss allows them to additionally assume the hard margin case with a modified Gram matrix $K \leftarrow (K + (1/C)\,\mathbf{I})$, where C is the regularization parameter; see also (Cortes and Vapnik, 1995). That leads to a very specific derivation of the gradient with respect to the hyper-parameters, which is not directly applicable to other loss functions. Instead, we develop a more general scheme that applies to any convex and twice differentiable loss. Our method is instantiated with a smooth version of the hinge loss given below in (3.26), which is constructed to be twice continuously differentiable. Furthermore, while Chapelle et al., (2002) perform optimization in the dual, we optimize (3.20) in the primal, following a more recent line of work by Chapelle, (2007).

The WSVM weights and classifier learning problem is defined as

$$c^\star = \arg\min_{c\in\mathbb{R}^n_+} \sum_{i=1}^{N} L\left(y'_i\left[\bar{K}_i^\top \alpha^\star(c) + b^\star(c)\right]\right), \tag{3.21}$$

$$(\alpha^\star(c), b^\star(c)) = \arg\min_{\alpha,\, b} \frac{1}{2}\alpha^\top K\alpha + \sum_{i=1}^{n} c_i L\left(y_i\left[K_i^\top\alpha + b\right]\right), \tag{3.22}$$

where the classifier f is expressed as

$$f(x) = \sum_{i=1}^{n} \alpha_i k(x_i, x) + b,$$

the matrix $\bar{K}$ is defined as $\bar{K}_{ij} = k(x_i, x'_j)$ for $i = 1, \dots, n$, $j = 1, \dots, N$, and the vectors K_i, $\bar{K}_i$ are the ith columns of K and $\bar{K}$ respectively.

Note that f depends on the weights c implicitly via the second optimization problem and the main challenge in applying gradient descent is the computation of $\partial\alpha^\star/\partial c$ and $\partial b^\star/\partial c$. These derivatives can be computed via implicit differentiation from the optimality conditions as we show below.

Theorem 3.5. Let the loss function L be convex and twice continuously differentiable and let the Gram matrix K be (strictly) positive definite. Define vectors u and v componentwise for $i = 1, \dots, n$ as

$$u_i \triangleq y_i L'\left(y_i\left[K_i^\top\alpha^\star + b^\star\right]\right), \qquad v_i \triangleq c_i L''\left(y_i\left[K_i^\top\alpha^\star + b^\star\right]\right),$$

where L' and L'' are respectively the first and second derivatives of L, and $(\alpha^\star, b^\star)$ is a solution of (3.22) for the given weights c. If $v \neq \mathbf{0}$, then the solution is unique, the points $\alpha^\star$ and $b^\star$ are continuously differentiable in c, and the corresponding derivatives can be computed as

$$\begin{bmatrix} \frac{\partial\alpha^\star}{\partial c} \\ \frac{\partial b^\star}{\partial c} \end{bmatrix} = -\begin{bmatrix} \mathbf{I} + \mathbf{diag}(v)K & v \\ \mathbf{1}^\top & 0 \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{diag}(u) \\ \mathbf{0}^\top \end{bmatrix}. \tag{3.23}$$

Proof. Uniqueness is established using the same argument as in Theorem 3.2. Uniqueness of α follows from (strict) positive definiteness of K. For b, let $b_1^\star$ and $b_2^\star$ be two optimal points and define $b_t^\star = (1-t)b_1^\star + tb_2^\star$. Considering the difference of the objective function at $b_t^\star$ and $b_1^\star$, and differentiating twice in t, we get

$$(b_2^\star - b_1^\star)\mathbf{1}^\top v = 0 \quad \Rightarrow \quad b_2^\star = b_1^\star.$$

The optimality conditions of (3.22) yield

$$K(\alpha^\star + \mathbf{diag}(u)c) = \mathbf{0}, \qquad u^\top c = 0.$$

Since K is non-singular it can be dropped from the first equation. Computation of the total derivatives yields the linear system below.

$$\begin{bmatrix} \mathbf{I} + \mathbf{diag}(v)K & v \\ v^\top K & \mathbf{1}^\top v \end{bmatrix} \begin{bmatrix} \frac{\partial\alpha^\star}{\partial c} \\ \frac{\partial b^\star}{\partial c} \end{bmatrix} = -\begin{bmatrix} \mathbf{diag}(u) \\ u^\top \end{bmatrix} \tag{3.24}$$

Note that (3.24) is equivalent to (3.23) since the last equation can be equivalently replaced by the sum of the first n equations minus the last one. To apply the implicit function theorem, it remains to show that the matrix in (3.23) is invertible.

Recall that the determinant of a block matrix factors as the determinant of a block and its Schur complement. It is thus sufficient to show that

$$\det\left(\mathbf{I} + \mathbf{diag}(v)K\right) \neq 0,$$
$$\mathbf{1}^\top\left(\mathbf{I} + \mathbf{diag}(v)K\right)^{-1} v \neq 0.$$

Assume w.l.o.g. that the first m components of v ($m \leq n$) are non-zero and define $M \triangleq \mathbf{I} + \mathbf{diag}(v)K$. We have

$$M = \begin{bmatrix} A & B \\ C & D \end{bmatrix} = \begin{bmatrix} \mathbf{I}_m + \mathbf{diag}(v_m)K_m & B \\ \mathbf{0}_{n-m,m} & \mathbf{I}_{n-m} \end{bmatrix},$$

where the B block is irrelevant for our purposes. It now follows that

$$\begin{aligned} \det(M) &= \det(D)\det(A - BD^{-1}C) = \det(A) \\ &= \det(\mathbf{diag}(v_m))\det(\mathbf{diag}(v_m)^{-1} + K_m) \neq 0, \end{aligned}$$

where we use that $\mathbf{diag}(v_m)^{-1}$ is positive definite since $v_m \succ \mathbf{0}_m$ due to convexity of L and the assumption $v \neq \mathbf{0}$. To complete the proof, we have

$$\begin{aligned} \mathbf{1}^\top M^{-1} v &= \mathbf{1}^\top \begin{bmatrix} A^{-1} & -A^{-1}BD^{-1} \\ \mathbf{0}_{n-m,m} & D^{-1} \end{bmatrix}^{-1} v \\ &= \mathbf{1}_m^\top \left(\mathbf{diag}(v_m)^{-1} + K_m\right)^{-1} \mathbf{1}_m > 0. \end{aligned}$$

□

Note that this scheme directly applies to many popular loss functions including the squared hinge loss and the logistic loss. Moreover, for the logistic loss it always holds that $v \neq \mathbf{0}$ for any $c \neq \mathbf{0}$ due to strict convexity of the loss.

If $v = \mathbf{0}$, it can be seen that $\alpha^\star$ is still uniquely defined and is continuously differentiable in c for any fixed b. The "gradient" in this case is given by

$$\partial\alpha^\star/\partial c = \mathbf{diag}(u), \qquad \partial b^\star/\partial c = \mathbf{0}^\top. \tag{3.25}$$

Theorem 3.5 applies to convex and twice differentiable loss functions. Since the standard hinge loss employed in SVM+ and WSVM is not smooth, we consider an approximation that is twice continuously differentiable and preserves certain desirable properties of the hinge loss. Specifically, we have chosen the loss function defined as (see Figure 3.5)

$$L_\delta(t) \triangleq \begin{cases} 1 - t - \delta & \text{if } t \leq 1 - 2\delta, \\ \frac{(1-t)^3(t-1+4\delta)}{16\delta^3} & \text{if } 1 - 2\delta < t < 1, \\ 0 & \text{if } t \geq 1. \end{cases} \tag{3.26}$$

Note that other differentiable approximations of the hinge loss have been considered in the literature, however, they either produce superlinear costs on extreme outliers, or have only the first derivative while we require two. In contrast, the function (3.26) is twice continuously differentiable and exhibits certain similarities to the hinge loss: (i) it does not penalize points with the margin $t \triangleq y_i f(x_i) \geq 1$, and (ii) it grows linearly for $t \leq 1 - 2\delta$.

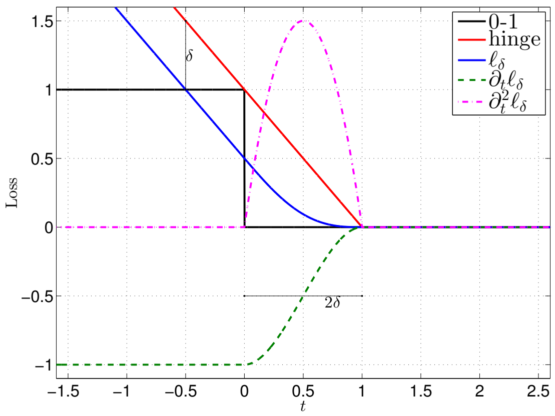


Figure 3.5.: The 0-1 loss, the hinge loss, and the twice differentiable loss L_δ with its derivatives. Note that L_δ approaches the hinge loss as $\delta \to 0$.

With the approximate hinge loss L_δ defined above, $v \neq \mathbf{0}$ means that at least one of the data points has to fall into the strictly convex region of the loss where $L_\delta''(t) > 0$. Clearly, this presents us with a trade-off between having a good approximation of the hinge loss (small δ) and a higher chance of being able to compute "correct" gradients using Theorem 3.5 (large δ). To address this issue in our experiments, we tune $\delta \in [0.01, 1]$ on a validation set.

3.6 Experiments

In this section, we present empirical evaluation of the algorithms introduced in this chapter. In our experiments, we use the WSVM implementation by Chang and Lin, (2011) and the code for the SVM+ provided by Pechyony and Vapnik, (2011). The weight learning problem is solved using our implementation of the BFGS algorithm (Nocedal and Wright, 2006). The general experimental setup is similar to that of Vapnik and Vashist, (2009): parameters are tuned on the validation set, which is not used for training, and performance is evaluated on the test set. Training subsets are randomly sampled from a fixed training set, and results over multiple runs are aggregated showing the mean error rate as well as the standard deviation. Depending on the experiment, the validation set is either fixed or subsampled randomly as well. The Gaussian RBF kernel is used in all experiments and features are rescaled to be in $[0, 1]$. The weights in (3.17) are computed from $\eta(x) = 2\,\mathbb{P}(Y = 1 \,|\, X = x) - 1$, which is either given directly by human experts or estimated using the Nadaraya-Watson technique:

$$\eta(x) = \frac{\sum_{i=1}^{n} K_h(x - x_i) y_i}{\sum_{i=1}^{n} K_h(x - x_i)}, \tag{3.27}$$

where K_h is the Gaussian kernel with bandwidth h.

Note that in all experiments each algorithm has access to exactly the same data, and the only difference between different splits is which data is used to construct a classifier (training) and which is used to tune the hyper-parameters (validation).

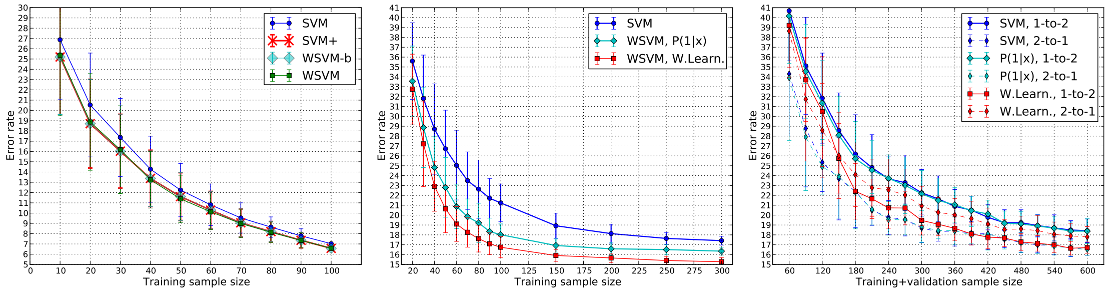


Figure 3.6.: SVM, SVM+, and WSVM comparison on MNIST (left) and synthetic data. **Left:** WSVM replicates SVM+ in our reproduction of the experiment by Vapnik and Vashist, (2009). **Middle:** Weight learning leads to significant performance improvement on synthetic data when a large validation set is available. **Right:** The same setting, but the training-to-validation splits are 1-to-2 and 2-to-1, which is more realistic in practice.

3.6.1 WSVM Replicates SVM+

We begin with an experimental verification of our theoretical findings from § 3.4. We reproduce the handwritten digit recognition experiment of Vapnik and Vashist, (2009), where the task is to discriminate between 5's and 8's, which are taken from the MNIST database and downsized to 10 × 10 pixels. We use the features provided by the authors and obtain the error rates shown in Figure 3.6, left. Our results are averaged across 100 runs and range from 10 to 100 training examples.

Following Theorem 3.3 (page 40), the weights for WSVM are computed as $c = \alpha^\star + \beta^\star$, where $\alpha^\star$ and $\beta^\star$ come from the SVM+ solution. We observed that indeed the solvers compute $\alpha^\star_{\text{WSVM}} \approx \alpha^\star_{\text{SVM+}}$. However, we also observed that in general $b^\star_{\text{WSVM}} \neq b^\star_{\text{SVM+}}$, which is explained by non-uniqueness of b (Theorem 3.1). If $b^\star_{\text{SVM+}}$ from the SVM+ model is used (WSVM-b in the plot), then the two classifiers are identical. However, if b is tuned within the constraints imposed by the KKT conditions (plain WSVM in the plot), then minor differences appear.

3.6.2 Synthetic Data

We now turn to the problem of choosing the weights for WSVM and evaluate the two weight generation schemes introduced in § 3.5. In this experiment, data comes from a mixture of 2D Gaussians that form a nonlinear shape resembling the letter W, see Figure 3.4 (page 50). Similar to the previous setting, we sample training subsets of different size from a fixed training set of 400 examples; then tune the hyper-parameters, estimate the $\mathbb{P}(Y = 1 \,|\, X = x)$ using (3.27), and perform weight learning on a validation set of size 4000, and finally test on a separate set of size 2000. The results are averaged over 50 runs and reported in Figure 3.6, middle. Note that, just like in the experiment of Vapnik and Vashist, (2009), this is an idealistic setting where the validation set is so large that model selection is close to optimal. In practice, one would never split the available sample as 1-to-40, therefore, we also evaluate more realistic splits 1-to-2 and 2-to-1 next.

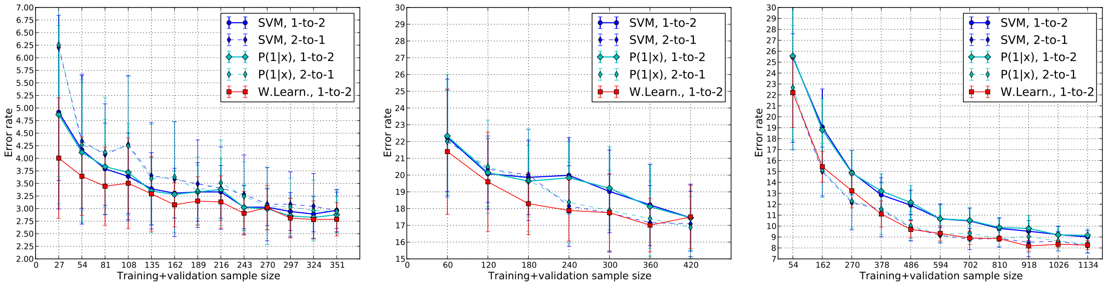


Figure 3.7.: SVM and WSVM error rates on UCI datasets with training-to-validation splits of 1-to-2 and 2-to-1. **Left:** Breast Cancer Wisconsin. **Middle:** Mammographic Mass. **Right:** Spambase.

Figure 3.6, right, shows results of a similar experiment where the validation sample is not fixed, but rather obtained by splitting the available training data. Since validation samples are now small, the estimation of $\mathbb{P}(Y = 1 \mid X = x)$ fails and the corresponding WSVM performs on par with the standard SVM. Weight learning, however, still yields performance improvement on the 1-to-2 splits. Moreover, WSVM with weight learning is able to achieve similar error rates as SVM trained on *twice* as much data. On the 2-to-1 splits, however, we observe that weight learning overfits and we omit that setting in further experiments.

Note that it is not surpsing that for the weight learning to succeed, the amount of validation data should be at least comparable to or even exceed the number of weights c_i that are to be learned.

3.6.3 UCI Data

In this set of experiments, we evaluate weight learning on three publicly available datasets from the UCI repository (Frank and Asuncion, 2010). For every dataset, we first remove any records with missing values and then split the remaining data randomly into training and test sets of roughly equal size approximately preserving the initial class distribution.

Dataset	Features	Training	Test
Breast Cancer Wisconsin	9	351	332
Mammographic Mass	4	420	410
Spambase	57	2430	2171

Table 3.1.: Statistics of datasets from the UCI repository.

Table 3.1 summarizes the statistics of the obtained UCI datasets. Smaller subsets are sampled from the training data, and split further into training and validation sets as 1-to-2 and 2-to-1. The subsets sampling process is repeated 20 to 50 times depending on the amount of data. The rest of the experimental setup is the same as before. Next, we briefly discuss the results presented in Figure 3.7.

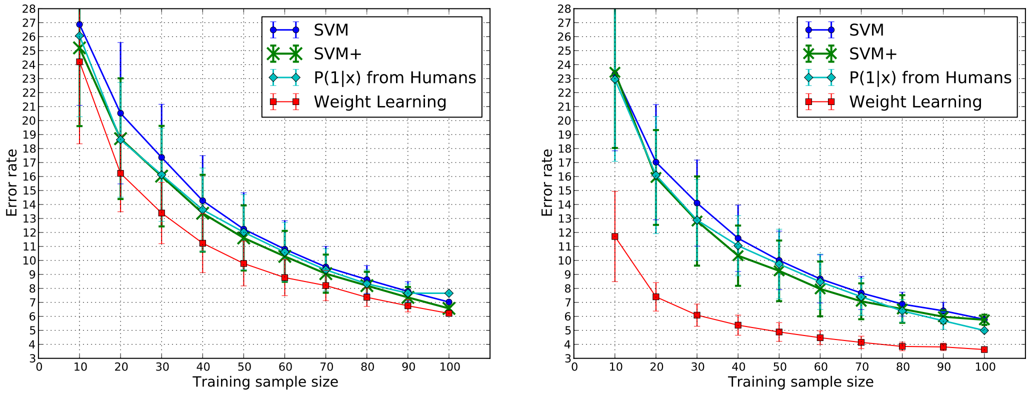


Figure 3.8.: Error rate comparison in the handwritten digit recognition experiment of Vapnik and Vashist, (2009). $\mathbb{P}(Y = 1 \mid X = x)$ was estimated from human rankings. **Left:** The original setting. **Right:** The extended setting where each digit is translated by one pixel in each of the eight directions.

Breast Cancer Wisconsin. Weight learning on the 1-to-2 split in this experiment performs on par or better than SVM trained on either of the two splits. SVM performs worse on the 2-to-1 split, which we attribute to overfitting. This is not too surprising considering the small amount of data and the capacity of the RBF kernel, which makes the weight learning result even more remarkable.

Mammographic Mass. On this dataset, weight learning performs on par or better than SVM on all splits and all subsets, except for the last one. Even though the variance is quite large on the small samples, one can see that the improvement is consistent across all the subsets from 60 to 360 examples. On the last subset (420 examples), weight learning did not yield any improvement and the amount of training data was sufficient for SVM to achieve comparable performance.

Spambase. Here, the benefit of weight learning is more prominent, as WSVM with the learned weights achieves the performance comparable to SVM that is trained on *twice* as much data. Note that the variance is now much smaller and we see, for example, 2% improvement over the SVM trained on the same data, when the sample size is 486. Therefore, the additional knowledge about the importance of each training data point, which is represented as instance weights, results in more efficient use of the limited training sample.

3.6.4 Handwritten Digit Recognition

In this section, we consider the handwritten digit recognition experiment of Vapnik and Vashist, (2009) and evaluate our weight generation schemes on that data.

The first scheme is described in § 3.5.2 (page 50) and is based on the assumption that digit ranking is available as additional information. Specifically, the confidence scores $\hat{\eta}_i \in [-1, 1]$ are provided along with the class labels $y_i = \pm 1$, and are used as the estimates of the true *regression function*:

$$\hat{\eta}_i \approx \mathbb{E}_{Y \mid X}[Y \mid X = x_i] = 2\,\mathbb{P}(Y = 1 \mid X = x_i) - 1.$$

Such confidence scores may be available on the datasets where robust annotation is obtained by aggregating labelings from several human experts. In fact, that is exactly how we collected these scores as we describe next.

We collected additional annotation in the form of rankings from three human experts. The humans were presented with random samples of the 10×10 pixel digits and were asked to label them using one of 5 possible labels, which we translated into a score in $\{-1, -0.5, 0, 0.5, 1\}$. Each of the 100 digits from the training set was ranked 16 times and the average score was then used. Finally, the weights for WSVM were computed using (3.18) (page 51). A similar in spirit setting was previously considered by Wu and Srihari, (2004).

Figure 3.8 shows the experimental results. We observe that additional information from human experts helps on small subsets, but its influence diminishes on larger subsets. This might be in part due to the difference in image representation used by SVMs and humans. In particular, humans' recognition of digits is translation invariant, while the pixelwise representation is not. This leads us to our final experiment on the extended version of that dataset.

We extend the original training sample of 100 digits by shifting each digit by one pixel in all eight directions, thus obtaining 9 times the initial sample size. We assume that both the human rankings and the privileged features from the experiment of Vapnik and Vashist, (2009) are unaffected by such translations and we simply replicate them. The experimental results are presented in Figure 3.8, right. Note that WSVM with human rankings is now consistently on par or better than SVM and is somewhat comparable to SVM+.

Remarkably, weight learning gives significant performance boost on the extended version of the dataset, which shows that it can be successfully combined with other sources of additional information, like the hint about translation invariance in this case. Interestingly enough, Lauer and Bloch, (2008) discussed the possibility of combining the virtual sample method, which we used to extend the training set, with instance weighting where each virtual point is given a confidence score c_i. Our weight learning algorithm does exactly that, but without trying to model the measure of confidence explicitly. Instead, it attempts to directly optimize an estimate of the expected loss $R(f)$.

3.7 Conclusion

In this chapter, we explored the framework of *learning using privileged information* that was recently introduced by Vapnik and Vashist, (2009). In particular, we studied certain properties of the SVM+ algorithm, such as uniqueness of its solution, and showed that it is closely related to the well-known weighted SVM (WSVM). Furthermore, we revealed that all SVM+ solutions are constrained to have a certain dependency between the dual variables and the incurred loss on the training sample, and that the prior knowledge from the SVM+ framework can be encoded via instance weights in WSVM.

Privileged information is not the only source of additional information that could be used to improve the performance when training data is limited. In particular, we proposed a *weight learning* method in § 3.5.3 which allows one to learn the WSVM weights directly from data using a validation set. Experimental results confirmed our intuition that importance weighting is a powerful technique of incorporating prior knowledge which can lead to significant performance improvements.

In the next chapter, we consider another approach – *multitask learning* – which can also be used to improve the generalization performance of learning algorithms when the amount of training data is limited.

4 Scalable Multitask Representation Learning for Scene Classification

We continue our exploration of techniques that improve classification performance of learning algorithms in the regime of limited training data. In Chapter 3, we considered binary classification problems and saw that *privileged features* as well as *importance weighting* can help in training classifiers that are more robust against outliers and outperform the classical SVM when the number of training examples is small. Here, we consider *multiclass* problems and use a **multitask learning** framework to learn an effective low dimensional image representation that is shared across multiple classes.

The underlying idea of multitask learning is that learning tasks jointly is better than learning each task individually. In particular, if only a few training examples are available for each task, sharing a jointly trained representation improves classification performance. Here, we view classifiers for each class as separate tasks and propose to train them jointly with the shared representation. Consequently, the classifiers are able to exploit latent inter-class correlations that may exist between closely related tasks, which ultimately improves the performance.

Our method employs dual optimization and is scalable with respect to the original feature dimension. In particular, it can be used with high dimensional image descriptors based on the Fisher Vector encoding (Sánchez et al., 2013). We consistently outperform the current state of the art on the SUN 397 scene classification benchmark with varying amounts of training data.

The material in this chapter is based on the following publication:

- M. Lapin, B. Schiele, and M. Hein (2014b). "Scalable Multitask Representation Learning for Scene Classification." In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*

4.1 Introduction

Recently, Sánchez et al., (2013) showed that feature encoding based on the Fisher kernel yields significantly better results compared to the bag of visual words (BOW) model. The Fisher kernel was introduced by Jaakkola, Haussler, et al., (1999) as a model that combines the benefits of generative and discriminative approaches, and was later popularized in computer vision by Perronnin et al., (2010), who proposed an encoding scheme based on visual vocabularies: the *Fisher Vector (FV)*. FV encoding was instrumental to achieving state of the art performance in scene classification, as shown by Juneja et al., (2013). However, one of the major shortcomings of FV compared to BOW is that FV descriptors are dense and high dimensional. In our experiments, for example, we work with features that have over 260K dimensions, which obviously presents a scalability problem.

The de facto standard to address image, object, and scene classification today is to train separate classifiers in a one-vs-all (OVA) regime. However, it has long been argued that joint learning of class representations and knowledge transfer across classes are the key ingredients to solving the following outstanding problems in computer vision: (i) scaling classification to a large number of categories, and (ii) learning from a small number of training examples. This calls for a multitask learning framework where each binary classifier becomes a separate task and all classifiers as well as the shared representation are learned jointly. In contrast to the independent OVA training, this enables the classifiers to exploit inter-class correlations that may exist between related classes.

While there has been significant progress in the area of multitask learning in the last decade both on the theoretical as well as the algorithmic side (Argyriou et al., 2008; Caruana, 1997; Kang et al., 2011; Maurer et al., 2013; Romera-Paredes et al., 2012), most of the proposed methods do not scale well to very large feature dimensions encountered in computer vision problems. To enable inter-class transfer using modern image representations, we propose, as our first main contribution, a new scalable formulation of *multitask representation learning*. Our method jointly learns a linear mapping into a lower dimensional space which is then used to build the classifiers for each class.

To address the large scale of the resulting optimization problem, we adapt the *stochastic dual coordinate ascent* (SDCA) optimization scheme recently developed by Shalev-Shwartz and Zhang, (2013b). The SDCA algorithm can be applied to smooth as well as Lipschitz losses (e.g. the hinge loss), it has a clean stopping criterion (the duality gap), and fast convergence rate which is superior to that of the vanilla stochastic gradient descent. Importantly, the algorithm operates on dual variables which is a significant advantage in our setting with the number of training examples being much smaller than the feature dimension. Our method, which we call MTL-SDCA, is efficient since variable updates for the hinge loss can be computed in closed form.

Finally, we would like to highlight a connection between multitask learning of Maurer et al., (2013) and supervised dictionary learning of Mairal et al., (2012). Both methods learn a set of vectors, called the dictionary, that defines a subspace that is shared across tasks. However, dictionary learning is usually formulated to optimize the reconstruction error in image processing applications, while our primary goal is to find a new representation where the classes are well separated.

As a second contribution, we apply our MTL-SDCA method to the challenging problem of scene classification on the SUN 397 benchmark (Xiao et al., 2010). In line with previous findings, we observe that an important ingredient for the best performance on this dataset is the high dimensional FV encoding. Surprisingly, it achieves excellent performance even with a *single* image descriptor based on SIFT (Lowe, 2004). We also confirm that the state of the art performance in scene classification is a result of a carefully engineered feature extraction pipeline, which we design following the established best practices (Chatfield et al., 2011; Juneja et al., 2013; Sánchez et al., 2013).

We improve upon the state of the art[1] on SUN 397, which asserts efficiency and effectiveness of our MTL-SDCA method. Furthermore, we validate that the approach performs well even in the case where only little training data is available, as it is expected from a multitask learning method.

4.2 Multitask Learning

In this section, we introduce the multitask representation learning framework and develop a scalable optimization scheme based on SDCA. We discuss a general multitask setting first, and then specialize to multiclass problems, where the shared representation and the classifiers are learned jointly.

Here, we introduce our notation and the learning problem. Let T be the number of tasks (classes), and consider a training sample $\{(x_i, y_{ti}) \,|\, 1 \leq t \leq T,\ 1 \leq i \leq n\}$, where $x_i \in \mathbb{R}^d$ and $y_{ti} \in \{\pm 1\}$. We assume that all tasks have the same training examples (but different labels), even though this can be easily generalized. The setting we have in mind is that the feature space is high dimensional, but the sample size is limited, that is $d \gg n$. This is quite common in modern computer vision problems: one can easily have $d \geq 10^5$ with the FV encoding, while n is at most on the order of 10^4 in the SUN 397 challenge, see § 4.4 (page 69).

We learn a matrix U in $\mathbb{R}^{d \times k}$ with $k \ll d$, which is used to generate the low dimensional representation of the data, $z_i = U^\top x_i$. Moreover, we learn linear predictors w_t in $\mathbb{R}^k$ that operate on the data in the low dimensional space. Let $X \in \mathbb{R}^{d \times n}$ be the matrix of stacked feature vectors x_i, $W \in \mathbb{R}^{k \times T}$ the matrix of stacked predictors w_t, $K = X^\top X$ the Gram matrix, and $M = W^\top W$. Note that since $\langle w_t, U^\top x_i \rangle = \langle U w_t, x_i \rangle$, the matrix U can be interpreted as a *dictionary* and vectors w_t as the corresponding decomposition coefficients (Maurer et al., 2013).

Multitask Representation Learning

Our multitask representation learning problem is formulated as

$$\min_{U \in \mathbb{R}^{d \times k}} \frac{1}{T} \sum_{t=1}^{T} \min_{w_t \in \mathbb{R}^k} P_{U,t}(w_t) + \frac{\mu}{2} \|U\|_F^2 , \tag{4.1}$$

where the objective for task t given a fixed U is

$$P_{U,t}(w_t) = \frac{1}{n} \sum_{i=1}^{n} L_{ti} \left(\langle w_t, U^\top x_i \rangle \right) + \frac{\lambda}{2} \|w_t\|_2^2 , \tag{4.2}$$

and $\lambda > 0$, $\mu > 0$ are the regularization parameters, L_{ti} is a convex margin-based loss function, and $\|\cdot\|_F$ denotes the Frobenius norm. For the hinge loss, which we consider in our experiments, we let

$$L_{ti} \left(\langle w_t, U^\top x_i \rangle \right) \triangleq \max \left\{ 0, 1 - y_{ti} \langle w_t, U^\top x_i \rangle \right\} .$$

[1] At the time of publication of (Lapin et al., 2014b).

Here, we keep the general notation for the loss function, but instantiate it with the hinge loss when we discuss our optimization algorithm and experiments.

Note that the inner subproblems are standard OVA SVMs trained in a *lower dimensional subspace*, which is determined by the matrix $U \in \mathbb{R}^{d \times k}$. The latter is learned jointly for all tasks which facilitates knowledge transfer across the classes. This is of particular interest when the amount of training examples per class is limited and at least some of the classes are related.

Let us now discuss the relation to an alternative multitask feature learning formulation proposed by Argyriou et al., (2008):

$$\min_{\substack{U \in \mathbb{R}^{d \times d}, UU^T = \mathbf{I} \\ W \in \mathbb{R}^{d \times T}}} \sum_{t=1}^{T} \sum_{i=1}^{n} L_{ti} \left(\langle w_t, U^\top x_i \rangle \right) + \gamma \left\| W \right\|_{2,1}^2 , \tag{4.3}$$

where $\|W\|_{2,1}^2 = \sum_{i=1}^d \|w_{(i)}\|_2$ and $w_{(i)} \in \mathbb{R}^T$ are the *rows* of W. The key difference to our approach is that we work with a low dimensional representation $U \in \mathbb{R}^{d \times k}$, whereas the method above works with a *square* matrix $U \in \mathbb{R}^{d \times d}$ and enforces certain features to be discarded via the sparsity inducing penalizer $\|W\|_{2,1}^2$, which also couples the tasks. While (4.3) is convex and, therefore, has a strong theoretical guarantee of convergence to the global optimum, it does not scale to a high dimensional feature representation, since $U \in \mathbb{R}^{d \times d}$ is a dense matrix that requires $O(d^2)$ memory. Our approach is scalable since our matrix requires only $O(k\,d)$ memory with $k \ll d$. Moreover, we enforce the coupling of the tasks directly by requiring that U maps to a low dimensional subspace. Therefore, we do not need to additionally enforce the coupling of tasks via a sparsity enforcing regularizer on the predictors w_t. This allows to formulate the optimization problem in a way that it reduces to standard OVA SVMs when $\mu = 0$, which is not possible in the framework of Argyriou et al., (2008).

4.2.1 MTL-SDCA Algorithm

The optimization problem (4.1) of our multitask representation learning framework is *biconvex*: it is convex in $W \in \mathbb{R}^{k \times T}$ for a fixed U and vice versa. It is not jointly convex in U and W, which is prevalent to most multitask formulations. A common optimization method in that case is block coordinate descent, see e.g. (Gorski et al., 2007, Algorithm 4.1). We alternate between fixing U and optimizing W, and then fixing W and optimizing U. Each subproblem is convex and one achieves monotonic descent in each iteration. This guarantees convergence to a critical point of the objective (4.1), see (Gorski et al., 2007), which is a standard convergence result for nonconvex problems. For the two convex subproblems, we propose specialized variations of SDCA, which is currently among the state of the art algorithms in large scale optimization (Shalev-Shwartz and Zhang, 2013b). We summarize our **MTL-SDCA** method in Algorithm 4.1.

Scalability of our approach crucially depends on the algorithm for learning $U \in \mathbb{R}^{d \times k}$. The choice of an algorithm that solves the *dual* problem is primarily motivated by our experiments on the SUN 397 benchmark. We use dense high

Algorithm 4.1 MTL-SDCA

```
Input: data $\{(x_i, y_{ti})\}$, initial $U^{(0)}$, parameters $\lambda$, $\mu$, $\epsilon$
Let: $W^{(0)} = \mathbf{0}$.
repeat $\{s = 1, \dots\}$
    for $t = 1$ to $T$ do
        Train OVA SVMs on $z_i = {U^{(s-1)}}^\top x_i$ (using SDCA):
        $w_t^{(s)} \leftarrow \arg\min_w (1/n) \sum_{i=1}^n L_{ti}\left(\langle w_t, {U^{(s-1)}}^\top x_i \rangle\right) + (\lambda/2) \|w_t\|_2^2$
    end for
    Update the shared representation $U$ (using SDCA with updates (4.7)):
    $U^{(s)} \leftarrow \arg\min_{U \in \mathbb{R}^{d \times k}} (1/nT) \sum_{i,t} L_{ti}\left(\langle w_t^{(s)}, U^\top x_i \rangle\right) + (\mu/2) \|U\|_F^2$
until change in variables is below $\epsilon$
```

dimensional feature vectors with the number of dimensions d being an order of magnitude larger than the number of training examples n, which makes dual optimization a natural choice.

For simplicity, we describe the algorithm in terms of primal variables U and W. However, to be computationally efficient, our implementation works only with the corresponding dual variables α and precomputed kernel matrices K and M, which in our setting fit into memory. The actual U and W are never computed at any stage. Further technical details can be found in § B.1 (page 187).

Learning W. Note that learning the predictor matrix $W \in \mathbb{R}^{k \times T}$ when U is fixed is the easier subproblem as the problems for each task decouple. Thus they can be trained in parallel using any SVM solver and the choice of SDCA here is more a matter of convenience.

Learning U. We now show that the matrix U can be learned efficiently via an adaptation of the SDCA algorithm of Shalev-Shwartz and Zhang, 2013b. If W is fixed, the problem (4.1) reduces to

$$\min_{U \in \mathbb{R}^{d \times k}} \frac{1}{nT} \sum_{i,t} L_{ti}\left(\langle w_t, U^\top x_i \rangle\right) + \frac{\mu}{2} \|U\|_F^2 . \tag{4.4}$$

The analogy to SVM now comes from the fact that

$$\langle w_t, U^\top x_i \rangle = \langle U, x_i w_t^\top \rangle.$$

Therefore, we can interpret U as the weight vector of an SVM model with the feature representation $x_i w_t^\top$. Moreover, note that the Frobenius norm of U is nothing else but the Euclidean norm of the matrix U rearranged as a vector. This analogy allows us to rely on SDCA convergence results that were developed for SVM and expect the same convergence guarantees when learning the matrix U. However, as the correspondence may not be obvious, we derive efficient SDCA updates specifically for the case of learning U with the hinge loss. Our derivation is based on Fenchel duality which is briefly covered in Appendix A.2 (page 183).

The Fenchel dual problem associated with (4.4) is

$$\max_{\alpha \in \mathbb{R}^{T \times n}} D_W(\alpha), \tag{4.5}$$
$$D_W(\alpha) = \frac{1}{nT} \sum_{i,t} -L_{ti}^*(-\alpha_{ti}) - \frac{\mu}{2} \left\| \frac{1}{\mu n T} \sum_{i,t} \alpha_{ti} x_i w_t^\top \right\|_F^2,$$

where L_{ti}^* is the convex conjugate of L_{ti}, and for the hinge loss we have

$$L_{ti}^*(-b) = \begin{cases} -y_{ti} b & 0 \le y_{ti} b \le 1, \\ \infty & \text{otherwise.} \end{cases}$$

Note that α is a T by n matrix of dual variables, where the ith column is associated with the training example (x_i, y_i), and the row t is associated with the task t. Let

$$U(\alpha) \triangleq \frac{1}{\mu n T} \sum_{i,t} \alpha_{ti} x_i w_t^\top, \tag{4.6}$$

then, from the optimality conditions, we have $U^* = U(\alpha^*)$, where U^* is the solution of the primal problem (4.4) and α^* is the solution of the dual problem (4.5).

We solve the dual problem using SDCA. At every step s, an index i in $\{1, \ldots, n\}$ and a task t in $\{1, \ldots, T\}$ are chosen uniformly at random. The update of $\alpha_{ti}^{(s)}$ is then computed as

$$\alpha_{ti}^{(s)} = \alpha_{ti}^{(s-1)} + \Delta\alpha_{ti},$$

where $\Delta\alpha_{ti}$ is the stepsize that is chosen to achieve maximal ascent of the dual objective $D_W(\alpha)$ when all other variables are fixed. We have

$$\Delta\alpha_{ti} = \arg\max_{a \in \mathbb{R}} -L_{ti}^*\Big(-(\alpha_{ti}^{(s-1)} + a)\Big) - a\Big\langle U(\alpha^{(s-1)}), x_i w_t^\top \Big\rangle - \frac{a^2}{2\mu n T} \left\| x_i w_t^\top \right\|_F^2,$$

which for the hinge loss can be computed in closed form.

Efficient updates of α. Following Shalev-Shwartz and Zhang, (2013a), we provide a *closed form solution* for $\Delta\alpha_i$ when $L_{ti}(a) = \phi_\gamma(y_{ti} a)$ is the smooth hinge loss, with ϕ_γ defined as

$$\phi_\gamma(a) \triangleq \begin{cases} 1 - a - \gamma/2 & a \le 1 - \gamma, \\ \frac{1}{2\gamma}(1-a)^2 & 1 - \gamma < a < 1, \\ 0 & a \ge 1. \end{cases}$$

In our experiments, we set $\gamma = 0$ which recovers the standard (non-smooth) SVM hinge loss. The formula for the update $\Delta\alpha_{ti}$ is given below:

$$\Delta\alpha_{ti} = y_{ti} \max\Bigg(- y_{ti}\alpha_{ti}^{(s-1)}, \min\Bigg(1 - y_{ti}\alpha_{ti}^{(s-1)}, \\ \frac{1 - y_{ti}x_i^\top U(\alpha^{(s-1)})w_t - \gamma y_{ti}\alpha_{ti}^{(s-1)}}{\frac{1}{\mu n T}\|x_i\|_2^2 \|w_t\|_2^2 + \gamma}\Bigg)\Bigg). \tag{4.7}$$

Note that the norms $\|x_i\|_2^2$ and $\|w_t\|_2^2$ are directly available from the precomputed matrices K and M, and the inner product $x_i^\top U(\alpha^{(s-1)})w_t$ can be computed using (4.6). We provide further technical details in § B.1 (page 187).

Initialization. In all the experiments, we let $k = T$. This choice is motivated by a two-layer architecture, where the output of OVA SVMs is fed into a second layer of OVA classifiers. In this case, we have a natural initialization for U, which is required by our MTL-SDCA algorithm. Specifically, we let $U^{(0)} = W_{\mathrm{SVM}}$, where $W_{\mathrm{SVM}} \in \mathbb{R}^{d \times T}$ is the matrix of stacked predictors $\tilde{w}_t$ that have been trained using the *original* features x_i. This initialization worked well in our experiments.

Stopping criterion. We use the *relative duality gap* as a stopping criterion in our SDCA algorithms that solve the two subproblems of learning W and U. Let P and D be the primal and dual objectives, the condition we check is

$$\Big(P(U(\alpha)) - D(\alpha)\Big) / \max\Big\{ |P(U(\alpha))|, |D(\alpha)| \Big\} < \epsilon,$$

with $\epsilon = 10^{-3}$. In the master problem, we stop when the change in dual variables of the two subproblems is below ϵ as measured by the root mean square error (RMSE), which is defined as $\mathrm{RMSE}(\Delta) = \sqrt{(\sum_{i=1}^m \Delta_i^2)/m}$.

4.2.2 MTL-SDCA Extensions

Our method fully benefits from the generality of the SDCA framework which can be applied to different loss functions and different regularizers (Shalev-Shwartz and Zhang, 2013b, 2014). We discuss a few examples below.

Other scalar losses: The method can be applied to other convex loss functions, e.g. the squared loss, for which SDCA updates are also computed in closed form. If there is no closed form solution (e.g. there is none for the logistic loss), then $\Delta\alpha_i$ can be computed via a few iterations of the Newton method.

Other regularizers: Another straightforward generalization would be the introduction of ℓ_1/ℓ_2 regularization, also known as the elastic net (Zou and Hastie, 2005). That would require keeping a copy of the primal variable and performing ℓ_1-shrinkage after every update.

Structured losses: Finally, SDCA can be applied to structured loss functions, such as the ones used in Multiclass SVM of Crammer and Singer, (2001) and in Latent SVM of Felzenszwalb et al., (2008).

Method	Dataset	$n_{\text{class}} = 5$	$n_{\text{class}} = 10$	$n_{\text{class}} = 20$	$n_{\text{class}} = 50$	$n_{\text{class}} = 100$
Kang et al.	USPS					**91.6 (0.3)**
STL-SDCA		69.4 (0.6)	76.3 (1.1)	83.7 (0.2)	88.5 (0.5)	90.8 (0.3)
MTL-SDCA		**71.4 (0.7)**	**77.2 (0.5)**	**84.6 (0.4)**	**90.0 (0.5)**	90.6 (0.2)
Kang et al.	MNIST					84.8 (0.3)
STL-SDCA		65.6 (0.7)	73.6 (0.8)	**79.8 (1.0)**	83.1 (0.6)	85.7 (0.4)
MTL-SDCA		**66.2 (0.7)**	**74.0 (1.0)**	79.7 (0.9)	**83.4 (0.6)**	**86.0 (0.2)**

Table 4.1.: Mean accuracy across 5 splits on two handwritten digit recognition datasets (numbers in parenthesis show standard deviation scaled by $1/\sqrt{5}$, as reported in Kang et al., 2011), n_{class} indicates the number of training examples per class. Original images were preprocessed with PCA reducing dimensionality to $d = 87$ (USPS) and $d = 64$ (MNIST) retaining 95% of the variance.

4.3 Handwritten Digit Recognition

Before delving into scene classification on a challenging SUN 397 benchmark, we begin with a first set of experiments on two handwritten digit recognition datasets, where direct comparison to other multitask learning methods is readily available. Our algorithm is compared against two baselines:

Kang et al.: multitask feature learning method of Kang et al., (2011), which recently outperformed established multitask methods on the same data;

STL-SDCA: single task learning approach based on OVA SVM.

The main goal of these experiments is twofold: (i) compare the proposed approach to a state of the art multitask learning algorithm, and (ii) experimentally verify that classification using the shared representation $z = U^{\top}x$ is superior to single task learning in the original feature space.

We use two handwritten digit recognition datasets that are based on the subsets of USPS and MNIST. The data is provided by Kang et al., (2011), and we follow their evaluation protocol: parameters are tuned on a validation set, which is not used for training, and performance is evaluated on a fixed test set of 500 examples. Training and validation subsets are sampled randomly 5 times from a fixed set of 1500 examples. Experimental results are reported in Table 4.1.

On the USPS dataset (upper part of Table 4.1), single task and multitask learning algorithms perform on par when we use 100 training examples per class, and Kang et al. outperforms our methods in this case. When the amount of training data is successively reduced from 100, over 50, 20, 10, to 5 examples, the performance of STL-SDCA, as well as MTL-SDCA, decreases as expected. A similar trend is observed on the MNIST dataset, where both STL-SDCA and MTL-SDCA outperform Kang et al. with $n_{\text{class}} = 100$. However, the advantage of multitask over single-task learning becomes apparent as MTL-SDCA consistently outperforms STL-SDCA in every setting, with a single exception at $n_{\text{class}} = 20$ on MNIST data. As expected, multitask learning is particularly helpful when the amount of training data is extremely limited, e.g. we observe strong improvements with $n_{\text{class}} = 5$ and 10 training examples per class.

Discussion. Experimental results on the two small scale datasets suggest that our approach is competitive with the state of the art multitask learning method of Kang et al., (2011). The benefit of multitask learning is more pronounced on smaller training sets, which agrees both with the general intuition behind multitask learning and the related theoretical results of Maurer et al., (2013).

4.4 Scene Classification on SUN 397

In this section, we report our main experimental results on the challenging SUN 397 benchmark (Xiao et al., 2010), where the task is to classify real world images into one of the 397 scene categories that cover both indoor and outdoor sites. Our multitask learning method consistently outperforms single-task learning baselines and advances the current state of the art by over 2% in accuracy. The structure of this section is summarized below.

- In § 4.4.1, we provide details about the SUN dataset and describe the general experimental setup.
- In § 4.4.2, we concentrate on the feature extraction pipeline for images and investigate the effect of various engineering decisions, such as PCA preprocessing, normalization, and application of nonlinear feature maps.
- In § 4.4.3, we establish a strong baseline for further comparison by reproducing the state of the art results of Sánchez et al., (2013).
- In § 4.4.4, we provide an in-depth evaluation and analysis of our MTL-SDCA method. We compare against the single-task learning baselines, as well as against a two-layer architecture mentioned in § 4.2.1. We also highlight the advantage of multitask learning as measured by top-k accuracy, where k guesses are allowed, and discuss the runtime analysis of our pipeline.

4.4.1 Experimental Setup

We follow the evaluation protocol proposed by Xiao et al., (2010) in all experiments: we use $n_{\text{class}} \in \{5, 10, 20, 50\}$ images per class for training and $n_{\text{class}} = 50$ images per class for testing. We use the 10 splits provided on the website of the dataset[2] and report mean accuracy and standard deviation over the 10 splits. We consider every training subset in each split as an independent dataset and run the whole experimental pipeline – including the feature extraction, codebook learning, and model selection – on each of them separately.

Our feature extraction pipeline follows closely the one described by Sánchez et al., (2013): images are resized to 100K pixels, if larger, and approximately 10K

[2] http://people.csail.mit.edu/jxiao/SUN

descriptors are extracted per image from 24 × 24 patches on a regular grid every 4 pixels at 5 scales $2^{-2:.5:0}$. We use 128-dim SIFT descriptors of Lowe, (2004) and 96-dim local color statistic (LCS) descriptors of Clinchant et al., (2007).

The descriptors are processed by PCA as discussed below and we use on the order of 10^6 descriptors to learn the PCA projections. Finally, descriptors are encoded using the Fisher Vector (FV) encoding and pooled over a spatial pyramid with 4 regions (the entire image and three horizontal stripes). The codebook for FV is given by a GMM with 256 Gaussians, which is learned using the EM algorithm. This yields the following feature dimensions of the final descriptor: $d = 131,072$ (SIFT) and $d = 262,144$ (SIFT+LCS).

We use the VLFeat library of Vedaldi and Fulkerson, (2008) for feature extraction and FV encoding. To facilitate reproduction of our results, we publish all the necessary code[3], including the solvers for STL-SDCA and MTL-SDCA that are implemented in C++ and have an interface to Matlab.

4.4.2 Feature Engineering

				$n_{\text{class}} = 5$			$n_{\text{class}} = 10$			$n_{\text{class}} = 20$			$n_{\text{class}} = 50$		
LCS	PN	L2	PCA	Lin	Sqr	Chi	Lin	Sqr	Chi	Lin	Sqr	Chi	Lin	Sqr	Chi
			64	18.5	20.6	20.8	26.0	28.8	28.8	33.6	35.8	36.0	43.2	45.1	45.7
•			64	18.6	20.8	20.8	27.0	29.3	29.1	35.2	37.6	37.5	45.0	47.2	47.2
•	•		64	18.6	20.5	20.6	27.2	29.2	29.3	35.3	37.3	37.4	45.0	47.4	47.3
•		•	64	18.2	19.4	19.5	26.4	28.8	28.7	36.7	**39.2**	39.1	44.0	45.9	46.0
•	•	•	64	18.5	20.4	20.3	26.7	29.1	29.2	34.1	36.8	36.7	44.4	46.1	46.1
•			128	19.0	21.4	21.4	26.9	29.6	29.8	35.8	39.1	39.0	44.7	47.4	47.6
•	•		128	18.6	21.0	21.1	26.8	29.5	29.5	35.3	38.3	38.0	44.3	47.0	47.2
•		•	128	19.5	21.8	21.8	28.0	30.6	30.8	35.9	38.2	38.2	45.8	48.0	48.3
•	•	•	128	20.0	22.3	**22.5**	28.5	**31.2**	**31.2**	36.1	38.6	38.6	46.2	48.3	**48.4**

Table 4.2.: STL-SDCA accuracy on the first split of SUN 397 (Xiao et al., 2010). See §§ 4.4.1 and 4.4.2 for further details. **LCS:** local color statistic descriptor; **PN:** power normalization; **L2:** ℓ_2-normalization; **PCA:** independent PCA of SIFT and LCS to 64-dim each vs. joint PCA to 128-dim; **Lin/Sqr/Chi:** linear/Hellinger/χ^2 kernel; $\mathbf{n}_{\text{class}}$**:** the number of training examples per class.

In this section, we explore the impact of several implementation details on the final classification performance. Sánchez et al., (2013) provide an extensive evaluation of the effects of PCA, ℓ_2-normalization, power normalization ($z \leftarrow \mathbf{sign}(z)\,|z|^\rho$, $0 < \rho \leq 1$), and other parameters on the PASCAL VOC 2007 dataset. While their findings suggest that these details have significant effect on the final performance, a similar evaluation was not done on SUN 397. It is not clear which combination

[3] https://github.com/mlapin/cvpr14mtl

of the engineering decisions performs best, in particular for the LCS descriptor. We aim to fill this gap in this section.

To save computation time and avoid overfitting to other splits, we perform all experiments in this section on the first split only. We set the SVM parameter $C = (1/\lambda n)$ by 2-fold cross-validation and retrain models with the best parameter on the full training subsets. Our results are summarized in Table 4.2.

Impact of PCA. When both SIFT and LCS descriptors are used, there are two ways to perform PCA preprocessing: (i) reduce each descriptor to 64 dimensions independently and then concatenate, (ii) perform PCA on the combined descriptor reducing it to 128 dimensions. We observe that performing PCA on the combined descriptor is generally better and we use this strategy in our further experiments.

Impact of power normalization. Power normalization, also called *square rooting*, can be motivated in a number of ways (Sánchez et al., 2013). For example, it can be interpreted as a variance stabilizing transform (Jegou et al., 2012), and is often used with the BOW model (Vedaldi and Zisserman, 2012). In our experiments, we observe that performing power normalization with $\rho = 0.5$ on the LCS descriptor improves the performance when it is combined with ℓ_2-normalization and joint PCA. This setting yields the best accuracy in our experiments.

Impact of ℓ_2-normalization. The results for ℓ_2-normalization seem to depend on the way PCA preprocessing is done and generally improve the performance when dimensionality reduction is performed jointly.

Impact of the kernel map. We compare three SVM kernels: linear, Hellinger, and the χ^2 kernel. The Hellinger kernel in our setting is equivalent to performing power normalization with $\rho = 0.5$ on the combined SIFT+LCS feature vector. We observe that the Hellinger kernel performs better than the linear one and is comparable to the χ^2 kernel at significantly lower computational cost. Therefore, we avoid the χ^2 kernel in further experiments.

4.4.3 Baseline Methods

In this section, we mainly pursue two goals: (i) establish a strong single task learning baseline by reproducing the results of Sánchez et al., (2013), which is the current state of the art method on the SUN 397 benchmark; (ii) show that we can achieve further performance improvements using the feature tuning techniques discussed in § 4.4.2. Our experimental results, which we discuss next, are given in Table 4.3. As before, the SVM parameter C for single task learning is selected by 2-fold cross-validation and the final model is retrained on the full training subset. The results are obtained using the Hellinger kernel, PCA on the combined SIFT and LCS descriptor, with power- and ℓ_2-normalization on the LCS feature.

We make a few interesting observations. First, we confirm that the FV encoding exhibits striking performance even when only a single type of descriptor (SIFT) is used to represent images. For example, consider the last column in Table 4.3 that corresponds to $n_{\text{class}} = 50$. Our single task learning baseline, STL-SDCA, yields an average of 45.1% accuracy across 10 splits using only the SIFT descriptor,

Method	Features	$n_{\text{class}} = 5$	$n_{\text{class}} = 10$	$n_{\text{class}} = 20$	$n_{\text{class}} = 50$
Xiao et al., (2010)	12 combined	14.5	20.9	28.1	38.0
Su and Jurie, (2012)	Context+Semantic				35.6 (0.4)
Donahue et al., (2013)	DeCAF$_6$				40.9 (0.3)
Sánchez et al., (2013)	SIFT	19.2 (0.4)	26.6 (0.4)	34.2 (0.3)	43.3 (0.2)
STL-SDCA, Lin	SIFT	17.4 (1.5)	25.8 (0.2)	33.6 (0.3)	43.2 (0.2)
STL-SDCA, Sqr	SIFT	20.4 (0.3)	28.2 (0.3)	35.9 (0.3)	45.1 (0.3)
STL-SDCA-Stacked, Sqr	SIFT	20.6 (0.4)	28.4 (0.3)	36.1 (0.3)	45.3 (0.3)
MTL-SDCA, Sqr	**SIFT**	**20.8 (0.4)**	**28.9 (0.4)**	**37.6 (0.3)**	**46.9 (0.3)**
Sánchez et al., (2013)	SIFT+LCS	21.1 (0.3)	29.1 (0.3)	37.4 (0.3)	47.2 (0.2)
STL-SDCA, Sqr	SIFT+LCS	21.0 (0.5)	29.2 (0.3)	37.8 (0.6)	47.2 (0.4)
STL-SDCA-Stacked, Sqr	SIFT+LCS	21.1 (0.4)	29.3 (0.3)	37.9 (0.6)	47.3 (0.4)
MTL-SDCA, Sqr	**SIFT+LCS**	**21.2 (0.2)**	**29.4 (0.4)**	**38.5 (0.5)**	**47.9 (0.5)**
STL-SDCA, Sqr	SIFT+LCS+PN	20.4 (0.6)	29.0 (0.4)	37.4 (0.4)	47.1 (0.3)
STL-SDCA-Stacked, Sqr	SIFT+LCS+PN	20.8 (0.3)	29.1 (0.4)	37.5 (0.4)	47.2 (0.4)
MTL-SDCA, Sqr	**SIFT+LCS+PN**	**20.9 (0.4)**	**29.2 (0.4)**	**38.2 (0.4)**	**48.1 (0.4)**
STL-SDCA, Sqr	SIFT+LCS+L2	21.4 (0.4)	29.8 (0.5)	38.2 (0.4)	47.9 (0.3)
STL-SDCA-Stacked, Sqr	SIFT+LCS+L2	21.6 (0.3)	30.0 (0.5)	38.3 (0.4)	48.0 (0.4)
MTL-SDCA, Sqr	**SIFT+LCS+L2**	**21.7 (0.3)**	**30.3 (0.5)**	**39.0 (0.4)**	**49.0 (0.5)**
STL-SDCA, Sqr	SIFT+LCS+PN+L2	22.1 (0.6)	30.5 (0.6)	38.8 (0.3)	48.4 (0.2)
STL-SDCA-Stacked, Sqr	SIFT+LCS+PN+L2	22.3 (0.6)	30.7 (0.6)	38.9 (0.3)	48.5 (0.2)
MTL-SDCA, Sqr	**SIFT+LCS+PN+L2**	**22.4 (0.5)**	**31.0 (0.7)**	**39.5 (0.3)**	**49.5 (0.3)**

Table 4.3.: Mean accuracy and standard deviation across 10 splits on SUN 397. **STL-SDCA:** single task learning (OVA SVM); **STL-SDCA-Stacked:** two layer architecture described in § 4.4.4; **MTL-SDCA:** our proposed multitask learning method; **Lin/Sqr/Chi:** linear/Hellinger/χ^2 kernel; **LCS:** local color statistic descriptor; **PN:** power normalization; **L2:** ℓ_2-normalization; $\mathbf{n}_{\text{class}}$**:** the number of training examples per class.

and is further improved to 48.4% when color information is added, see the results with LCS+PN+L2. Similar improvements are also obtained with fewer training examples, $n_{\text{class}} = 5$, 10, and 20.

The STL-SDCA baseline outperforms the best published result of Sánchez et al., (2013), who obtained 43.3% accuracy using SIFT only and 47.2% accuracy using SIFT+LCS. It also exceeds the initial results published by the authors of the dataset (Xiao et al., (2010) reports 38.0% accuracy), as well as the more recent work (Donahue et al., 2013; Su and Jurie, 2012) that reports 35.6% and 40.9% respectively. We note that the DeCAF features used by Donahue et al., (2013) were learned on ImageNet data which may explain why a deep ConvNet is outperformed in our experiments.

The question we ask next is whether a shared low dimensional representation can exploit commonalities across scene classes to further improve the performance.

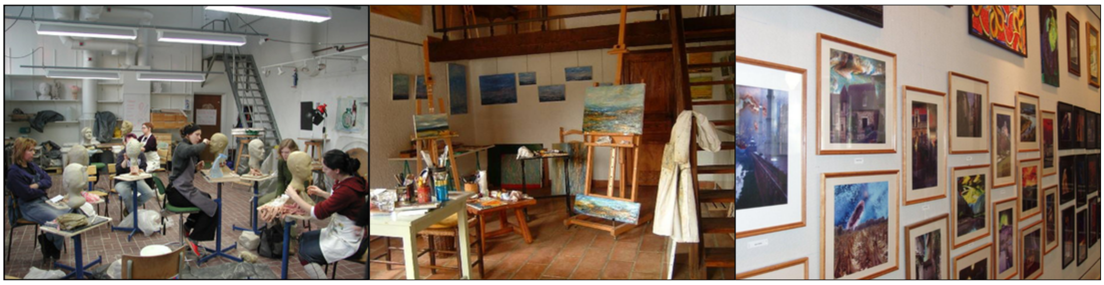


Figure 4.1.: Illustration of class ambiguity on SUN 397 (Xiao et al., 2010). **Labels (left to right):** *art school*, *art studio*, *art gallery*. Visual differences between these classes are rather subtle and are likely to be dominated by non-discriminative information in the high dimensional image descriptor.

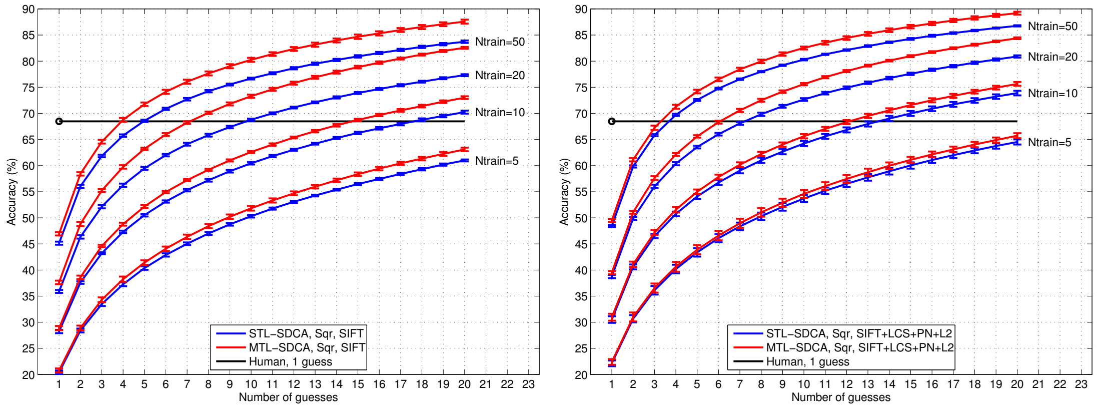
4.4.4 Multitask Learning

Apart from the obvious scalability issues, the downside of having a dense high dimensional image descriptor is that it also captures significant amounts of noise irrelevant to the given object category. When the number of training examples is small, it is difficult to identify features that generalize well and separate them from noise. The situation becomes even worse when there are highly related tasks that are trained using the one-vs-all approach.

The nature of the SUN dataset is such that there are intrinsically related classes that have very similar visual appearance. For example, consider the illustration given in Figure 4.1. There are three different categories related to art: "art school", "art studio", and "art gallery". Visual differences between these classes are rather subtle and are likely to be dominated by non-discriminative information in the high dimensional image descriptor. A classifier trained using the one-vs-all technique is likely to pick a random subset of features that just happen to discriminate between these related classes on few examples and will not generalize well. Our multitask learning approach, on the other hand, addresses this issue by forcing all classifiers to first agree on a significantly lower dimensional subspace of features and only then attempt to discriminate between the classes.

One natural baseline for comparison in this case is a two layer feed-forward architecture where the outputs of SVMs from the first layer are used as features (inputs) to the SVMs in the second layer. We refer to this approach as STL-SDCA-Stacked. Note that the matrix of the first layer predictors in this case is fixed and the resulting subspace cannot be influenced by the second layer predictors. On the contrary, our MTL method allows the matrix U to be iteratively updated thus propagating information from the second layer back to the first layer.

We tune the regularization parameters for STL-SDCA-Stacked and MTL-SDCA on the first split of the SUN 397 benchmark and then keep them fixed on the other 9 splits. Our results are reported in Table 4.3. Looking at the STL-SDCA-Stacked performance, it is evident that the improvement over the single task learning approach is minor (on the order of .1%–.2%), yet consistent. That gives us hope that there are inter-class correlations that could be exploited, even though the stacked architecture with a fixed first layer may be suboptimal in this case.



(a) SIFT

(b) SIFT + Color

Figure 4.2.: Mean top-k accuracy and standard deviation across 10 splits on the SUN 397 dataset. The number of guesses k is varied between 1 and 20. **STL-SDCA:** single task learning (OVA SVM); **MTL-SDCA:** multitask learning method described in Algorithm 4.1; **Human:** estimated top-1 human accuracy from AMT workers. **LCS:** local color statistic descriptor; **PN:** power normalization; **L2:** ℓ_2-normalization; **Sqr:** Hellinger kernel; **Ntrain:** the number of training examples per class.

Let us now discuss the results of our multitask learning approach, MTL-SDCA.

Top-1 accuracy. Results in Table 4.3 clearly indicate superiority of a learned representation that is shared across multiple classes. MTL-SDCA is consistently better for every training subset and all choices of image descriptors. Furthermore, the improvement is more significant when using the multitask learning approach compared to the stacked single task learning method.

Take for example the performance for $n_{\text{class}} = 50$. MTL-SDCA achieves 46.9% accuracy using only the SIFT descriptor and 49.5% using SIFT with LCS+PN+L2. That is better than the best published results as well as our strong baselines reported above. While the improvement is not overwhelming (correspondingly, 1.6% and 1% when compared to the stacked classifier), it is consistent across all settings: using SIFT vs. SIFT+LCS, and training with different amounts of data.

Top-k accuracy. Because there are intrinsically ambiguous classes like the art scenes mentioned above, or a factory and assembly line scenes, or different types of shops, we believe that the top-1 accuracy is a suboptimal performance measure on this dataset. We thus extend our evaluation and report mean top-k accuracy for each $k = 1, \dots, 20$ in Figure 4.2. The *top-k accuracy* assumes that the method outputs a set of k labels and counts the prediction as correct if the ground truth label is included in that set. As our methods produce a ranking of labels, we obtain the top k labels by sorting the prediction scores in descending order.

Again, we observe that MTL-SDCA consistently improves the performance not only for every image descriptor and every training subset, but also for every number of allowed guesses k. Moreover, the improvement is more significant for $k \geq 3$.

For example, using SIFT only and $n_{\text{class}} = 20$ examples per class, MTL-SDCA improves the top-5 accuracy by 3.7% and top-15 accuracy by 5%.

Finally, we explore how multitask learning on this dataset compares to the human performance, which is estimated based on the confusion matrix of "good" AMT[4] workers provided by Xiao et al., (2010). The estimated top-1 accuracy of humans is 68.48%, and we observe that already 3–4 guesses are generally sufficient to reach human performance on that data.

Runtime analysis. The overhead of multitask learning is relatively small (approximately a factor of 4) if the cost for computing the kernel matrices is taken into account, and is close to negligible (6%–12%) when complete image classification pipeline is considered, since most of the time is spent on computation of image descriptors. Further details can be found in § B.1 (page 187).

4.5 Conclusion

In this chapter, we considered the problem of learning in the regime of limited training data from the perspective of *multitask learning*. Specifically, we proposed a multitask representation learning scheme that jointly learns a set of linear classifiers and a shared low dimensional representation. Our method employs dual optimization and scales to high dimensional dense image descriptors, such as the ones based on the FV encoding.

The principle idea and the main motivation of our method is that a shared representation allows to leverage task relatedness, which is ubiquitous in computer vision. The running example of this chapter is scene classification, where different types of scenes, e.g. art schools and art studios, share common elements of visual appearance. Therefore, joint learning of a shared representation seems like a reasonable step towards training the classifiers from relatively small samples.

Our multitask approach outperforms the state of the art on SUN 397 and consistently improves classification performance over the respective single task baselines. Moreover, the improvement is particularly evident in *top-k accuracy* for $k > 1$, which we interpret as the ability of multitask learning to discover groups of related classes. Motivated by that observation, as well as by the fact that certain scene categories are inherently ambiguous or *multilabel* in nature, we take a closer look at top-k performance optimization in the following part of this thesis.

[4] AMT – Amazon Mechanical Turk.

✦

This chapter concludes our exploration of learning with limited training data. We have considered two important frameworks, *learning using privileged information* (LUPI) in Chapter 3, and *multitask learning* (MTL) in this chapter. Both methodologies facilitate introduction of *prior knowledge* into the learning problem, but do so in different ways. The LUPI paradigm asserts the existence of privileged features that are related to the learning problem and can be used during training. Such features could come from a teacher that guides the learning process. Alternatively, the prior knowledge may be provided in the form of instance weights that describe the relative importance or hardness of the training examples.

The MTL framework explores an orthogonal direction. Here, the motivation is to increase the efficiency of existing training examples by exploiting task relatedness. We have considered a basic formulation where there is no prior knowledge about the exact groups of related tasks, although that knowledge could also be included, e.g. using structured sparsity-inducing norms (Jenatton et al., 2011a). The latter hints at the important role that regularizers play in the learning problem – a direction that we explored independently of this thesis in the context of *output kernel learning* with multiple tasks in (Jawanpuria et al., 2015).

In the following, we pivot to focus on the analysis and optimization of loss functions. In particular, we consider top-k error minimization and scalability to large data sets, as motivated by recent advances in computer vision.

Part II

Learning with Class Ambiguity

Collecting high quality ground truth annotation in modern large scale datasets requires significant effort and is not always feasible. Trading off quality and rigor for scale leads to a new challenge that we seek to address in this part.

- In Chapter 5, we discuss the problem of *class ambiguity* that arises in large scale datasets, and recognize the top-k error as an appropriate target performance measure. We propose top-k multiclass SVM as a suitable learning algorithm for the top-k objective, and discuss an efficient algorithm for the Euclidean projection onto the top-k simplex. The latter enables top-k SVM optimization within the SDCA framework.
- In Chapter 6, we extend our analysis of class ambiguity and top-k error optimization along multiple directions. In particular, we introduce smooth top-k SVM and top-k extensions of the softmax loss, analyze top-k calibration of multiclass methods, consider the transition from multiclass to multilabel learning, and propose smooth multilabel SVM. We discuss SDCA optimization of the considered methods, contribute novel projection algorithms, and perform an extensive empirical evaluation on multiclass and multilabel datasets which leads to interesting insights.

5 Top-k Multiclass SVM

Chapters 3 and 4 were primarily concerned with learning in the regime of limited training data. The difficulty of collecting the ground truth annotation motivated us to consider the LUPI and MTL learning frameworks which improved the performance while using just 5 to 50 training examples per class. In this chapter, we tackle a different challenge that exists in modern *large scale* image classification. Recent datasets have 200 to 1000 image categories and millions of training examples, which is a significant advancement compared to the older benchmarks. However, another issue related to the ground truth annotation arises.

Image classification on large scale is naturally susceptible to **class ambiguity**. As the number of classes increases, they become more fine-grained and less easy to discriminate; they may overlap or exhibit a hierarchical structure. Furthermore, most real-world images are multilabel in nature as they usually depict multiple objects or visual categories. These are inherent limitations of single label annotation, where every image is tagged with a single class label.

One way to address class ambiguity is to improve the annotation by collecting a complete and consistent list of labels for every training example. However attractive, that approach is not cost-effective on large scale. Instead, we set to explore ways that improve learning with class ambiguity using single label annotation. We recognize that the **top-k error** is a better performance metric in that case, as it allows k attempts to guess the ground truth label. We propose *top-k multiclass SVM* as a direct method to optimize for top-k performance, and a *fast optimization* scheme based on the stochastic dual coordinate ascent (SDCA) framework of Shalev-Shwartz and Zhang, (2013b). The key component of our optimization scheme is an efficient algorithm to compute projections onto what we call the *top-k simplex*, which is of its own interest. Our experiments show consistent improvements in top-k accuracy compared to various baseline methods.

The material in this chapter is based on the following publication:

- M. Lapin, M. Hein, and B. Schiele (2015). "Top-k Multiclass SVM." in: *Advances in Neural Information Processing Systems 29 (NIPS).*

5.1 Introduction

As the number of classes increases, two important issues emerge: class overlap and multilabel nature of examples (Gupta et al., 2014). That phenomenon asks for adjustment of both the evaluation metrics as well as the loss functions employed. When a predictor is allowed k guesses and is not penalized for $k-1$ mistakes, such an evaluation measure is known as the *top-k error*. It is an important metric that is particularly well suited to performance evaluation with class ambiguity, as the illustration in Figure 5.1 indicates.

How obvious is it that each row of Figure 5.1 shows examples of *different* classes? Can we imagine a human to predict the ground truth label correctly on the first attempt? Does it even make sense to penalize a learning system for predicting the label “River” instead of “Park” for the first image? While the problem of class ambiguity is apparent in computer vision, similar problems arise in other domains when the number of classes becomes large.

To address that problem, we propose *top-k multiclass SVM* as a generalization of the well-known multiclass SVM of Crammer and Singer, (2001). Our proposed loss function, which we call the **top-k hinge loss**, is based on a tight convex upper bound of the discrete top-k error. A closely related loss function can be found in the general family of ranking based losses that was recently proposed by Usunier et al., (2009). We show that our top-k hinge loss is a lower bound on their version and is thus a tighter bound on the top-k error. As both loss functions are intended to optimize the top-k error, we refer to them as the *top-k hinge loss*, and use the suffixes α and β to differentiate between our and their versions correspondingly.

To facilitate classifier training with the two loss functions, we propose an efficient optimization scheme based on the stochastic dual coordinate ascent (SDCA) framework of Shalev-Shwartz and Zhang, 2013b. A key ingredient in our optimization method is an algorithm to efficiently compute projections onto what we call the *top-k simplex*. That projection turns out to be an interesting generalization of the continuous quadratic knapsack problem and the Euclidean projection onto the standard simplex. The proposed algorithm computes the projection of a point $x \in \mathbb{R}^d$ in time $O(d \log d + kd)$ using a procedure based on sorting.

Our top-k multiclass SVM, or simply *top-k SVM*, scales to large datasets like Places 205 (Zhou et al., 2014) and ImageNet 2012 (Russakovsky et al., 2015) featuring 200 to 1000 image categories and millions of training examples. An extensive experimental evaluation shows that top-k SVM consistently improves the performance in top-k error over the baseline multiclass SVM, which is equivalent to our top-1 SVM, as well as over the one-vs-all (OVA) SVM, and other methods based on ranking losses: SVM$^{\text{Perf}}$ of Joachims, (2005), TopPush of Li et al., (2014a), and Wsabie^{++} of Gupta et al., (2014).

5.2 Top-k Multiclass SVM

In this section, we consider the general multiclass classification problem, introduce our top-k SVM, compute the convex conjugate losses for the optimization algorithm, and finally discuss the related methods. First, we introduce our notation and formally define the top-k error, and then we proceed as follows.

- In § 5.2.1, we recall the classical multiclass SVM of Crammer and Singer, (2001), the corresponding (primal) loss function and its convex conjugate, which is used in the optimization method.

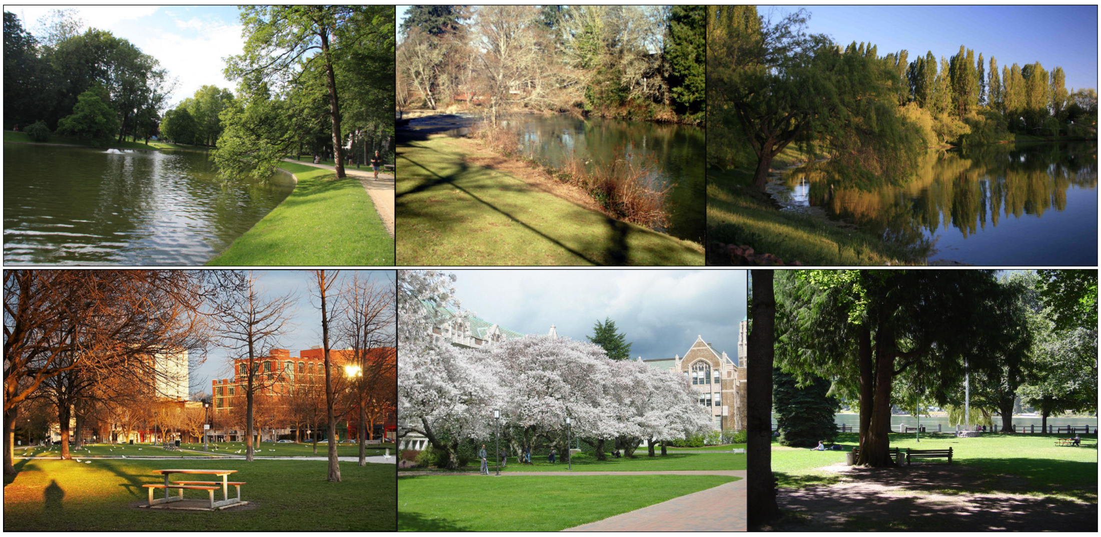


Figure 5.1.: Class ambiguity in the SUN 397 benchmark dataset (Xiao et al., 2010). **Top:** Park, River, Pond. **Bottom:** Park, Campus, Picnic area.

- In § 5.2.2, we introduce our novel top-k SVM and the corresponding top-k hinge loss. We also compute its conjugate and define the *top-k simplex* as the effective domain of the conjugate loss.
- In § 5.2.3, we elaborate on the connection between our top-k hinge loss (α) and the top-k hinge loss (β) from the family of losses by Usunier et al., (2009). We also discuss a reduction scheme that converts multiclass classification into binary classification, and which is different from the standard OVA approach. We use that scheme in our experiments with the SVM$^{\text{Perf}}$ of Joachims, (2005).

Notation. Let $S = \{(x_i, y_i) \,|\, i = 1, \ldots, n\}$ be a set of n training examples $x_i \in \mathcal{X}$ along with the corresponding labels $y_i \in \mathcal{Y}$, let $\mathcal{X} = \mathbb{R}^d$ be the feature space, and $\mathcal{Y} = \{1, \ldots, m\}$ the set of labels. The task is to learn a set of m linear predictors $w_y \in \mathbb{R}^d$ such that the expected loss of the classifier $\hat{f}(x) = \arg\max_{y \in \mathcal{Y}} \langle w_y, x \rangle$ is minimized for a given loss function L. The loss L is usually chosen to be a convex upper bound on a discrete performance metric, such as the misclassification error (a.k.a. the zero-one loss), and the top-k error. Although we mainly consider linear classifiers of the form $f(x) = (\langle w_y, x \rangle)_{y \in \mathcal{Y}}$, the approach that we discuss is general and can be extended to nonlinear classifiers using kernels.

Classification is challenging in the presence of a large number of ambiguous classes. As the scale of the problem increases, the standard zero-one error becomes excessively stringent, and it is natural, therefore, to extend the evaluation protocol to allow k guesses instead of one. That leads us to the *top-k error* and the *top-k accuracy* performance measures, which are well recognized in the computer vision community following the popular ImageNet benchmark (Russakovsky et al., 2015).

Consider a ranking of labels induced by the prediction scores $\langle w_y, x \rangle$. Let π be a permutation such that π_j is the index of the jth largest score, i.e.

$$\langle w_{\pi_1}, x \rangle \geq \langle w_{\pi_2}, x \rangle \geq \ldots \geq \langle w_{\pi_m}, x \rangle.$$

The **top-k error**, denoted err_k, is defined as

$$\mathrm{err}_k(f(x), y) = [\![\langle w_{\pi_k}, x\rangle > \langle w_y, x\rangle]\!],$$

where $f(x) = (\langle w_1, x\rangle, \ldots, \langle w_m, x\rangle)^\top$, and $[\![P]\!] = 1$ if P is true and 0 otherwise. Note that the standard zero-one loss is recovered when $k = 1$, and $\mathrm{err}_k(f(x), y)$ is always 0 for $k = m$. Therefore, we are interested in the regime $1 \leq k < m$.

5.2.1 Multiclass Support Vector Machine

In this section, we review the multiclass SVM of Crammer and Singer, (2001) which will be the basis for the top-k multiclass SVM in the following.

The **multiclass hinge loss** on a training example (x_i, y_i) is defined as

$$L(y_i, f(x_i)) = \max_{y \in \mathcal{Y}} \left\{ [\![y \neq y_i]\!] + \langle w_y, x_i\rangle - \langle w_{y_i}, x_i\rangle \right\}, \tag{5.1}$$

where $[\![y \neq y_i]\!]$ plays the same role as the function $\Delta(y_i, y)$ in structured SVM (Nowozin and Lampert, 2011; Tsochantaridis et al., 2005) that measures a distance in label space between y_i and y.

Next, we compute the convex conjugate of the loss function (5.1) for the optimization scheme that is based on Fenchel duality (see § A.2.2, page 184). In the following, we adopt the notation of Shalev-Shwartz and Zhang, (2014): let $c \triangleq \mathbf{1} - e_{y_i}$, where $\mathbf{1}$ is the all ones vector and e_j is the jth standard basis vector in $\mathbb{R}^m$, let $a \in \mathbb{R}^m$ be defined componentwise as $a_j \triangleq \langle w_j, x_i\rangle - \langle w_{y_i}, x_i\rangle$, and let

$$\Delta \triangleq \{x \in \mathbb{R}^m \mid \langle \mathbf{1}, x\rangle \leq 1,\ 0 \leq x_i,\ i = 1, \ldots, m\}$$

be the unit simplex. Note that we can re-write the loss L in (5.1) equivalently as $\phi(a) = \max\{0, (a + c)_{\pi_1}\}$, where thresholding with 0 is actually redundant in this case as $(a + c)_{\pi_1} = \max_y (a + c)_y \geq (a + c)_{y_i} = 0$. It is introduced here to enhance similarity to the top-k version of the loss defined later.

Proposition 5.1 (Shalev-Shwartz and Zhang, 2014, § 5.1)**.** A primal-conjugate pair for the multiclass SVM loss (5.1) is

$$\phi(a) = \max\{0, (a + c)_{\pi_1}\}, \qquad \phi^*(b) = \begin{cases} -\langle c, b\rangle & \text{if } b \in \Delta, \\ +\infty & \text{otherwise.} \end{cases} \tag{5.2}$$

The original proof can be found in the respective paper; we follow a similar argumentation in the proof of Proposition 5.2 below.

5.2.2 Top-k Support Vector Machine

The main motivation for the top-k loss is to relax the penalty for making an error in the first $(k-1)$ predictions. Looking at the function ϕ in (5.2), a direct extension to the top-k setting would be

$$\psi_k(a) = \max\{0, (a+c)_{\pi_k}\},$$

which incurs a loss if and only if $(a+c)_{\pi_k} > 0$. Since the ground truth score $(a+c)_{y_i} = 0$, we conclude that

$$\psi_k(a) > 0 \iff \langle w_{\pi_1}, x_i \rangle \geq \ldots \geq \langle w_{\pi_k}, x_i \rangle > \langle w_{y_i}, x_i \rangle - 1,$$

which directly corresponds to the top-k error err_k, with a margin of 1. Note that the function ψ_k ignores the values of the first $(k-1)$ scores. Some of those top prediction scores could be quite large if there are very similar classes, and that would be fine in this model as long as the correct prediction is still within the first k guesses. However, the function ψ_k is unfortunately nonconvex as the function $f_k(x) = x_{\pi_k}$ returning the kth largest coordinate is nonconvex for $k \geq 2$. Therefore, finding a globally optimal solution is computationally intractable.

Instead, we propose the following convex upper bound on ψ_k, which we call the **top-k hinge loss (α)**:

$$\phi_k(a) = \max\left\{0, \frac{1}{k}\sum_{j=1}^{k}(a+c)_{\pi_j}\right\}, \tag{5.3}$$

where the sum of the k largest components is known to be convex (§ A.3, page 185). We have that

$$\psi_k(a) \leq \phi_k(a) \leq \phi_1(a) = \phi(a),$$

for any $k \geq 1$ and $a \in \mathbb{R}^m$. Moreover, $\phi_k(a) < \phi(a)$ unless all k largest scores are the same. This extra slack can be used to increase the margin between the current and the $(m-k)$ remaining least similar classes, which should then lead to an improvement in the top-k metric.

Top-k Simplex

Here, we define a set Δ_k that arises naturally as the effective domain[1] of the conjugate of the top-k hinge loss (5.3). By analogy, we call it the top-k simplex as for $k=1$ it reduces to the standard simplex with the inequality constraint, $\Delta = \{x \in \mathbb{R}^m \mid \langle \mathbf{1}, x \rangle \leq 1,\ x_i \geq 0,\ 1 \leq i \leq m\}$.

Definition 5.1. The *top-k simplex* is a convex polytope defined as

$$\Delta_k(r) \triangleq \{x \in \mathbb{R}^m \mid \langle \mathbf{1}, x \rangle \leq r,\ 0 \leq x_i \leq (1/k)\langle \mathbf{1}, x \rangle,\ 1 \leq i \leq m\},$$

[1] A convex function $f : X \to \mathbb{R} \cup \{\pm\infty\}$ has an *effective domain* $\mathbf{dom}\, f = \{x \in X \mid f(x) < +\infty\}$.

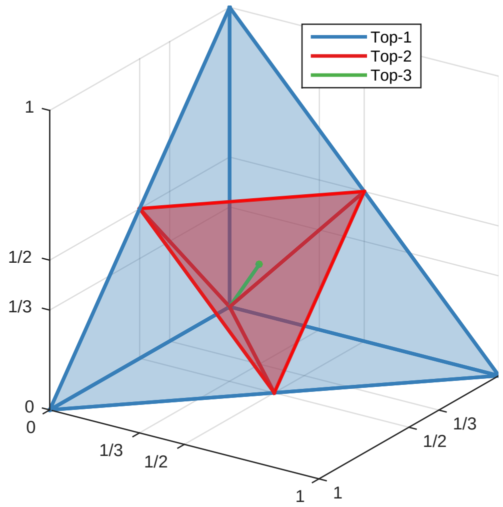


Figure 5.2.: Top-k simplex $\Delta_k \in \mathbb{R}^3$ for $k = 1, 2, 3$. Note that $\Delta_1 = \Delta$ and $\Delta_k \subsetneq \Delta$ for all $k > 1$. Moreover, the top-k simpex has $\binom{m}{k} + 1$ vertices in $\mathbb{R}^m$, as there can be at most k elements x_i for which $x_i = (1/k) \langle \mathbf{1}, x \rangle$.

where the *radius* $r \geq 0$ is the bound on the sum $\langle \mathbf{1}, x \rangle$. We let $\Delta_k \triangleq \Delta_k(1)$.

The crucial difference between the standard simplex and the top-k simplex is the upper bound on x_i, which now introduces a coupling between all the elements in x. The bound limits the maximal contribution of any x_i to the total sum $\langle \mathbf{1}, x \rangle$ thus reducing the set Δ_k compared to Δ, see Figure 5.2 for an illustration. As will become clear in the following, the top-k simplex is the feasible set for the dual variables of top-k SVM. Our intuition is that such a bound on x_i prevents the classifier to unequivocally commit to a single class (represented by a vertex of the standard simplex), and instead the classifier has to concentrate the weight on at least k classes (represented by a vertex of the top-k simplex).

Convex Conjugate of the Top-k Hinge Loss (α)

Next, we derive the convex conjugate of the proposed top-k hinge loss (5.3). We begin with a known result that is used later in the derivation of the conjugate.

Lemma 5.1 (Ogryczak and Tamir, 2003, Lemma 1)**.** For any $h \in \mathbb{R}^m$, $1 \leq k \leq m$,

$$\sum_{j=1}^{k} h_{\pi_j} = \min_{t \in \mathbb{R}} \Big\{ kt + \sum_{j=1}^{m} \max\{0, h_j - t\} \Big\},$$

where π reorders $(h_j)_{j=1}^m$ in nonincreasing order, i.e., $h_{\pi_1} \geq h_{\pi_2} \geq \ldots \geq h_{\pi_m}$.

Proof. For a $t_0 \in [h_{\pi_{k+1}}, h_{\pi_k}]$, we have

$$\begin{aligned}\min_t \Big\{kt + \sum_{j=1}^m \max\{0, h_j - t\}\Big\} &\leq kt_0 + \sum_{j=1}^m \max\{0, h_j - t_0\} \\ &= kt_0 + \sum_{j=1}^k \Big(h_{\pi_j} - t_0\Big) = \sum_{j=1}^k h_{\pi_j}.\end{aligned}$$

On the other hand, for any $t \in \mathbb{R}$, we have

$$\begin{aligned}\sum_{j=1}^k h_{\pi_j} = kt + \sum_{j=1}^k \Big(h_{\pi_j} - t\Big) &\leq kt + \sum_{j=1}^k \max\{0, h_{\pi_j} - t\} \\ &\leq kt + \sum_{j=1}^m \max\{0, h_j - t\}.\end{aligned}$$

□

Now, we are ready to compute the conjugate loss. The derivation follows the proof of Proposition 5.1 and relies on the Lagrangian duality (§ A.1, page 179).

Proposition 5.2. A primal-conjugate pair for the top-k hinge loss (5.3) is

$$\phi_k(a) = \max\Big\{0, \frac{1}{k}\sum_{j=1}^k (a+c)_{\pi_j}\Big\}, \qquad \phi_k^*(b) = \begin{cases} -\langle c, b\rangle & \text{if } b \in \Delta_k, \\ +\infty & \text{otherwise.}\end{cases} \tag{5.4}$$

Moreover, $\phi_k(a) = \max\{\langle a + c, \lambda\rangle \mid \lambda \in \Delta_k\}$.

Proof. We use Lemma 5.1 and write

$$\phi_k(a) = \min\Big\{s \mid s \geq t + \frac{1}{k}\sum_{j=1}^m \xi_j,\ s \geq 0,\ \xi_j \geq a_j + c_j - t,\ \xi_j \geq 0\Big\}.$$

The Lagrangian is given as

$$\begin{aligned}\mathcal{L}(s, t, \xi, \alpha, \beta, \lambda, \mu) = s + \alpha\Big(t + \frac{1}{k}\sum_{j=1}^m \xi_j - s\Big) - \beta s \\ + \sum_{j=1}^m \lambda_j\left(a_j + c_j - t - \xi_j\right) - \sum_{j=1}^m \mu_j \xi_j.\end{aligned}$$

Minimizing over (s, t, ξ), we get

$$\alpha + \beta = 1, \quad \alpha = \sum_{j=1}^m \lambda_j, \quad \lambda_j + \mu_j = \frac{1}{k}\alpha.$$

As $\beta \geq 0$ and $\mu_j \geq 0$, it follows that $\langle \mathbf{1}, \lambda \rangle \leq 1$ and $0 \leq \lambda_j \leq \frac{1}{k} \langle \mathbf{1}, \lambda \rangle$. Since the duality gap is zero, we finally get

$$\phi_k(a) = \max\{\langle a + c, \lambda \rangle \mid \lambda \in \Delta_k\}.$$

The conjugate $\phi_k^*(b)$ can now be computed as

$$\begin{aligned}\max_a \{\langle a, b \rangle - \phi_k(a)\} &= \max_a \min_{\lambda \in \Delta_k} \{\langle a, b \rangle - \langle a + c, \lambda \rangle\} \\ &= \min_{\lambda \in \Delta_k} \{- \langle c, \lambda \rangle + \max_a \langle a, b - \lambda \rangle\}.\end{aligned}$$

Since $\max_a \langle a, b - \lambda \rangle = \infty$ unless $b = \lambda$, we get $\phi_k^*(b)$ as in (5.4). □

We see that the proposed formulation (5.3) naturally extends the multiclass SVM of Crammer and Singer, (2001), which is recovered when $k = 1$. We have also obtained an interesting extension (or rather contraction, since $\Delta_k \subset \Delta$) of the standard simplex. The set Δ_k plays an important role in our optimization scheme as a feasible set for the dual variables of top-k SVM.

5.2.3 Ranking Based Losses

In this section, we discuss how the proposed top-k hinge loss (α) relates to existing ranking based losses as well as to the SVM$^{\text{Perf}}$ method of Joachims, (2005), which is designed to optimize multivariate performance measures. We introduce the top-k hinge loss (β), compute its conjugate, and define the top-k simplex (β). We also discuss a multiclass to binary reduction scheme that we use in the experiments with SVM$^{\text{Perf}}$, as well as the related precision@k, recall@k measures.

Recently, Usunier et al., (2009) formulated a general family of convex losses for ranking and multiclass classification. They proposed the *ordered weighted pairwise classification (OWPC)* loss, which in our notation can be written as

$$L_\omega(a) = \sum_{j=1}^{m} \omega_j \max\{0, (a + c)_{\pi_j}\},$$

where $\omega_1 \geq \ldots \geq \omega_m \geq 0$ is a nonincreasing sequence of nonnegative weights. The relation to the top-k hinge loss becomes apparent if we choose $\omega_j = \frac{1}{k}$ if $j \leq k$, and 0 otherwise. In that case, we obtain another version of the top-k hinge loss, which we call the **top-k hinge loss (β)**:

$$\tilde{\phi}_k\big(a\big) = \frac{1}{k} \sum_{j=1}^{k} \max\{0, (a + c)_{\pi_j}\}. \tag{5.5}$$

It is straightforward to check that

$$\psi_k(a) \leq \phi_k(a) \leq \tilde{\phi}_k(a) \leq \phi_1(a) = \tilde{\phi}_1(a) = \phi(a).$$

The bound $\phi_k(a) \leq \tilde{\phi}_k(a)$ holds with equality whenever there are no classification mistakes ($(a+c)_{\pi_1} = 0$) or when all top k guesses are wrong ($(a+c)_{\pi_k} \geq 0$). Otherwise, there is a gap between the α version $\phi_k(a)$ and the β version $\tilde{\phi}_k(a)$, and our top-k loss $\phi_k(a)$ is a strictly better upper bound on the discrete top-k error. We perform extensive evaluation and comparison of both losses in § 5.5.

To train a classifier with the OWPC loss, Usunier et al., (2009) used LaRank (Bordes et al., 2007), while Gupta et al., (2014) and Weston et al., (2011) optimized an *approximation* of $L_\omega(a)$. Instead, we show that the top-k hinge loss (5.5) can be directly and efficiently optimized within the SDCA framework without the need of an approximation.

Convex Conjugate of the Top-k Hinge Loss (β)

Here, we derive the convex conjugate of the top-k hinge loss (5.5), which will be used later in the optimization framework in § 5.3. We obtain a similar statement to that of Proposition 5.2, with the main difference being the effective domain of the conjugate loss. By analogy, we introduce the following set.

Definition 5.2. The *top-k simplex (β)* is a convex polytope defined as

$$\tilde{\Delta}_k(r) \triangleq \{x \in \mathbb{R}^m \mid \langle \mathbf{1}, x \rangle \leq r,\ 0 \leq x_i \leq r/k,\ 1 \leq i \leq m\},$$

where the *radius* $r \geq 0$ is the bound on the sum $\langle \mathbf{1}, x \rangle$. We let $\tilde{\Delta}_k \triangleq \tilde{\Delta}_k(1)$.

Note that unlike in the top-k simplex Δ_k, the upper bound on x_i is now a fixed constant r/k. This has two important implications:

- $\Delta_k(r) \subset \tilde{\Delta}_k(r)$ for all $r \geq 0$ and $1 \leq k \leq m$, which is directly related to the fact that $\phi_k(a) \leq \tilde{\phi}_k(a)$, i.e. the top-$k$ hinge loss (α) is a tighter bound on the top-k error than the top-k hinge loss (β);
- the Euclidean projection onto the set $\tilde{\Delta}_k(r)$ is easier to compute, as we discuss in § 5.4. This is useful for the computation of the dual loss, but not for the SDCA optimization itself, as we require a *biased* projection there. These details are covered in §§ 5.3 and 5.4.

We are now ready to formulate the convex conjugate of the top-k hinge loss (β).

Proposition 5.3. A primal-conjugate pair for the top-k hinge loss (5.5) is

$$\tilde{\phi}_k(a) = \frac{1}{k}\sum_{j=1}^{k} \max\left\{0, (a+c)_{\pi_j}\right\}, \qquad \tilde{\phi}_k^*(b) = \begin{cases} -\langle c, b \rangle & \text{if } b \in \tilde{\Delta}_k, \\ +\infty & \text{otherwise.} \end{cases} \tag{5.6}$$

Moreover, $\tilde{\phi}_k(a) = \max\{\langle a+c, \lambda \rangle \mid \lambda \in \tilde{\Delta}_k\}$.

Proof. The proof is similar to the proof of Proposition 5.2; the main step is

$$\begin{aligned}\tilde{\phi}_k(a) &= \min_{t,\xi,h}\left\{t + \tfrac{1}{k}\langle \mathbf{1}, \xi \rangle \mid \xi_j \geq h_j - t,\ \xi_j \geq 0,\ h_j \geq a_j + c_j,\ h_j \geq 0\right\} \\ &= \max_{\lambda}\left\{\langle a+c, \lambda \rangle \mid \langle \mathbf{1}, \lambda \rangle \leq 1,\ 0 \leq \lambda_j \leq \tfrac{1}{k},\ 1 \leq j \leq m\right\}.\end{aligned}$$

□

As before, we observe that the loss (5.5) naturally reduces to the multiclass hinge loss of Crammer and Singer, (2001) for $k = 1$. We also would like to highlight the relation $\Delta_k(r) \subset \tilde{\Delta}_k(r) \subset \Delta$, where the inclusion is proper for $k > 1$.

Multiclass to binary reduction

Next, we show that it is possible to compare top-k multiclass SVM to methods that solve a *binary* ranking problem, which is more prevalent in information retrieval. We use this reduction scheme in our experiments to compare top-k SVM with SVM$^{\text{Perf}}$ of Joachims, (2005) and TopPush of Li et al., (2014a). The trick that we use is to augment the training set by embedding each $x_i \in \mathbb{R}^d$ into $\mathbb{R}^{md}$ using a feature map Φ_y for each $y \in \mathcal{Y}$. The mapping Φ_y places x_i at the yth position in $\mathbb{R}^{md}$ and puts zeros everywhere else. The example $\Phi_{y_i}(x_i)$ is labeled $+1$ and all $\Phi_y(x_i)$ for $y \neq y_i$ are labeled -1. Therefore, we have a new training set with mn examples and md dimensional (sparse) features. Moreover, $\langle w, \Phi_y(x_i) \rangle = \langle w_y, x_i \rangle$ which establishes the relation to the original multiclass problem.

Precision@k and Recall@k

The structured SVM$^{\text{Perf}}$ of Joachims, (2005) optimizes general multivariate performance measures that are based on the confusion matrix. In particular, it can optimize a convex upper bound on the recall@k measure, which is defined as the recall, i.e. the fraction of all positive examples retrieved, of a classifier that predicts the top k examples as positive. If we consider the reduction scheme discussed above with m classes and n training examples, then we have a set of mn examples with binary labels where exactly n examples are positive and $(m-1)n$ are negative. Therefore, we have that recall@$n = 1$ if and only if every training example is classified correctly, i.e. the ground truth class is ranked above the $(m-1)$ remaining classes for every example, and thus $\text{err}_1 = 0$ for the original multiclass problem. Similarly, we conclude that for all $1 \leq k \leq m$, recall@$kn = 1$ if and only if $\text{err}_k = 0$. That motivated us to compare the proposed top-k SVM with SVM$^{\text{Perf}}$ optimizing recall@kn on the associated binary problem. We have also experimented in § 5.5.2 with the precision@kn metric, which computes the fraction of positive examples in the top kn results. For convenience, we use the notation Prec@k and Recall@k for precision@kn and recall@kn respectively.

Note, however, that the SVM$^{\text{Perf}}$ loss function is not directly comparable to our top-k hinge loss, as it is not decomposable into a sum of instance based losses. Moreover, while the method of Joachims, (2005) is theoretically elegant, the corresponding implementation did not scale to very large datasets.

5.3 Optimization Framework

In this section, we describe an optimization framework based on Fenchel duality for a general ℓ_2-regularized multiclass classification problem.

- In § 5.3.1, we define j-compatible loss functions and obtain primal and Fenchel dual optimization problems for them. In particular, our Theorem 5.1 allows one to quickly obtain dual problems for the multiclass SVM and the top-k SVM considered next.

- In § 5.3.2, we discuss specific instantiations with the multiclass SVM and the top-k SVM loss functions, and show how they can be optimized using the SDCA framework.

5.3.1 Fenchel Duality for ℓ_2-Regularized Multiclass Classification

In this section, we introduce the notion of a j-compatible function, formulate a dual optimization problem for such functions, and prove that the multiclass SVM loss as well as the proposed top-k hinge loss are j-compatible. The main technical results are based on Fenchel duality, which is covered in appendix § A.2 (page 183).

Let $X \in \mathbb{R}^{d \times n}$ be the matrix of training examples $x_i \in \mathbb{R}^d$, let $W \in \mathbb{R}^{d \times m}$ be the matrix of primal variables obtained by stacking the vectors $w_y \in \mathbb{R}^d$, and $A \in \mathbb{R}^{m \times n}$ the matrix of dual variables. Before we prove our main result of this section (Theorem 5.1), we first impose a technical constraint on a loss function to be compatible with the choice of the ground truth coordinate.

Definition 5.3. A convex function ϕ is *j-compatible* if for any $y \in \mathbb{R}^m$ with $y_j = 0$,

$$\sup\{\langle y, x\rangle - \phi(x) \,|\, x_j = 0\} = \phi^*(y).$$

This is a technical constraint that is needed to prove the equality in the following lemma. It applies to the multiclass SVM loss and the top-k hinge loss, as we show in Proposition 5.4, but it does *not* apply to the softmax loss. Next, we prove an auxiliary lemma that is used in the proof of Theorem 5.1.

Lemma 5.2. Let ϕ be j-compatible, $H_j \triangleq \mathbf{I} - \mathbf{1}e_j^\top$, and $\Phi(x) \triangleq \phi(H_j x)$, then

$$\Phi^*(y) = \begin{cases} \phi^*(y - y_j e_j) & \text{if } \langle \mathbf{1}, y\rangle = 0, \\ +\infty & \text{otherwise.} \end{cases}$$

Proof. We have that $\operatorname{Ker} H_j = \{x \,|\, H_j x = 0\} = \{t\mathbf{1} \,|\, t \in \mathbb{R}\}$ and its orthogonal complement is $\operatorname{Ker}^\perp H_j = \{x \,|\, \langle \mathbf{1}, x\rangle = 0\}$.

$$\begin{aligned} \Phi^*(y) &= \sup\{\langle y, x\rangle - \Phi(x) \,|\, x \in \mathbb{R}^m\} \\ &= \sup\{\langle y, x^\parallel\rangle + \langle y, x^\perp\rangle - \phi(H_j x^\perp) \,|\, x = x^\parallel + x^\perp\}, \end{aligned}$$

where $x^{\|} \in \operatorname{Ker} H_j$ and $x^{\perp} \in \operatorname{Ker}^{\perp} H_j$. It follows that $\Phi^*(y)$ can only be finite if $\langle y, x^{\|} \rangle = 0$, which implies $y \in \operatorname{Ker}^{\perp} H_j$. Let $H_j^{\dagger}$ be the Moore-Penrose pseudoinverse of H_j. For a $y \in \operatorname{Ker}^{\perp} H_j$, we can write

$$\begin{aligned}
\Phi^*(y) &= \sup\{\langle y, H_j^{\dagger} H_j x^{\perp} \rangle - \phi(H_j x^{\perp}) \,|\, x^{\perp} \in \operatorname{Ker}^{\perp} H_j\} \\
&= \sup\{\langle (H_j^{\dagger})^{\top} y, z \rangle - \phi(z) \,|\, z \in \operatorname{Im} H_j\} \qquad (5.7) \\
&\leq \sup\{\langle (H_j^{\dagger})^{\top} y, z \rangle - \phi(z) \,|\, z \in \mathbb{R}^m\} = \phi^*((H_j^{\dagger})^{\top} y),
\end{aligned}$$

where $\operatorname{Im} H_j = \{H_j x \,|\, x \in \mathbb{R}^m\}$. Using rank-1 update of the Moore-Penrose pseudoinverse (Petersen, Pedersen, et al., 2008, § 3.2.7), we can compute

$$(H_j^{\dagger})^{\top} = \mathbf{I} - e_j e_j^{\top} - \frac{1}{m}(\mathbf{1} - e_j)\mathbf{1}^{\top}.$$

Since $y \in \operatorname{Ker}^{\perp} H_j$, the last term is zero and we have $(H_j^{\dagger})^{\top} y = y - y_j e_j$. Finally, we use the fact that ϕ is j-compatible to prove that the inequality in (5.7) is satisfied with equality. We have that $\operatorname{Im} H_j = \{z \,|\, z_j = 0\}$ and $(y - y_j e_j)_j = 0$. Therefore, when $\langle \mathbf{1}, y \rangle = 0$,

$$\Phi^*(y) = \sup\{\langle y - y_j e_j, z \rangle - \phi(z) \,|\, z_j = 0\} = \phi^*(y - y_j e_j).$$

□

Lemma 5.2 can be used to easily compute the convex conjugate of a j-compatible loss function. Next, we use it in the derivation of the dual objective for a general multiclass classification problem.

Theorem 5.1. Let ϕ_i be y_i-compatible for each $i = 1, \ldots, n$, let $\lambda > 0$ be a regularization parameter, and let $K = X^{\top} X$ be the Gram matrix. The primal and Fenchel dual objective functions are given as

$$\begin{aligned}
P(W) &= \frac{1}{n} \sum_{i=1}^{n} \phi_i \left(W^{\top} x_i - \langle w_{y_i}, x_i \rangle \mathbf{1} \right) + \frac{\lambda}{2} \operatorname{\mathbf{tr}} \left(W^{\top} W \right), \qquad (5.8) \\
D(A) &= \begin{cases} -\frac{1}{n} \sum_{i=1}^{n} \phi_i^* \left(-\lambda n (a_i - a_{y_i,i} e_{y_i}) \right) - \frac{\lambda}{2} \operatorname{\mathbf{tr}} \left(A K A^{\top} \right) & \text{if } \langle \mathbf{1}, a_i \rangle = 0 \; \forall i, \\ +\infty & \text{otherwise.} \end{cases}
\end{aligned}$$

Moreover, the primal variables W can be computed as $W = X A^{\top}$, and the prediction scores on x_i can be computed as $f(x_i) = W^{\top} x_i = A K_i$, where K_i is the i-th column of the Gram matrix K.

Proof. We use Fenchel duality (see § A.2.2, page 184), to write

$$\begin{aligned}
P(W) &= g(X^{\top} W) + f(W), \\
D(A) &= -g^*(-A^{\top}) - f^*(X A^{\top}),
\end{aligned}$$

for the functions g and f defined as

$$g(X^\top W) = \frac{1}{n} \sum_{i=1}^{n} \Phi_i \left(W^\top x_i \right) = \frac{1}{n} \sum_{i=1}^{n} \phi_i \left(H_{y_i} W^\top x_i \right),$$
$$f(W) = \frac{\lambda}{2} \mathbf{tr} \left(W^\top W \right) = \frac{\lambda}{2} \|W\|_F^2,$$

where $H_{y_i} = \mathbf{I} - \mathbf{1} e_{y_i}^\top$. One can easily verify that

$$g^*(-A^\top) = \frac{1}{n} \sum_{i=1}^{n} \Phi_i^*(-n a_i), \qquad f^*(X A^\top) = \frac{\lambda}{2} \left\| \frac{1}{\lambda} X A^\top \right\|_F^2.$$

From Lemma 5.2, we have that

$$\Phi_i^*(-n a_i) = \begin{cases} \phi^*(-n(a_i - a_{y_i,i} e_{y_i})) & \text{if } \langle \mathbf{1}, -n a_i \rangle = 0, \\ +\infty & \text{otherwise.} \end{cases}$$

To complete the proof, we redefine $A \leftarrow \frac{1}{\lambda} A$ for convenience, and use the first order optimality condition (Theorem A.1) for the $W = X A^\top$ formula. □

Finally, we show that Theorem 5.1 applies to the loss functions that we consider. Even though the next proposition is formulated for the top-k hinge loss, it also applies to the multiclass SVM loss as the latter is a special case with $k = 1$.

Proposition 5.4. The top-k hinge losses (α) and (β) are y_i-compatible.

Proof. Let $c = \mathbf{1} - e_{y_i}$ and consider the loss ϕ_k. As in Proposition 5.2, we have

$$\max_{a,\, a_{y_i}=0} \{ \langle a, b \rangle - \phi_k(a) \} = \min_{\lambda \in \Delta_k} \{ - \langle c, \lambda \rangle + \max_{a,\, a_{y_i}=0} \langle a, b - \lambda \rangle \} = \phi_k^*(b),$$

where we used that $c_{y_i} = 0$ and $b_{y_i} = 0$ (see Definition 5.3), i.e. the y_i-th coordinate has no influence in the conjugate. The same holds for the top-k hinge loss (β). □

We have repeated the derivation from § 5.7 in (Shalev-Shwartz and Zhang, 2014) as there is a typo in their optimization problem (20) leading to the conclusion that $a_{y_i,i}$ must be 0 at the optimum. Lemma 5.2 fixes this by making the requirement $a_{y_i,i} = -\sum_{j \neq y_i} a_{j,i}$ explicit. Note that this modification is already mentioned in their pseudo-code for Prox-SDCA.

5.3.2 SDCA Optimization for Top-k SVM

In this section, we describe how to apply the *stochastic dual coordinate ascent* (SDCA) framework of Shalev-Shwartz and Zhang, (2013b) to optimize the proposed top-k hinge loss. SDCA is a well developed optimization scheme that has various extensions covering scalar and vector valued loss functions, nondifferentiable Lipschitz and smooth loss functions, as well as the vanilla and the accelerated schemes (Shalev-Shwartz and Zhang, 2014). It also enjoys strong convergence guarantees and is easy to adapt to our problem. We use the vanilla optimization scheme for vector valued Lipschitz losses. In particular, we iteratively update a batch $a_i \in \mathbb{R}^m$ of dual variables corresponding to the training pair (x_i, y_i), so as to maximize the dual objective $D(A)$ from Theorem 5.1. We also maintain the primal variables $W = XA^\top$ and stop when the relative duality gap is below ϵ. This procedure is summarized in Algorithm 5.1.

Algorithm 5.1 Top-k Multiclass SVM

```
 1: Input: training data $\{(x_i, y_i)_{i=1}^n\}$, parameters: $1 \le k < m$ and $\lambda > 0$, $\epsilon > 0$
 2: Output: $W \in \mathbb{R}^{d \times m}$, $A \in \mathbb{R}^{m \times n}$
 3: Initialize: $W \leftarrow \mathbf{0}$, $A \leftarrow \mathbf{0}$
 4: repeat
 5:     randomly permute training data
 6:     for $i = 1$ to $n$ do
 7:         $s_i \leftarrow W^\top x_i$    {prediction scores}
 8:         $a_i^{\text{old}} \leftarrow a_i$    {cache previous values}
 9:         $a_i \leftarrow \mathit{update}(k, \lambda, \|x_i\|^2, y_i, s_i, a_i)$    {see Propositions 5.5 and 5.6}
10:         $W \leftarrow W + x_i(a_i - a_i^{\text{old}})^\top$    {rank-1 update}
11:     end for
12: until relative duality gap is below $\epsilon$
```

Let us make a few comments on the advantages of the proposed method. First, apart from the update step which we discuss below, all main operations can be computed using a BLAS library, which makes the overall implementation efficient. Second, the update step in Line 9 is optimal in the sense that it yields maximal dual objective increase jointly over m variables. This is opposed to SGD updates with data-independent step sizes, as well as to maximal but *scalar* updates in scalar SDCA variants. Finally, we have a well-defined stopping criterion as we can compute the duality gap. The latter is especially attractive if there is a time budget for learning; see also the discussion on the tradeoffs of large scale learning by Bousquet and Bottou, (2008). Finally, we note that top-k SVM can be easily kernelized since $f(x_i) = W^\top x_i = AK_i$ (see Theorem 5.1).

Dual Variables Update (α)

In this section, we derive an SDCA update step for the proposed top-k hinge loss (5.3) from § 5.2.2. Specifically, we show that optimization of the dual objective $D(A)$ over $a_i \in \mathbb{R}^m$ given other variables fixed is an instance of a regularized (biased) projection problem onto the top-k simplex $\Delta_k(\frac{1}{\lambda n})$.

Let $a^{\setminus j}$ be obtained by removing the j-th coordinate from a vector $a \in \mathbb{R}^m$.

Proposition 5.5. Given a sample (x_i, y_i), let $c \triangleq \mathbf{1} - e_{y_i}$, and let the loss function ϕ_i in (5.8) be the top-k hinge loss (α),

$$\phi_i(a) = \max\left\{0, \frac{1}{k}\sum_{j=1}^{k}(a + c)_{\pi_j}\right\}.$$

The following two problems are equivalent with $a_i^{\setminus y_i} = -x$ and $a_{y_i,i} = \langle \mathbf{1}, x \rangle$,

$$\max_{a_i}\{D(A) \mid \langle \mathbf{1}, a_i \rangle = 0\} \;\equiv\; \min_x\{\|b - x\|^2 + \rho \langle \mathbf{1}, x \rangle^2 \mid x \in \Delta_k(\tfrac{1}{\lambda n})\},$$

where $b = \frac{1}{\langle x_i, x_i \rangle}\Big(q^{\setminus y_i} + (1 - q_{y_i})\mathbf{1}\Big)$, $q = W^\top x_i - \langle x_i, x_i \rangle a_i$, and $\rho = 1$.

Proof. Using Proposition 5.2 and Theorem 5.1, we write

$$\max_{a_i}\Big\{ -\frac{1}{n}\phi_i^*\left(-\lambda n(a_i - a_{y_i,i}e_{y_i})\right) - \frac{\lambda}{2}\,\mathbf{tr}\left(AKA^\top\right) \mid \langle \mathbf{1}, a_i \rangle = 0\Big\}.$$

For the loss function, we use that $\langle \mathbf{1}, a_i \rangle = 0$ and get

$$-\frac{1}{n}\phi_i^*\left(-\lambda n(a_i - a_{y_i,i}e_{y_i})\right) = -\lambda \langle c, a_i - a_{y_i,i}e_{y_i} \rangle = \lambda a_{y_i,i},$$

with $-\lambda n(a_i - a_{y_i,i}e_{y_i}) \in \Delta_k$. One can verify that the latter constraint is equivalent to $-a_i^{\setminus y_i} \in \Delta_k(\frac{1}{\lambda n})$, $a_{y_i,i} = \langle \mathbf{1}, -a_i^{\setminus y_i} \rangle$. Similarly for the regularization term,

$$\mathbf{tr}\left(AKA^\top\right) = K_{ii}\langle a_i, a_i \rangle + 2\sum_{j \neq i} K_{ij}\langle a_i, a_j \rangle + \text{const},$$

where the const does not depend on a_i. Note that

$$q = \sum_{j \neq i} K_{ij} a_j = AK_i - K_{ii}a_i$$

can be computed using the “old” a_i. Let $x \triangleq -a_i^{\setminus y_i}$, we have

$$\langle a_i, a_i \rangle = \langle \mathbf{1}, x \rangle^2 + \langle x, x \rangle, \qquad \langle q, a_i \rangle = q_{y_i}\langle \mathbf{1}, x \rangle - \langle q^{\setminus y_i}, x \rangle.$$

Plugging everything together and multiplying with $-2/\lambda$, we obtain

$$\min_{x \in \Delta_k(\frac{1}{\lambda n})} -2\langle \mathbf{1}, x \rangle + 2\Big(q_{y_i}\langle \mathbf{1}, x \rangle - \langle q^{\setminus y_i}, x \rangle\Big) + K_{ii}\Big(\langle \mathbf{1}, x \rangle^2 + \langle x, x \rangle\Big).$$

Collecting the corresponding terms finishes the proof. □

Dual Variables Update (β)

By analogy, we derive an SDCA update step for the top-k hinge loss (5.5) from § 5.2.3. We show that optimization of the dual objective $D(A)$ over $a_i \in \mathbb{R}^m$ given other variables fixed is an instance of a regularized (biased) projection problem onto the top-k simplex $\tilde{\Delta}_k(\frac{1}{\lambda n})$.

Proposition 5.6. Given a sample (x_i, y_i), let $c \triangleq \mathbf{1} - e_{y_i}$, and let the loss function ϕ_i in (5.8) be the top-k hinge loss (β),

$$\phi_i(a) = \frac{1}{k}\sum_{j=1}^{k}\max\Big\{0, (a + c)_{\pi_j}\Big\}.$$

The following two problems are equivalent with $a_i^{\setminus y_i} = -x$ and $a_{y_i,i} = \langle \mathbf{1}, x \rangle$,

$$\max_{a_i} \{ D(A) \mid \langle \mathbf{1}, a_i \rangle = 0 \} \;\equiv\; \min_{x} \{ \|b - x\|^2 + \rho \langle \mathbf{1}, x \rangle^2 \mid x \in \tilde{\Delta}_k(\tfrac{1}{\lambda n}) \},$$

where $b = \frac{1}{\langle x_i, x_i \rangle} \left(q^{\setminus y_i} + (1 - q_{y_i}) \mathbf{1} \right)$, $q = W^\top x_i - \langle x_i, x_i \rangle a_i$, and $\rho = 1$.

Proof. The proof is analogous to the proof of Proposition 5.5. □

Propositions 5.5 and 5.6 effectively cast the problem of finding the "best" a_i given other dual variables fixed as an equivalent problem of projecting a certain vector b, computed from the prediction scores $f(x_i)$, onto the top-k simplex $\Delta_k(\frac{1}{\lambda n})$, respectively the set $\tilde{\Delta}_k(\frac{1}{\lambda n})$, which are also the effective domains of the corresponding conjugate losses. Next, we propose efficient algorithms to compute these biased projections for any $1 \leq k < m$ and $\rho \geq 0$.

5.4 Efficient Projection Algorithms

One of the main technical contributions of this chapter is an algorithm for efficient projection onto the top-k simplex which we develop in this section. The optimization problem was introduced in Proposition 5.5, and we note that it reduces to the Euclidean projection onto $\Delta_k(r)$ for $\rho = 0$, while for $\rho > 0$ it biases the solution to be orthogonal to $\mathbf{1}$. Finally, we consider the biased projection onto $\tilde{\Delta}_k(r)$.

- In § 5.4.1, we highlight that the set $\Delta_k(r)$ is substantially different from the standard simplex and none of the existing methods can be used.
- In § 5.4.2, we consider the projection onto the top-k cone, which is obtained by dropping the constraint $\langle \mathbf{1}, x \rangle \leq r$ in the definition of $\Delta_k(r)$.
- In § 5.4.3, we discuss the complete algorithm for computing the projection onto the top-k simplex $\Delta_k(r)$, which is used for the top-k hinge loss (α).
- In § 5.4.4, we consider the biased projection onto the set $\tilde{\Delta}_k(r)$, which is used in SDCA optimization of the top-k hinge loss (β).

5.4.1 Continuous Quadratic Knapsack Problem

Finding the Euclidean projection onto the standard simplex is an instance of the general optimization problem

$$\min_{x \in \mathbb{R}^d} \{ \|a - x\|_2^2 \mid \langle b, x \rangle \leq r,\; l \leq x_i \leq u,\; 1 \leq i \leq d \},$$

known as the *continuous quadratic knapsack* problem. For example, to project onto the unit simplex we set $b = \mathbf{1}$, $l = 0$ and $r = u = 1$. This is a well examined problem and several highly efficient algorithms are available, e.g. see the surveys by Patriksson, (2008) and Patriksson and Strömberg, (2015). A common modification considered in the literature is to enforce the *equality* constraint, $\langle b, x \rangle = r$, which generally leads to simpler algorithms. In our approach, the equality constraint is considered as a special case and is handled separately.

Let us now discuss why projecting onto $\Delta_k(r)$ is substantially different from solving the knapsack problem. The problem that we seek to solve is

$$\min_{x \in \mathbb{R}^d} \{ \|a - x\|_2^2 + \rho \langle \mathbf{1}, x \rangle^2 \mid \langle \mathbf{1}, x \rangle \leq r, \ 0 \leq x_i \leq \tfrac{1}{k} \langle \mathbf{1}, x \rangle, \ 1 \leq i \leq d \}. \tag{5.9}$$

The first main difference is the upper bound on x_i. All existing algorithms expect that u is a fixed constant, which allows one to consider the decompositions

$$\min_{x_i \in \mathbb{R}} \{ (a_i - x_i)^2 \mid l \leq x_i \leq u \}, \quad 1 \leq i \leq d,$$

and solve them in closed-form. In our case, the upper bound $\frac{1}{k} \langle \mathbf{1}, x \rangle$ introduces coupling across all variables, which makes the existing algorithms not applicable. The second main difference is the bias term $\rho \langle \mathbf{1}, x \rangle^2$. The additional difficulty introduced by this term is relatively minor and, even though we only need $\rho = 1$ in Proposition 5.5, we develop an algorithm for the general case $\rho \geq 0$, which also includes the Euclidean projection as a special case with $\rho = 0$.

The only situation where (5.9) reduces to the knapsack problem is when the constraint $\langle \mathbf{1}, x \rangle \leq r$ is satisfied with equality. In that case, we let $u = r/k$ and note that any algorithm for the knapsack problem can be used. We choose the variable fixing algorithm of Kiwiel, (2008b) since it is easy to implement, does not require sorting, and scales linearly in practice. The bias in the projection problem reduces to a constant ρr^2 in this case and has, therefore, no effect.

5.4.2 Projection onto the Top-k Cone

Let us now consider the case where the constraint $\langle \mathbf{1}, x \rangle \leq r$ in the problem (5.9) is satisfied with strict inequality, i.e. $\langle \mathbf{1}, x^* \rangle < r$ for the optimal point x^*. In that case, the constraint has no influence on the solution and can be removed, which leads us to the (biased) projection onto the *top-k cone* addressed next.

Lemma 5.3. Let $x^* \in \mathbb{R}^d$ be the solution to the following optimization problem

$$\min_{x} \{ \|a - x\|^2 + \rho \langle \mathbf{1}, x \rangle^2 \mid 0 \leq x_i \leq \tfrac{1}{k} \langle \mathbf{1}, x \rangle, \ 1 \leq i \leq d \},$$

and let $U \triangleq \{ i \mid x_i^* = \frac{1}{k} \langle \mathbf{1}, x^* \rangle \}$, $M \triangleq \{ i \mid 0 < x_i^* < \frac{1}{k} \langle \mathbf{1}, x^* \rangle \}$, $L \triangleq \{ i \mid x_i^* = 0 \}$.

1. If $U = \varnothing$ and $M = \varnothing$, then $x^* = 0$.
2. If $U \neq \varnothing$ and $M = \varnothing$, then $U = \{\pi_1, \ldots, \pi_k\}$ and

$$x_i^* = \frac{1}{k + \rho k^2} \sum_{i=1}^{k} a_{\pi_i} \tag{5.10}$$

for all $i \in U$, where π_i is the index of the ith largest component in a.

3. Otherwise ($M \neq \varnothing$), the following system of linear equations holds

$$\begin{cases} u &= \Big(|M| \sum_{i \in U} a_i + (k - |U|) \sum_{i \in M} a_i \Big) / D, \\ t' &= \Big(|U| \, (1 + \rho k) \sum_{i \in M} a_i - (k - |U| + \rho k \, |M|) \sum_{i \in U} a_i \Big) / D, \\ D &= (k - |U|)^2 + (|U| + \rho k^2) \, |M| , \end{cases} \tag{5.11}$$

together with the feasibility constraints on $t \triangleq t' + \rho u k$

$$\max_{i \in L} a_i \leq t \leq \min_{i \in M} a_i, \qquad \max_{i \in M} a_i \leq t + u \leq \min_{i \in U} a_i, \tag{5.12}$$

and we have $x^* = \min\{\max\{0, a - t\}, u\}$.

Proof. We consider an equivalent problem

$$\min_{x,s} \{ \tfrac{1}{2} \|a - x\|^2 + \tfrac{1}{2} \rho s^2 \mid \langle \mathbf{1}, x \rangle = s, \ 0 \leq x_i \leq \tfrac{s}{k}, \ 1 \leq i \leq d \}.$$

Let t, $\mu_i \geq 0$, $\nu_i \geq 0$ be the dual variables, and let $\mathcal{L}$ be the Lagrangian:

$$\mathcal{L}(x, s, t, \mu, \nu) = \tfrac{1}{2} \|a - x\|^2 + \tfrac{1}{2} \rho s^2 + t(\langle \mathbf{1}, x \rangle - s) - \langle \mu, x \rangle + \left\langle \nu, x - \tfrac{s}{k} \mathbf{1} \right\rangle .$$

From the KKT conditions (§ A.1.2, page 180), we have that

$$x - a + t\mathbf{1} - \mu + \nu = 0, \quad \rho s - t - \tfrac{1}{k} \langle \mathbf{1}, \nu \rangle = 0, \quad \mu_i x_i = 0, \quad \nu_i (x_i - \tfrac{s}{k}) = 0.$$

We also obtain

$$x_i = \min\{\max\{0, a_i - t\}, \tfrac{s}{k}\}, \quad \nu_i = \max\{0, a_i - t - \tfrac{s}{k}\}, \quad s = \tfrac{1}{\rho}(t + \tfrac{1}{k} \langle \mathbf{1}, \nu \rangle).$$

Let $p \triangleq \langle \mathbf{1}, \nu \rangle$. We have $t = \rho s - \frac{p}{k}$ and, using the definition of U and M,

$$\begin{aligned} s &= \sum_{i \in U} \frac{s}{k} + \sum_{i \in M} (a_i - t) = \sum_{i \in M} a_i - |M| \, (\rho s - \frac{p}{k}) + |U| \, \frac{s}{k}, \\ p &= \sum_{i \in U} (a_i - t - \frac{s}{k}) = \sum_{i \in U} a_i - |U| \, (\rho s - \frac{p}{k}) - |U| \, \frac{s}{k}. \end{aligned}$$

In the case $U \neq \varnothing$ and $M = \varnothing$ we get the simplified equations

$$\begin{aligned} s &= \sum_{i \in U} \frac{s}{k} = |U| \frac{s}{k} &&\implies |U| = k, \\ p &= \sum_{i \in U} a_i - k \rho s + p - s &&\implies x_i = \frac{s}{k} = \frac{1}{k + \rho \, k^2} \sum_{i \in U} a_i, \ i \in U. \end{aligned}$$

In the remaining case, solving this system for $u \triangleq \frac{s}{k}$ and $t' \triangleq -\frac{p}{k}$, we get exactly the system in (5.11). The constraints (5.12) follow from the definition of the sets U, M, L, and ensure that the computed thresholds (t, u) are compatible with the corresponding partitioning of the index set. □

The above lemma suggests that the projection onto the top-k cone can be computed by considering distinction into three cases. The first special case is when the resulting projection is zero, and we focus on this case next. For the standard simplex, where the cone is simply the positive orthant $\mathbb{R}^d_+$, it is easy to see that the projection is 0 if and only if all $a_i \leq 0$. For the Δ_k, however, the situation is slightly more involved.

Lemma 5.4. The biased projection x^* onto the top-k cone is zero if $\sum_{i=1}^k a_{\pi_i} \leq 0$ (sufficient condition). If $\rho = 0$ this is also necessary.

Proof. Let $K \triangleq \{x \,|\, 0 \leq x_i \leq \frac{1}{k} \langle \mathbf{1}, x \rangle\}$ be the top-k cone. It is known, (e.g., see (Dattorro, 2010)) that the Euclidean projection of a onto K is 0 if and only if

$$a \in N_K(0) \triangleq \{y \,|\, \forall x \in K,\, \langle y, x \rangle \leq 0\},$$

i.e. a is in the *normal cone* to K at 0. Therefore, we obtain as an equivalent condition that $\max_{x \in K} \langle a, x \rangle \leq 0$. Take any $x \in K$ and let $s = \langle \mathbf{1}, x \rangle$. If $s > 0$, at least k components in x must be positive. To maximize $\langle a, x \rangle$, we need exactly k positive $x_i = \frac{s}{k}$ corresponding to the k largest components in a. That would result in $\langle a, x \rangle = \frac{s}{k} \sum_{i=1}^k a_{\pi_i}$, which is nonpositive if and only if $\sum_{i=1}^k a_{\pi_i} \leq 0$.

For $\rho > 0$, the objective function has an additional term $\rho \langle \mathbf{1}, x \rangle^2$ that vanishes at $x = 0$. Therefore, if $x = 0$ is optimal for the Euclidean projection, it must also be optimal for the biased projection. □

The second special case corresponds to another extreme situation where the k largest components of a are all set to a constant, while the remaining elements are set to zero. This case is also considered separately as we discuss next.

Projection onto the top-k cone. Lemmas 5.3 and 5.4 suggest a simple algorithm for the (biased) projection onto the top-k cone based on case distinction.

- In case 1, we check if $\rho = 0$ and $\sum_{i=1}^k a_{\pi_i} \leq 0$, then set $x^* = 0$. For the biased projection ($\rho > 0$), we leave $x^* = 0$ as the fallback case in the end.

- In case 2, we compute x^* using (5.10) and check if it is compatible with the corresponding sets U and L, using the constraints (5.12). Here, we note that $M = \varnothing$, and the threshold t can be computed from (5.12) as

$$t = \min_{i \in U} a_i - u = a_{\pi_k} - \frac{1}{k + \rho k^2} \sum_{i=1}^k a_{\pi_i}.$$

- In case 3, we suggest a simple exhaustive search strategy. We sort a and loop over the feasible partitions U, M, L until we find a solution to (5.11) that satisfies (5.12). Since we know that $0 \leq |U| < k$ and $k \leq |U| + |M| \leq d$, we can limit the search to $k(d - k + 1) = O(kd)$ iterations in the worst case, where each iteration requires a constant number of operations.

- For the biased projection ($\rho > 0$), we leave $x^* = 0$ as the fallback case since Lemma 5.4 gives only a sufficient condition.

The proposed algorithm has the runtime complexity of $O(d \log d + kd)$, which for small k is comparable to simplex projection algorithms based on sorting.

5.4.3 Projection onto the Top-k Simplex

In this section, we combine the results from §§ 5.4.1 and 5.4.2 to obtain a complete algorithm for the (biased) projection onto the top-k simplex $\Delta_k(r)$. As we argued above, the problem (5.9) becomes either the knapsack problem or the (biased) projection onto the top-k cone depending on the constraint $\langle \mathbf{1}, x \rangle \leq r$ at the optimum. We already have the algorithms for both problems, but we need a way to check which of the two cases apply. The following lemma will be useful.

Lemma 5.5. Let $x^* \in \mathbb{R}^d$ be the solution to the problem (5.9), let (t, u) be the optimal thresholds such that $x^* = \min\{\max\{0, a - t\}, u\}$, let U be defined as in Lemma 5.3, and let $p \triangleq \sum_{i \in U} a_i - |U| (t + u)$, then $\lambda = t + \frac{p}{k} - \rho r \geq 0$.

Proof. As in Lemma 5.3, we consider the equivalent problem:

$$\min_{x,s} \{ \tfrac{1}{2} \|a - x\|^2 + \tfrac{1}{2} \rho s^2 \mid \langle \mathbf{1}, x \rangle = s,\ s \leq r,\ 0 \leq x_i \leq \tfrac{s}{k},\ 1 \leq i \leq d \}.$$

Let t, $\lambda \geq 0$, $\mu_i \geq 0$, $\nu_i \geq 0$ be the dual variables, and let $\mathcal{L}$ be the Lagrangian,

$$\begin{aligned} \mathcal{L}(x, s, t, \lambda, \mu, \nu) = \tfrac{1}{2} \|a - x\|^2 + \tfrac{1}{2} \rho s^2 + t(\langle \mathbf{1}, x \rangle - s) \\ + \lambda(s - r) - \langle \mu, x \rangle + \left\langle \nu, x - \tfrac{s}{k} \mathbf{1} \right\rangle . \end{aligned}$$

From the KKT conditions, we have that

$$\begin{aligned} \partial_x \mathcal{L} = x - a + t\mathbf{1} - \mu + \nu = 0, \quad & \partial_s \mathcal{L} = \rho s - t + \lambda - \tfrac{1}{k} \langle \mathbf{1}, \nu \rangle = 0, \\ \mu_i x_i = 0, \quad & \nu_i (x_i - \tfrac{s}{k}) = 0, \quad \lambda(s - r) = 0. \end{aligned}$$

If $s < r$, then $\lambda = 0$ and we recover the top-k cone problem of Lemma 5.3. Otherwise, we have that $s = r$ and $\lambda = t + \frac{1}{k} \langle \mathbf{1}, \nu \rangle - \rho r \geq 0$. The fact that $\nu_i = \max\{0, a_i - t - u\}$, where $u = \frac{r}{k}$, completes the proof. □

Projection onto the top-k simplex. We can now use Lemma 5.5 to compute the (biased) projection onto $\Delta_k(r)$ as follows.

- First, we check the special cases of zero and constant projections, as we did before in § 5.4.2 (cases 1 and 2).
- If that fails, we proceed with the knapsack problem using the variable fixing algorithm of Kiwiel, (2008b), since it is the faster alternative.
- Having the thresholds (t, u) and the partitioning into the sets U, M, L from the knapsack problem, we compute the value of λ as given in Lemma 5.5. If $\lambda \geq 0$, we are done, the projection onto the top-k simplex coincides with the solution of the knapsack problem.

- Otherwise, we know that $\langle \mathbf{1}, x \rangle < r$ and use the algorithm for the (biased) projection onto the top-k cone from § 5.4.2 (case 3).

The proposed algorithm is generally fast in practice and has runtime comparable to common algorithms for the Euclidean projection onto the standard simplex. As described above, the worst case runtime is dominated by the algorithm of Kiwiel, (2008b), which has the complexity of $O(d^2)$ on pathological inputs with elements growing exponentially (Condat, 2014). In practice, however, the observed complexity is linear and is competitive with the breakpoint searching algorithms based on sorting (Condat, 2014; Kiwiel, 2008b). Given that the knapsack problem can also be solved in time $O(d \log d)$ (Kiwiel, 2008a), the overall complexity for solving the (biased) projection onto the top-k simplex is $O(d \log d + kd)$.

5.4.4 Biased Continuous Quadratic Knapsack Problem

In this section, we derive an algorithm to compute the biased projection onto $\tilde{\Delta}_k$, which is used in SDCA optimization of the top-k hinge loss (β). The resulting algorithm is similar in spirit to the one derived in § 5.4.3 above.

Recall that the upper bound on x_i in the set $\tilde{\Delta}_k(r)$ is a fixed constant r/k. Therefore, the Euclidean projection onto $\tilde{\Delta}_k(r)$ is an instance of the continuous quadratic knapsack problem from § 5.4.1. However, the update step in the SDCA framework corresponds to a *biased* projection, as we saw above, where the added bias $\rho \langle \mathbf{1}, x \rangle^2$ comes from the ℓ_2-regularizer in the training objective. To address this issue, we follow the proofs of Lemmas 5.3, 5.5, and derive an algorithm to compute the biased projection onto $\tilde{\Delta}_k(r)$.

We know from Proposition 5.6 that the SDCA update step for the top-k hinge loss (5.5) is equivalent with $l = 0$ and $u = r/k$ to the following problem:

$$\min_{x,s} \left\{ \tfrac{1}{2} \left\| a - x \right\|^2 + \tfrac{1}{2} \rho s^2 \mid \langle \mathbf{1}, x \rangle = s,\ s \le r,\ l \le x_i \le u,\ 1 \le i \le d \right\}.$$

Let t, $\lambda \ge 0$, $\mu_i \ge 0$, $\nu_i \ge 0$ be the dual variables, and let $\mathcal{L}$ be the Lagrangian,

$$\begin{aligned} \mathcal{L}(x, s, t, \lambda, \mu, \nu) = {} & \tfrac{1}{2} \left\| a - x \right\|^2 + \tfrac{1}{2} \rho s^2 + t(\langle \mathbf{1}, x \rangle - s) \\ & + \lambda(s - r) - \langle \mu, l\mathbf{1} - x \rangle + \langle \nu, x - u\mathbf{1} \rangle . \end{aligned}$$

From the KKT conditions, we have that

$$\begin{aligned} \partial_x \mathcal{L} = x - a + t\mathbf{1} - \mu + \nu = 0, \qquad & \partial_s \mathcal{L} = \rho s - t + \lambda = 0, \\ \mu_i (l - x_i) = 0, \qquad & \nu_i (x_i - u) = 0, \qquad \lambda(s - r) = 0, \end{aligned}$$

which then leads to

$$x = a - t\mathbf{1} + \mu - \nu = \min\{\max\{l, x - t\}, u\}, \qquad \lambda = t - \rho s.$$

Now, we can do case distinction based on the sign of λ. If $\lambda > 0$, then $\langle \mathbf{1}, x \rangle = s = r$ and $t > \rho r$. In this case $\frac{1}{2} \rho s^2 = \frac{1}{2} \rho r^2 \equiv \text{const}$, therefore this term can be ignored and we get the knapsack problem from § 5.4.1. Otherwise, if $s < r$, then $\lambda = 0$ and $t = \rho s$. Using the index sets U, M and L as in Lemma 5.3, we have

$$t = \rho\Big(\sum_L l + \sum_M (a_i - t) + \sum_U u \Big) = \rho\Big(l\,|L| + u\,|U| - t\,|M| + \sum_M a_i \Big).$$

Solving for t with $\rho > 0$, we obtain that

$$t = \Big(l\,|L| + u\,|U| + \sum_M a_i \Big) / \Big((1/\rho) + |M| \Big). \tag{5.13}$$

Biased continuous quadratic knapsack problem. To compute the biased projection onto $\tilde{\Delta}_k(r)$, we follow the same steps as in § 5.4.3.

- First, we solve the knapsack problem using the algorithm of Kiwiel, (2008b), which also computes the dual variable t. If $t > \rho r$, then we are done.
- Otherwise, we sort a and loop over the feasible index sets U, M, and L. We stop once we find a t that satisfies (5.13) and is compatible with the corresponding index sets.

5.5 Experiments

This section is concerned with the experimental evaluation of the proposed top-k SVM and the algorithm for computing the projection onto the top-k simplex.

- In § 5.5.1, we show that the (biased) projection onto the top-k simplex is scalable with respect to the input dimension, and is comparable in runtime to an efficient variable fixing algorithm of Kiwiel, (2008b), which can be used to solve the continuous quadratic knapsack problem and find the Euclidean projection onto the standard simplex.
- In § 5.5.2, we show that the top-k multiclass SVM with both versions of the top-k hinge loss, (5.3) and (5.5), denoted top-k SVM$^\alpha$ and top-k SVM$^\beta$ respectively, leads to improvements in top-k accuracy consistently over all datasets and choices of k. In particular, we note improvements compared to the multiclass SVM of Crammer and Singer, (2001), which corresponds in our notation to top-1 SVM$^\alpha$, top-1 SVM$^\beta$.

5.5.1 Scaling of the Projection onto the Top-k Simplex

In this section, our goal is to demonstrate effectively linear scaling that is observed in practice for the variable fixing algorithms that we employ in our SDCA training procedure. We follow the experimental setup of Liu and Ye, (2009). We sample 1000 points from the normal distribution $\mathcal{N}(0, \mathbf{I})$ in $\mathbb{R}^d$ for increasing values of d, and solve the corresponding projection problems using: (i) the algorithm of Kiwiel, (2008b) denoted as Knapsack, and (ii) using our proposed method from § 5.4.3 for projecting onto the set Δ_k for different values of $k = 1, 5, 10$.

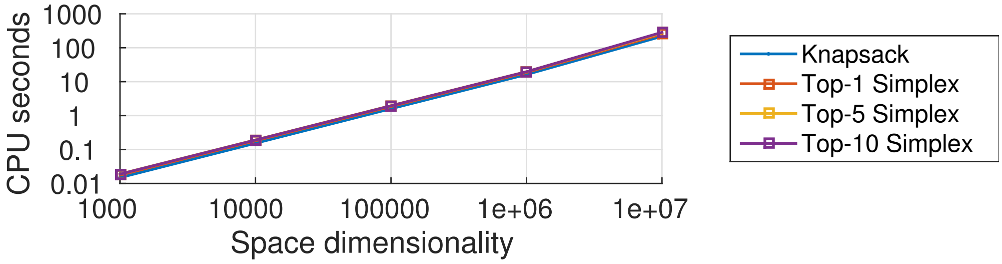


Figure 5.3.: Scaling of the proposed algorithm from § 5.4.3 for projecting onto the top-k simplex $\Delta_k(r)$ compared to the algorithm of Kiwiel, (2008b) for solving the continuous quadratic knapsack problem.

We report the total CPU time taken on a single Intel(R) Xeon(R) 2.20GHz processor. We plot the respective runtimes in Figure 5.3, and observe that the scaling in the problem dimension is essentially the same both for the knapsack problem and for the proposed projection onto the top-k simplex.

5.5.2 Image Classification Experiments

In this section, we present an extensive experimental evaluation of the proposed top-k multiclass SVM on real world data. We evaluate the method on five publicly available image classification datasets of different scale and complexity, please refer to Table 5.1 for the basic statistics.

Dataset	m	n	d
Caltech 101 Silhouettes (Swersky et al., 2012)	101	4100	784
MIT Indoor 67 (Quattoni and Torralba, 2009)	67	5354	4096
SUN 397 (Xiao et al., 2010)	397	19850	4096
Places 205 (Zhou et al., 2014)	205	2448873	4096
ImageNet 2012 (Russakovsky et al., 2015)	1000	1281167	4096

Table 5.1.: Statistics of the image classification benchmarks that we use in our experiments (m: # classes, n: # training examples, d: # feature dimensions).

Methods. We consider the following baseline methods in our experiments.

SVM$^{\text{OVA}}$**:** the classical one-vs-all (OVA) SVM using LibLinear of Fan et al., (2008).

Prec@k, Recall@k**:** the SVM$^{\text{Perf}}$ method of Joachims, (2005) with the corresponding loss function, as described in § 5.2.3 (page 86).

TopPush**:** we use the code provided by Li et al., (2014a).

$\text{W}_{++,\,\text{Q/m}}$**:** our implementation of the Wsabie^{++} algorithm based on the pseudo code by Gupta et al., (2014).

We cross-validate the hyperparameters in the range 10^{-5} to 10^{3}, and extend it when the optimal value is at the boundary. The parameter m in the $\text{W}_{++,\,\text{Q/m}}$

method is set $m = 101$ for Caltech 101 and $m = 67$ for Indoor 67. When a ranking method, like the Recall@k and the TopPush, does not scale to a particular dataset using the multiclass to binary reduction discussed in § 5.2.3, we use the OVA version of the corresponding method. Among the baselines that we tried, only TopPush$^{\text{OVA}}$ scaled to the Places and the ImageNet datasets both in terms of runtime and memory[2]. Additionally, we also report the published results from the literature. In the interest of space, we use the encoding given in Table 5.2 when we refer to a particular method in the following tables.

Features. For the Caltech 101 Silhouettes, we use the features provided by Swersky et al., (2012). For the other datasets, we use a relatively simple image recognition pipeline following Simonyan and Zisserman, (2015). Specifically, we extract the FC7-layer features (after ReLU) using a pre-trained ConvNet. For the scene recognition datasets, we use the Places 205 ConvNet of Zhou et al., (2014), while for the ImageNet experiment we use the Caffe reference model of Jia et al., (2014).

Discussion. Our experimental results are grouped by the scale of the datasets into three tables, which we discuss next.

On Caltech 101 Silhouettes and MIT Indoor 67 (Table 5.3), we make the following observations. First, our proposed top-k SVM consistently outperforms the baseline methods in all top-k metrics. For $k = 1$, we effectively recover the multiclass SVM of Crammer and Singer, (2001), which shows strong performance in top-1 accuracy. However, as we increase $k = 2, 3, 4, 5, 10$ in top-k SVM$^\alpha$ and top-k SVM$^\beta$, we observe an emerging “diagonal” pattern in the respective top-k performance. The correspondence may be not one to one, but the general tendency is that top-k SVM with $k > 1$ produces better top-k accuracy. On Indoor 67, that improvement comes at the cost of a decreased top-1 accuracy, which is in line with our expectations. On Caltech 101 Silhouettes, on the other hand, there is a noticeable increase in top-1 accuracy, e.g. with the top-10 SVM$^\beta$, which may indicate that the top-k error is a better target performance measure on datasets with a large number of classes and possible label noise.

On SUN 397 (Table 5.4), we observe a prominent diagonal pattern in top-k performance for top-k SVM$^\alpha$, while for top-k SVM$^\beta$ we see that even larger values ($k > 20$) may need to be considered to achieve peak top-k results. These

[2] LibLinear required too much memory due to non-sparse features in our experiments.

BLH	(Bu et al., 2013)	**RAS**	(Razavian et al., 2014)
BVLC	(Jia et al., 2014)	**SFT**	(Swersky et al., 2012)
DGE	(Doersch et al., 2013)	**SP**	(Sun and Ponce, 2013)
GWG	(Gong et al., 2014)	**SPM**	(Sánchez et al., 2013)
JVJ	(Juneja et al., 2013)	**XHE**	(Xiao et al., 2010)
KL	(Koskela and Laaksonen, 2014)	**ZLX**	(Zhou et al., 2014)
LSH	(Lapin et al., 2014b)		

Table 5.2.: Encoding of the reference methods for the results from the literature.

Caltech 101 Silhouettes							MIT Indoor 67					
Method	Top-1	Top-2	Top-3	Top-4	Top-5	Top-10	Method	Top-1	Method	Top-1	Method	Top-1
SFT (Top-1)	62.1	-	79.6	-	83.1	-	**BLH**	48.3	**DGE**	66.87	**RAS**	69.0
SFT (Top-2)	61.4	-	79.2	-	83.4	-	**SP**	51.4	**ZLX**	68.24	**KL**	70.1
SFT (Top-5)	60.2	-	78.7	-	83.4	-	**JVJ**	63.10	**GWG**	68.88		

Method	Top-1	Top-2	Top-3	Top-4	Top-5	Top-10	Top-1	Top-2	Top-3	Top-4	Top-5	Top-10
SVM$^{\mathrm{OVA}}$	61.81	73.13	76.25	77.76	78.89	83.57	71.72	81.49	84.93	86.49	87.39	90.45
TopPush	63.11	75.16	78.46	80.19	81.97	86.95	70.52	83.13	86.94	90.00	91.64	95.90
Prec@1	61.29	73.26	76.12	77.76	79.11	83.27	69.03	80.67	85.00	87.16	88.21	91.87
Prec@5	61.73	73.99	76.90	78.50	79.63	84.22	69.18	81.42	85.45	87.61	88.43	91.87
Prec@10	61.90	73.95	76.68	78.46	79.67	84.14	69.18	81.42	85.45	87.61	88.43	91.87
Recall@1	61.55	73.13	77.03	79.41	80.97	85.18	71.57	83.06	87.69	90.45	92.24	96.19
Recall@3	61.51	72.95	76.55	78.72	80.49	84.74	71.42	81.57	85.67	87.39	88.43	92.24
Recall@5	61.60	72.87	76.51	78.76	80.54	84.74	71.49	81.49	85.45	87.24	88.21	92.01
Recall@10	61.51	72.95	76.46	78.72	80.54	84.92	71.42	81.49	85.52	87.24	88.28	92.16
W++,$_{0/m}$	62.33	74.95	78.59	81.45	83.66	89.08	69.33	83.06	88.66	91.72	93.43	97.54
W++,$_{1/m}$	59.69	65.97	68.92	71.61	73.82	80.88	67.39	80.15	85.22	88.88	90.90	95.90
W++,$_{2/m}$	57.39	64.33	67.88	70.13	71.95	77.59	62.61	76.57	82.39	86.19	88.36	93.81
W++,$_{4/m}$	56.78	63.94	67.36	70.05	72.08	78.76	63.13	76.87	82.24	85.67	88.43	94.63
W++,$_{0/192}$	62.29	76.25	79.71	81.40	83.09	88.17	69.78	82.99	88.36	91.49	93.51	97.31
W++,$_{1/192}$	59.56	65.97	69.44	71.65	73.91	79.45	67.24	81.34	85.60	89.03	91.19	95.75
W++,$_{2/192}$	56.78	63.29	67.10	69.87	71.69	78.37	63.28	77.61	84.03	87.99	89.93	94.85
W++,$_{4/192}$	58.13	64.37	67.62	69.92	71.56	78.15	62.54	76.79	84.10	87.61	89.18	94.03
W++,$_{0/256}$	62.68	76.33	79.41	81.71	83.18	88.95	70.07	84.10	89.48	92.46	94.48	**97.91**
W++,$_{1/256}$	59.25	65.63	69.22	71.09	72.95	79.71	68.13	81.49	86.64	89.63	91.42	95.45
W++,$_{2/256}$	55.09	61.81	66.02	68.88	70.61	76.59	64.63	78.43	84.18	88.13	89.93	94.55
W++,$_{4/256}$	56.52	62.29	65.76	68.01	70.13	76.59	60.90	75.97	82.84	86.79	89.63	94.63
top-1 SVM$^{\alpha}$	62.81	74.60	77.76	80.02	81.97	86.91	**73.96**	85.22	89.25	91.94	93.43	96.94
top-2 SVM$^{\alpha}$	63.11	76.16	79.02	81.01	82.75	87.65	73.06	85.67	90.37	92.24	94.48	97.31
top-3 SVM$^{\alpha}$	**63.37**	76.72	79.67	81.49	83.57	88.25	71.57	**86.27**	**91.12**	93.21	94.70	97.24
top-4 SVM$^{\alpha}$	63.20	76.64	79.76	82.36	84.05	88.64	71.42	85.67	90.75	**93.28**	**94.78**	97.84
top-5 SVM$^{\alpha}$	63.29	76.81	80.02	82.75	84.31	88.69	70.67	85.75	90.37	93.21	94.70	**97.91**
top-10 SVM$^{\alpha}$	62.98	**77.33**	80.49	82.66	84.57	89.55	70.00	85.45	90.00	93.13	94.63	97.76
top-20 SVM$^{\alpha}$	59.21	75.64	**80.88**	**83.49**	**85.39**	**90.33**	65.90	84.10	89.93	92.69	94.25	97.54
top-1 SVM$^{\beta}$	62.81	74.60	77.76	80.02	81.97	86.91	73.96	85.22	89.25	91.94	93.43	96.94
top-2 SVM$^{\beta}$	63.55	76.25	79.28	81.14	82.62	87.91	**74.03**	85.90	89.78	92.24	94.10	97.31
top-3 SVM$^{\beta}$	63.94	76.64	79.71	81.36	83.44	87.99	72.99	**86.34**	90.60	92.76	94.40	97.24
top-4 SVM$^{\beta}$	63.94	76.85	80.15	82.01	83.53	88.73	73.06	86.19	**90.82**	92.69	**94.48**	97.69
top-5 SVM$^{\beta}$	63.59	77.03	80.36	82.57	84.18	89.03	72.61	85.60	90.75	92.99	**94.48**	97.61
top-10 SVM$^{\beta}$	**64.02**	77.11	80.49	83.01	84.87	89.42	71.87	85.30	90.45	**93.36**	94.40	**97.76**
top-20 SVM$^{\beta}$	63.37	**77.24**	**81.06**	**83.31**	**85.18**	**90.03**	71.94	85.30	90.07	92.46	94.33	97.39

Table 5.3.: Top: Results from the literature. **Middle:** Our results using the baseline methods. **Bottom:** Our proposed top-k SVM.

SUN 397 (10 splits)

Top-1 accuracy						
	XHE	38.0	**LSH**	49.48 ± 0.3	**ZLX**	54.32 ± 0.1
	SPM	47.2 ± 0.2	**GWG**	51.98	**KL**	54.65 ± 0.2

Method	Top-1	Top-2	Top-3	Top-4	Top-5	Top-10
SVM$^{\text{OVA}}$	55.23 ± 0.6	66.23 ± 0.6	70.81 ± 0.4	73.30 ± 0.2	74.93 ± 0.2	79.00 ± 0.3
TopPush$^{\text{OVA}}$	53.53 ± 0.3	65.39 ± 0.3	71.46 ± 0.2	75.25 ± 0.1	77.95 ± 0.2	85.15 ± 0.3
Recall@1$^{\text{OVA}}$	52.95 ± 0.2	65.49 ± 0.2	71.86 ± 0.2	75.88 ± 0.2	78.72 ± 0.2	86.03 ± 0.2
Recall@2$^{\text{OVA}}$	52.80 ± 0.2	64.18 ± 0.2	68.81 ± 0.2	71.42 ± 0.2	73.17 ± 0.2	77.69 ± 0.3
Recall@3$^{\text{OVA}}$	40.50 ± 0.3	56.01 ± 0.2	64.96 ± 0.2	70.95 ± 0.2	75.26 ± 0.2	86.32 ± 0.2
Recall@4$^{\text{OVA}}$	46.59 ± 0.4	59.87 ± 0.6	66.77 ± 0.5	70.95 ± 0.4	73.75 ± 0.3	79.86 ± 0.2
Recall@5$^{\text{OVA}}$	50.72 ± 0.2	64.74 ± 0.3	70.75 ± 0.3	74.02 ± 0.3	76.06 ± 0.3	80.66 ± 0.2
Recall@10$^{\text{OVA}}$	50.92 ± 0.2	64.94 ± 0.2	70.95 ± 0.2	74.14 ± 0.2	76.21 ± 0.2	80.68 ± 0.2
top-1 SVM$^{\alpha}$	58.16 ± 0.2	71.66 ± 0.2	78.22 ± 0.1	82.29 ± 0.2	84.98 ± 0.2	91.48 ± 0.2
top-2 SVM$^{\alpha}$	58.81 ± 0.2	72.71 ± 0.2	79.33 ± 0.2	83.29 ± 0.2	85.94 ± 0.2	92.19 ± 0.2
top-3 SVM$^{\alpha}$	$\mathbf{58.97 \pm 0.1}$	73.19 ± 0.2	79.86 ± 0.2	83.83 ± 0.2	86.46 ± 0.2	92.57 ± 0.2
top-4 SVM$^{\alpha}$	58.95 ± 0.1	73.54 ± 0.2	80.25 ± 0.2	84.20 ± 0.2	86.78 ± 0.2	92.82 ± 0.2
top-5 SVM$^{\alpha}$	58.92 ± 0.1	$\mathbf{73.66 \pm 0.2}$	80.46 ± 0.2	84.44 ± 0.3	87.03 ± 0.2	92.98 ± 0.2
top-10 SVM$^{\alpha}$	58.00 ± 0.2	73.65 ± 0.1	$\mathbf{80.80 \pm 0.1}$	$\mathbf{84.81 \pm 0.2}$	$\mathbf{87.45 \pm 0.2}$	93.40 ± 0.2
top-20 SVM$^{\alpha}$	55.98 ± 0.3	72.51 ± 0.2	80.22 ± 0.2	84.54 ± 0.2	87.37 ± 0.2	$\mathbf{93.62 \pm 0.2}$
top-1 SVM$^{\beta}$	58.16 ± 0.2	71.66 ± 0.2	78.22 ± 0.1	82.29 ± 0.2	84.98 ± 0.2	91.48 ± 0.2
top-2 SVM$^{\beta}$	58.80 ± 0.2	72.65 ± 0.2	79.26 ± 0.2	83.21 ± 0.2	85.85 ± 0.2	92.14 ± 0.2
top-3 SVM$^{\beta}$	59.14 ± 0.2	73.21 ± 0.2	79.81 ± 0.2	83.77 ± 0.2	86.36 ± 0.2	92.51 ± 0.2
top-4 SVM$^{\beta}$	59.24 ± 0.1	73.58 ± 0.2	80.18 ± 0.2	84.15 ± 0.2	86.71 ± 0.2	92.73 ± 0.2
top-5 SVM$^{\beta}$	59.28 ± 0.2	73.78 ± 0.2	80.45 ± 0.3	84.36 ± 0.3	86.96 ± 0.3	92.93 ± 0.2
top-10 SVM$^{\beta}$	$\mathbf{59.32 \pm 0.1}$	$\mathbf{74.13 \pm 0.2}$	80.91 ± 0.2	84.92 ± 0.2	87.49 ± 0.2	93.36 ± 0.2
top-20 SVM$^{\beta}$	58.65 ± 0.2	73.96 ± 0.2	$\mathbf{80.95 \pm 0.2}$	$\mathbf{85.05 \pm 0.2}$	$\mathbf{87.70 \pm 0.2}$	$\mathbf{93.64 \pm 0.2}$

Table 5.4.: Top: Results from the literature. **Middle:** Our results using the baseline methods. **Bottom:** Our proposed top-k SVM.

results suggest that, in general, one has to tune the k parameter in top-k SVM independently of the k' in top-k' accuracy. We also see that top-k SVM$^{\beta}$ performs slightly better compared to top-k SVM$^{\alpha}$, although the differences are small.

On Places 205 and ImageNet 2012 (Table 5.5), we see that our method is one of the few that are scalable to large datasets with millions of training examples. We also observe that optimizing the top-k hinge loss (both α and β versions) yields consistently better top-k performance, and produces the familiar diagonal pattern of peak top-k results.

Overall, top-k SVM obtains systematic increase in top-k accuracy over the datasets that we examined. We discover a diagonal pattern of peak top-k performance suggesting that it is beneficial to optimize for the top-k error if that is the target metric. Moreover, top-k SVM with $k > 1$ may even improve the top-1 performance on datasets with a large number of classes and ambiguities in the ground truth annotation. The most promising results, however, are obtained in the top-k performance with $k > 1$.

	Places 205 (val)						**ImageNet 2012** (val)					
Method	Top-1	Top-2	Top-3	Top-4	Top-5	Top-10	Top-1	Top-2	Top-3	Top-4	Top-5	Top-10
ZLX / BVLC	50.0	-	-	-	81.1	-	**57.4**	-	-	-	**80.4**	-
TopPush$^{\mathrm{OVA}}$	38.45	47.33	53.25	57.29	60.30	69.91	55.49	68.05	**73.89**	**77.34**	79.72	**85.99**
top-1 SVM$^{\alpha}$	50.63	64.47	71.44	75.50	78.54	86.17	**56.61**	67.31	72.43	75.45	77.67	83.71
top-2 SVM$^{\alpha}$	51.05	65.74	73.10	77.49	80.74	88.43	56.60	68.09	73.25	76.36	78.62	84.55
top-3 SVM$^{\alpha}$	**51.31**	66.17	73.23	77.86	81.26	89.37	56.56	68.27	73.60	76.76	79.03	84.96
top-4 SVM$^{\alpha}$	51.24	**66.30**	73.48	78.08	81.40	89.74	56.52	68.36	73.80	77.06	79.30	85.25
top-5 SVM$^{\alpha}$	50.80	66.23	**73.67**	78.19	81.43	89.95	56.46	**68.40**	**73.85**	77.20	79.39	85.41
top-10 SVM$^{\alpha}$	50.10	65.76	73.38	**78.30**	**81.62**	**90.14**	55.89	68.16	73.80	**77.31**	**79.75**	85.77
top-20 SVM$^{\alpha}$	49.25	64.85	72.62	77.67	81.14	89.99	54.94	67.53	73.50	77.08	79.59	**85.88**
top-1 SVM$^{\beta}$	50.63	64.45	71.45	75.50	78.54	86.17	56.61	67.31	72.43	75.45	77.67	83.71
top-2 SVM$^{\beta}$	51.03	65.58	72.73	77.40	80.55	88.40	56.91	67.98	73.19	76.23	78.50	84.43
top-3 SVM$^{\beta}$	51.27	65.98	73.37	77.91	81.25	89.30	57.00	68.27	73.51	76.68	78.89	84.84
top-4 SVM$^{\beta}$	**51.38**	66.20	73.56	78.04	81.40	89.78	56.99	68.39	73.62	76.86	79.15	85.09
top-5 SVM$^{\beta}$	51.25	**66.25**	**73.66**	78.26	81.42	89.91	**57.09**	**68.45**	73.68	76.95	79.27	85.24
top-10 SVM$^{\beta}$	50.94	66.13	73.52	**78.36**	**81.69**	**90.19**	56.90	68.42	**73.95**	77.31	79.53	85.62
top-20 SVM$^{\beta}$	50.50	65.79	73.38	78.17	81.60	90.12	56.48	68.29	73.83	**77.32**	**79.60**	**85.81**

Table 5.5.: Top: Results from the literature. **Middle:** Our results using the baseline methods. **Bottom:** Our proposed top-k SVM.

5.6 Conclusion

In this chapter, we moved the focus from learning with limited training data towards a challenge that is particularly prominent in modern large scale learning – handling the increased *class ambiguity*. We argued that categorizing real world images by tagging them with a single label is suboptimal and leads to confusion that is often difficult to overcome even for humans. In the case of scene categorization, for example, the issue is evidently seen in Figure 5.1, which illustrates class ambiguity in the SUN 397 dataset. Looking at the images of a body of water, it can be difficult to decide whether they depict a river or a pond, as we only get to see a small part of the full scene. Furthermore, the same image may as well be labeled as 'park', or even a 'picnic area', since a real world scene is inherently multilabel. Finally, it is known that the ground truth annotation in large scale datasets, such as the ImageNet for example, contains a certain level of label noise. That issue is naturally the result of the great effort required to annotate millions of images, but may be even more pertinent to fine-grained categorization where the differences between the classes are more subtle and may require expert domain knowledge.

While it is interesting to consider long term solutions for the issues outlined above (e.g., by fixing the label noise, enriching the annotations, or using additional modalities with discriminative information), we have focused on a solution that is a reasonable first step and is directly applicable in modern benchmarks. Specifically, we argue that the *top-k error*, which allows one to make k guesses instead of one, is a natural target performance metric in the presence of class ambiguity. It is also a well recognized metric commonly used in popular benchmarks, such as the ImageNet and Places (Russakovsky et al., 2015; Zhou et al., 2014).

Our contributions are twofold. First, we proposed *top-k multiclass SVM* as a method to directly optimize for top-k performance. The underlying idea is to use a tight convex upper bound on the discrete top-k error as the surrogate loss in the training objective. We have explored two formulations for the top-k hinge loss and demonstrated consistent improvements in top-k performance on five image recognition datasets. Second, we implemented an efficient optimization scheme based on the SDCA framework of Shalev-Shwartz and Zhang, (2013b). Our algorithm is scalable to large datasets and can be used to train the proposed top-k SVM$^\alpha$, top-k SVM$^\beta$, as well as the well known multiclass SVM of Crammer and Singer, (2001). At the heart of the employed optimization method are efficient algorithms for the Euclidean and the biased projections onto the top-k simplex $\Delta_k(r)$ and the set $\tilde{\Delta}_k(r)$, which are of independent interest.

In the next chapter, we continue the exploration of top-k performance optimization and provide an in-depth study that broadens the scope along multiple directions, including the introduction of smooth top-k hinge losses, top-k extensions of the softmax loss, theoretical analysis of top-k calibration, and a study of *multilabel* classification in relation to multiclass and top-k classification.

6 Analysis and Optimization of Loss Functions for Multiclass, Top-k, and Multilabel Classification

In Chapter 5, we introduced the problem of *class ambiguity*, which is present in modern large scale image classification benchmarks, and proposed to consider the top-k error as the target performance measure. To optimize the top-k error, we formulated *top-k SVM* with two versions of the top-k hinge loss, and developed efficient optimization algorithms for them.

In this chapter, we extend the study of class ambiguity and top-k performance optimization. We recognize a connection that exists between **top-k multiclass** and **multilabel** classification, and organize the chapter to highlight that relationship. Our main goal is to provide an **in-depth analysis** of the established as well as the proposed classification methods, both multiclass and multilabel.

We start with an overview of the target performance measures that are accepted in the literature for multiclass and multilabel classification, and then consider the surrogate losses that are used in the training objectives. Here, we contribute a number of novel functions: smooth top-k hinge loss, top-k extensions of the softmax loss, and smooth multilabel SVM loss. We finally conclude with a theoretical analysis of top-k calibration for the multiclass methods.

To facilitate training, we develop efficient SDCA-based optimization schemes for the considered methods, where our technical contribution is a number of algorithms for computing SDCA updates. Here, we would like to highlight the entropic projections using the Lambert W function for the softmax loss and the Euclidean projection onto the *bipartite simplex* for the multilabel SVM.

Finally, we perform an extensive empirical evaluation on a wide range of multiclass and multilabel datasets which reveals a few interesting insights. First, our results indicate that the softmax loss and the smooth multiclass SVM are surprisingly competitive in top-k error uniformly across all k, which can be explained by our analysis of multiclass top-k calibration. Further improvements for a specific k are possible with the proposed top-k loss functions. And second, we use the top-k methods to explore the transition from multiclass to multilabel learning. Here, we find that it is possible to obtain effective multilabel classifiers using a single label per image for training, and that the gap between the multiclass and multilabel methods depends on label cardinality.

The material in this chapter is based on the following publications:

- M. Lapin, M. Hein, and B. Schiele (2016a). "Analysis and Optimization of Loss Functions for Multiclass, Top-k, and Multilabel Classification." In: *arXiv preprint arXiv:1612.03663* (submitted to PAMI).

- M. Lapin, M. Hein, and B. Schiele (2016b). “Loss Functions for Top-k Error: Analysis and Insights.” In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*

6.1 Introduction

Real world images are often multilabel in nature and maintaining a single label per image while increasing the number of classes generally amplifies class ambiguity. As we discussed in the previous chapter, the standard classification error becomes too stringent and is not a well suited performance metric in this case.

Allowing k guesses instead of one leads to what we call the *top-k error*, which is one of the main subjects of this part. While previous literature is focused on minimizing top-1 error, we consider $k \geq 1$. We are mainly interested in two cases: (i) achieving small top-k error for *all k* simultaneously; and (ii) minimization of a specific top-k error. These goals are pursued in the first part of the chapter which is concerned with single label multiclass classification.

Following the general theme of the previous chapter, we propose extensions of the established multiclass loss functions to address top-k error minimization and derive appropriate SDCA-based optimization schemes. Extending the material of Chapter 5, we introduce *smooth top-k SVM* and two top-k versions of the softmax loss: *top-k entropy* and *truncated top-k entropy*. We also analyze which of the multiclass methods are calibrated for the top-k error and perform an extensive empirical evaluation to better understand their benefits and limitations.

Moving forward, we see top-k classification as a natural transition step between multiclass learning with a single label per training example and multilabel learning with a complete set of relevant labels. Multilabel learning forms the second part of this chapter, where we introduce a smoothed version of the multilabel SVM loss of Crammer and Singer, (2003), and contribute two novel projection algorithms for efficient optimization of multilabel losses in the SDCA framework. Furthermore, we compare all multiclass, top-k, and multilabel methods in a novel experimental setting, where we want to quantify the utility of multilabel annotation. Specifically, we want to understand if it is possible to obtain effective multilabel classifiers from single label annotation.

The contributions of this chapter are summarized below.

- In § 6.2, we introduce the multiclass and multilabel learning problems, and discuss the respective performance metrics. We propose 4 novel loss functions for minimizing the top-k error and a *smooth* multilabel SVM loss. A brief summary of the methods that we consider is given in Table 6.1.

- In § 6.3, we introduce the notion of top-k calibration and analyze which of the multiclass methods are calibrated for the top-k error. In particular, we highlight that the softmax loss is uniformly top-k calibrated for all $k \geq 1$.

- In § 6.4, we develop efficient optimization schemes based on the SDCA framework. Specifically, we contribute a set of algorithms for computing

Method	Name	Loss function	Conjugate	Update	Top-k calibrated
SVM$^{\text{OVA}}$	One-vs-all (OVA) SVM	$\max\{0,\, 1 - yf(x)\}$	(Shalev-Shwartz and Zhang, 2014)		no† (Prop. 6.8)
LR$^{\text{OVA}}$	OVA logistic regression	$\log\big(1 + \exp(-yf(x))\big)$			yes (Prop. 6.9)
SVM$^{\text{Multi}}$	Multiclass SVM	$\max\big\{0, (a + c)_{\pi_1}\big\}$	Prop. 5.2	Prop. 5.5	no (Prop. 6.11)
LR$^{\text{Multi}}$	Softmax (cross entropy)	$\log\big(\sum_{j \in \mathcal{Y}} \exp(a_j)\big)$	Prop. 6.3	Prop. 6.15	yes (Prop. 6.12)
top-k SVM$^{\alpha}$	Top-k SVM (α)	$\max\big\{0, \frac{1}{k}\sum_{j=1}^{k}(a + c)_{\pi_j}\big\}$	Prop. 5.2	Prop. 5.5	open question
top-k SVM$^{\beta}$	Top-k SVM (β)	$\frac{1}{k}\sum_{j=1}^{k} \max\big\{0, (a + c)_{\pi_j}\big\}$	Prop. 5.3	Prop. 5.6	
top-k SVM$^{\alpha}_{\gamma}$	Smooth top-k SVM (α)	L_γ in Prop. 6.2 w/ Δ^{α}_{k}	Prop. 6.2	Prop. 6.14	
top-k SVM$^{\beta}_{\gamma}$	Smooth top-k SVM (β)	L_γ in Prop. 6.2 w/ Δ^{β}_{k}			
top-k Ent	Top-k entropy	L in Prop. 6.4	Prop. 6.3	Prop. 6.15	
top-k Ent$_{\text{tr}}$	Truncated top-k entropy	$\log\big(1 + \sum_{j \in \mathcal{J}^k_y} \exp(a_j)\big)$	n/a (nonconvex)		yes (Prop. 6.13)
SVM$^{\text{ML}}$	Multilabel SVM	$\max\limits_{y,\, \bar{y}} \max\{0, 1 + u_{\bar{y}} - u_y\}$	Prop. 6.5	Prop. 6.18	see, e.g., (Gao and Zhou, 2011)
SVM$^{\text{ML}}_{\gamma}$	Smooth multilabel SVM	L_γ in Prop. 6.6	Prop. 6.6	Prop. 6.18	
LR$^{\text{ML}}$	Multilabel Softmax	$\frac{1}{\lvert Y \rvert}\sum_y \log\big(\sum_{\bar{y}} e^{(u_{\bar{y}} - u_y)}\big)$	Prop. 6.7	Prop. 6.20	

Let $a \triangleq (f_j(x) - f_y(x))_{j \in \mathcal{Y}}$, $u \triangleq (f_y(x))_{y \in \mathcal{Y}}$, $c \triangleq \mathbf{1} - e_y$; π: $a_{\pi_1} \geq \ldots \geq a_{\pi_m}$; $\mathcal{J}^k_y$ is defined in § 6.2.2.

Note that SVM$^{\text{Multi}}$ $\equiv$ top-1 SVM$^{\alpha}$ $\equiv$ top-1 SVM$^{\beta}$ and LR$^{\text{Multi}}$ $\equiv$ top-1 Ent $\equiv$ top-1 Ent$_{\text{tr}}$.

† However, smooth SVM$^{\text{OVA}}_{\gamma}$ is top-k calibrated, see Prop. 6.10.

Table 6.1.: Overview of the methods considered in this chapter and our contributions.

the proximal maps that can be used to train classifiers with the specified multiclass, top-k, and multilabel loss functions.

- In § 6.5, the methods are evaluated empirically in three different settings: on synthetic data (§ 6.5.1), on multiclass datasets with a single label per example (§ 6.5.2), and on multilabel datasets (§ 6.5.3).

- In § 6.5.2, we perform a set of experiments on 11 multiclass benchmarks including the ImageNet 2012 (Russakovsky et al., 2015) and the Places 205 (Zhou et al., 2014) datasets. Our evaluation reveals, in particular, that the softmax loss and the proposed smooth SVM$^{\text{Multi}}_{\gamma}$ loss are competitive uniformly in all top-k errors, while improvements for a specific k can be obtained with the proposed top-k losses.

- In § 6.5.3, we evaluate the multilabel methods on 10 datasets following Madjarov et al., (2012), where our smooth multilabel SVM$^{\text{ML}}_{\gamma}$ shows particularly encouraging results. Next, we perform experiments on Pascal VOC 2007 (Everingham et al., 2010) and Microsoft COCO (Lin et al., 2014), where we train multiclass and top-k methods using only a single label of the most prominent object per image, and then compare their multilabel performance on test data to that of multilabel methods trained with full annotation. Surprisingly, we observe a gap of just above 2% mAP on Pascal VOC between the best multiclass and multilabel methods.

6.2 Loss Functions for Classification

When choosing a loss function, one may want to consider several aspects. First, at the basic level, the loss function depends on the available annotation and the performance metric one is interested in, e.g. we distinguish between (single label) multiclass and multilabel losses in this thesis. Next, there are two fundamental factors that control the *statistical* and the *computational* behavior of learning. For computational reasons, we work with convex surrogate losses rather than with the performance metric directly. In that context, a relevant distinction is between the nonsmooth Lipschitz functions ($\text{SVM}^{\text{Multi}}$, top-$k$ SVM) and the smooth functions (LR^{Multi}, $\text{SVM}^{\text{Multi}}_{\gamma}$, top-$k$ SVM_{γ}) with strongly convex conjugates that lead to faster convergence rates. From the statistical perspective it is important to understand if the surrogate loss is classification calibrated as this attractive asymptotic property leads to Bayes consistent classifiers. Finally, one may exploit duality and introduce modifications to the conjugates of existing functions that have desirable effects on the primal loss (top-k Ent).

The rest of this section covers the technical background that is used later in the chapter. We discuss our notation, introduce multiclass and multilabel classification, recall the standard approaches to classification, as well as our proposed top-k SVM from Chapter 5.

- In § 6.2.1, we discuss multiclass and multilabel performance evaluation measures that are used later in our experiments.
- In § 6.2.2, we review established multiclass approaches and introduce our novel top-k loss functions; we also recall Moreau-Yosida regularization as a smoothing technique and compute convex conjugates for SDCA.
- In § 6.2.3, we discuss multilabel classification methods, introduce the smooth multilabel SVM, and compute the corresponding convex conjugates.

Notation. We consider classification problems with a predefined set of m classes. We begin with *multiclass* classification, where every example $x_i \in \mathcal{X}$ has exactly *one* label $y_i \in \mathcal{Y} \triangleq \{1, \ldots, m\}$, and later generalize to the *multilabel* setting, where each example is associated with a *set* of labels $Y_i \subset \mathcal{Y}$. In this chapter, a classifier is a function $f : \mathcal{X} \to \mathbb{R}^m$ that induces a ranking of class labels via the prediction scores $f(x) = \big(f_y(x)\big)_{y \in \mathcal{Y}}$. In the linear case, each predictor f_y has the form $f_y(x) = \langle w_y, x \rangle$, where $w_y \in \mathbb{R}^d$ is the parameter to be learned. We stack the individual parameters into a weight matrix $W \in \mathbb{R}^{d \times m}$, so that $f(x) = W^\top x$. While we focus on linear classifiers with $\mathcal{X} \equiv \mathbb{R}^d$ in the exposition below and in most of our experiments, all loss functions are formulated in the general setting where the kernel trick (Schölkopf and Smola, 2002) can be employed to construct nonlinear decision surfaces. In fact, we have a number of experiments with the RBF kernel as well.

At test time, prediction depends on the evaluation metric and generally involves sorting / producing the top-k highest scoring class labels in the multiclass setting, and predicting the labels that score above a certain threshold δ in multilabel classification. We come back to performance metrics shortly.

We use π and τ to denote permutations of (indexes) $\mathcal{Y}$. Unless stated otherwise, a_π reorders components of a vector a in descending order, $a_{\pi_1} \geq a_{\pi_2} \geq \ldots \geq a_{\pi_m}$. Therefore, for example, $a_{\pi_1} = \max_j a_j$. If necessary, we make it clear which vector is being sorted by writing $\pi(a)$ to mean $\pi(a) \in \arg\operatorname{sort} a$ and let $\pi_{1:k}(a) \triangleq \{\pi_1(a), \ldots, \pi_k(a)\}$. We also use the Iverson bracket defined as $[\![P]\!] = 1$ if P is true and 0 otherwise; and introduce a shorthand for the conditional probability $p_y(x) \triangleq \Pr(Y = y \,|\, X = x)$. Finally, we let $a^{\setminus y}$ be obtained by removing the yth coordinate from a.

We consider ℓ_2-regularized objectives in this chapter, so that if $L : \mathcal{Y} \times \mathbb{R}^m \to \mathbb{R}_+$ is a multiclass loss and $\lambda > 0$ is a regularization parameter, classifier training amounts to solving

$$\min_{W \in \mathbb{R}^{d \times m}} \frac{1}{n} \sum_{i=1}^{n} L(y_i, W^\top x_i) + \frac{\lambda}{2} \|W\|_F^2 .$$

Binary and multilabel classification problems only differ in the loss L.

6.2.1 Performance Metrics

Here, we briefly review performance evaluation metrics employed in multiclass and multilabel classification.

Multiclass. A standard performance measure for classification problems is the zero-one loss, which simply counts the number of classification mistakes (Duda et al., 2012; Friedman et al., 2001). While that metric is well understood and inspired such popular surrogate losses as the SVM hinge loss, it naturally becomes more stringent as the number of classes increases. An alternative to the standard zero-one error is to allow k guesses instead of one, as discussed in § 5.2 (page 80). Formally, the **top-k zero-one loss** (**top-k error**) is

$$\operatorname{err}_k(y, f(x)) \triangleq [\![f_{\pi_k}(x) > f_y(x)]\!]. \tag{6.1}$$

That is, we count a mistake if the ground truth label y scores below k other class labels. Note that for $k = 1$ we recover the standard zero-one error. **Top-k accuracy** is defined as 1 minus the top-k error, and performance on the full test sample is computed as the mean across all test examples.

Multilabel. Several groups of multilabel evaluation metrics are established in the literature and it is generally suggested that multiple contrasting measures should be reported to avoid skewed results. Here, we give a brief overview of the metrics that we report and for further details refer the interested reader to (Koyejo et al., 2015; Madjarov et al., 2012; Zhang and Zhou, 2014).

Ranking based. This group of performance measures compares the ranking of the labels induced by $f_y(x)$ to the ground truth ranking. We report the **rank loss**

$$\mathrm{RLoss}(f) = \tfrac{1}{n}\textstyle\sum_{i=1}^{n} |D_i| / (|Y_i| \left|\bar{Y}_i\right|),$$

where $D_i = \{(y, \bar{y}) \mid f_y(x_i) \leq f_{\bar{y}}(x_i),\ (y, \bar{y}) \in Y_i \times \bar{Y}_i\}$ is the set of reversely ordered pairs, and $\bar{Y}_i \triangleq \mathcal{Y} \setminus Y_i$ is the complement of Y_i. This is the loss that is implicitly optimized by all multiclass / multilabel loss functions that we consider since they induce a penalty when $f_{\bar{y}}(x_i) - f_y(x_i) > 0$.

Ranking class labels for a given image is similar to ranking documents for a user query in information retrieval (Liu, 2009). While there are many established metrics (Manning et al., 2008), a popular measure that is relevant to our discussion is **precision-at-k** (P@k), which counts the fraction of relevant items within the top k retrieved (Joachims, 2005; McFee and Lanckriet, 2010). We have already encountered the precision-at-k metric in § 5.2.3 (page 86), where it was used for a *binary* problem obtained through a reduction scheme. Here, instead, we consider the multilabel setting and compute the precision-at-k on each example separately. Although this measure makes perfect sense when $k \ll |Y_i|$, i.e. there are many more relevant documents than we possibly want to examine, it is not very useful when there are only a few correct labels per image – once all the relevant labels are in the top k list, P@k starts to decrease as k increases. A better alternative in our multilabel setting is a complementary measure, **recall-at-k**, defined as

$$\mathrm{R@}k(f) = \tfrac{1}{n}\textstyle\sum_{i=1}^{n} \Big(\pi_{1:k}(f(x_i)) \cap |Y_i|\Big) / |Y_i|\,,$$

which measures the fraction of relevant labels in the top k list. Note that R@k is a natural generalization of the top-k error to the multilabel setting and coincides with that multiclass metric whenever Y_i is singleton.

Finally, we report the standard Pascal VOC (Everingham et al., 2010) performance measure, mean average precision (**mAP**), which is computed as the one-vs-all AP averaged over all classes.

Partition based. In contrast to ranking, partition based measures assess the quality of the actual multilabel prediction which requires a cut-off **threshold** $\delta \in \mathbb{R}$. Several threshold selection strategies have been proposed:

(i) setting a constant threshold prior to experiments (Dembczynski et al., 2010);

(ii) selecting a threshold *a posteriori* by matching label cardinality (Madjarov et al., 2012; Read et al., 2009);

(iii) tuning the threshold on a validation set (Koyejo et al., 2015; Yang, 1999);

(iv) learning a regression function (Elisseeff and Weston, 2001);

(v) bypassing threshold selection altogether by introducing a (dummy) calibration label (Fürnkranz et al., 2008).

We have experimented with options (ii) and (iii), as discussed in § 6.5.3.

Let $h(x) \triangleq \{y \in \mathcal{Y} \mid f_y(x) \geq \delta\}$ be the set of predicted labels for a given threshold δ, and let

$$\begin{aligned} \widehat{\text{TP}}_{i,j} &= [\![j \in h(x_i), j \in Y_i]\!], & \widehat{\text{TN}}_{i,j} &= [\![j \notin h(x_i), j \notin Y_i]\!], \\ \widehat{\text{FP}}_{i,j} &= [\![j \in h(x_i), j \notin Y_i]\!], & \widehat{\text{FN}}_{i,j} &= [\![j \notin h(x_i), j \in Y_i]\!], \end{aligned}$$

be a set of $m \cdot n$ primitives defined as in (Koyejo et al., 2015). One can use any performance measure Ψ that is based on the binary confusion matrix, but, depending on where the averaging occurs, the following three cases are recognized.

Instance-averaging. The binary metrics are computed on the averages over labels and then averaged across examples:

$$\Psi^{\text{inst}}(h) = \tfrac{1}{n} \textstyle\sum_{i=1}^{n} \Psi\Big(\tfrac{1}{m} \sum_{j=1}^{m} \widehat{\text{TP}}_{i,j}, \ldots, \tfrac{1}{m} \sum_{j=1}^{m} \widehat{\text{FN}}_{i,j} \Big).$$

Macro-averaging. The metrics are averaged across labels:

$$\Psi^{\text{mac}}(h) = \tfrac{1}{m} \textstyle\sum_{j=1}^{m} \Psi\Big(\tfrac{1}{n} \sum_{i=1}^{n} \widehat{\text{TP}}_{i,j}, \ldots, \tfrac{1}{n} \sum_{i=1}^{n} \widehat{\text{FN}}_{i,j} \Big).$$

Micro-averaging. The metric is applied on the averages over both labels and examples:

$$\Psi^{\text{mic}}(h) = \Psi\Big(\tfrac{1}{mn} \textstyle\sum_{i,j} \widehat{\text{TP}}_{i,j}, \ldots, \tfrac{1}{mn} \sum_{i,j} \widehat{\text{FN}}_{i,j} \Big).$$

Following Madjarov et al., (2012), we consider the $\mathbf{F_1}$ **score** as the binary metric Ψ with all three types of averaging. We also report multilabel **accuracy**, **subset accuracy**, and the **hamming loss** defined respectively as

$$\begin{aligned} \text{Acc}(h) &= \tfrac{1}{n} \textstyle\sum_{i=1}^{n} (\lvert h(x_i) \cap Y_i \rvert)/(\lvert h(x_i) \cup Y_i \rvert), \\ \text{SAcc}(h) &= \tfrac{1}{n} \textstyle\sum_{i=1}^{n} [\![h(x_i) = Y_i]\!], \\ \text{HLoss}(h) &= \tfrac{1}{mn} \textstyle\sum_{i=1}^{n} \lvert h(x_i) \triangle Y_i \rvert, \end{aligned}$$

where $\triangle$ is the symmetric set difference.

6.2.2 Multiclass Methods

In this section, we switch from performance evaluation at test time to how the quality of a classifier is measured during training. In particular, we introduce the loss functions that are used in established multiclass methods as well as our novel loss functions for optimizing the top-k error (6.1).

OVA. A multiclass problem is often solved using the one-vs-all (OVA) reduction to m independent binary classification problems. Every class is trained versus the rest which yields m classifiers $\{f_y\}_{y \in \mathcal{Y}}$. Typically, each classifier f_y is trained with a convex margin-based loss function $L(\tilde{y} f_y(x))$, where $L : \mathbb{R} \to \mathbb{R}_+$, $\tilde{y} = \pm 1$. Simplifying the notation, we consider

$$L(yf(x)) = \max\{0,\, 1 - yf(x)\}, \qquad (\mathrm{SVM}^{\mathrm{OVA}})$$

$$L(yf(x)) = \log(1 + e^{-yf(x)}). \qquad (\mathrm{LR}^{\mathrm{OVA}})$$

The hinge ($\mathrm{SVM}^{\mathrm{OVA}}$) and logistic ($\mathrm{LR}^{\mathrm{OVA}}$) losses correspond to the SVM and logistic regression methods respectively.

Multiclass. An alternative to the OVA scheme above is to use a *multiclass* loss $L : \mathcal{Y} \times \mathbb{R}^m \to \mathbb{R}_+$ directly. All multiclass losses that we consider only depend on pairwise differences between the ground truth score $f_y(x)$ and all the other scores $f_j(x)$. Loss functions from the SVM family additionally require a *margin* $\Delta(y, j)$, which can be interpreted as a distance in the label space (Tsochantaridis et al., 2005) between y and j. To simplify the notation, we use vectors a (for the differences) and c (for the margin) defined for a given (x, y) pair as

$$a_j \triangleq f_j(x) - f_y(x), \; c_j \triangleq 1 - [\![y = j]\!], \; j = 1, \ldots, m.$$

We also write $L(a)$ instead of the full $L(y, f(x))$.

We consider two generalizations of $\mathrm{SVM}^{\mathrm{OVA}}$ and $\mathrm{LR}^{\mathrm{OVA}}$:

$$L(a) = \max_{j \in \mathcal{Y}} \{a_j + c_j\}, \qquad (\mathrm{SVM}^{\mathrm{Multi}})$$

$$L(a) = \log\Big(\textstyle\sum_{j \in \mathcal{Y}} \exp(a_j)\Big). \qquad (\mathrm{LR}^{\mathrm{Multi}})$$

Both the multiclass SVM loss ($\mathrm{SVM}^{\mathrm{Multi}}$) of Crammer and Singer, (2001) and the softmax loss ($\mathrm{LR}^{\mathrm{Multi}}$) are common in multiclass problems.

The OVA and multiclass methods were designed with the goal of minimizing the standard error. Now, if we consider the top-k error (6.1) which does not penalize $(k-1)$ mistakes, we discover that convexity of the above losses leads to phenomena where $\mathrm{err}_k(y, f(x)) = 0$, but $L(y, f(x)) \gg 0$. That happens, e.g., when $f_{\pi_1}(x) \gg f_y(x) \geq f_{\pi_k}(x)$, and creates a bias if we are working with rigid function classes such as linear classifiers. Next, we introduce loss functions that are modifications of the above losses with the goal of alleviating that phenomenon.

Top-k SVM. We introduced top-k SVM in the previous chapter (page 79), where two modifications of the multiclass hinge loss ($\mathrm{SVM}^{\mathrm{Multi}}$) were proposed. The first version (α) is motivated directly by the top-k error while the second version (β) falls into a general family of ranking losses introduced earlier by Usunier et al., (2009). The two top-k SVM losses are

$$L(a) = \max\Big\{0, \tfrac{1}{k} \textstyle\sum_{j=1}^{k} (a + c)_{\pi_j}\Big\}, \qquad (\text{top-}k\ \mathrm{SVM}^{\alpha})$$

$$L(a) = \tfrac{1}{k} \textstyle\sum_{j=1}^{k} \max\Big\{0, (a + c)_{\pi_j}\Big\}, \qquad (\text{top-}k\ \mathrm{SVM}^{\beta})$$

where π reorders the components of $(a + c)$ in descending order. We have shown in § 5.2 that top-k SVM^{α} offers a tighter upper bound on the top-k error than top-k SVM$^\beta$. However, both losses perform similarly in our experiments with only a small advantage of top-k SVM$^\beta$ in some settings. Therefore, when the distinction is not important, we simply refer to them as the top-k hinge or the top-k SVM loss. Note that they both reduce to SVM$^{\text{Multi}}$ for $k = 1$.

Top-k SVM losses are not smooth which has implications for their optimization (§ 6.4) and top-k calibration (§ 6.3.1). Following Shalev-Shwartz and Zhang, (2014), who employed Moreau-Yosida regularization (Beck and Teboulle, 2012; Nesterov, 2005) to obtain a smoothed version of the binary hinge loss (SVM$^{\text{OVA}}$), we apply the same technique and introduce the smooth top-k SVM.

Moreau-Yosida regularization. We follow Parikh and Boyd, (2014) and give the main points here for completeness. The *Moreau envelope* or *Moreau-Yosida regularization* M_f of the function f is

$$M_f(v) \triangleq \inf_x \Big(f(x) + (1/2) \left\| x - v \right\|_2^2 \Big).$$

It is a smoothed or regularized form of f with the following nice properties: it is continuously differentiable on $\mathbb{R}^d$, even if f is not, and the sets of minimizers of f and M_f are the same[1]. To compute a smoothed top-k hinge loss, we use

$$M_f = \Big(f^* + (1/2) \left\| \cdot \right\|_2^2 \Big)^*,$$

where f^* is the convex conjugate of f. A classical result in convex analysis (Hiriart-Urruty and Lemaréchal, 2001) states that a conjugate of a strongly convex function has Lipschitz smooth gradient, therefore, M_f is indeed a smooth function.

Top-k hinge conjugate. Here, we recall the conjugates of the top-k hinge losses α and β. As shown in § 5.2, their effective domains are given by the **top-k simplex** (α and β respectively) of radius r defined as

$$\Delta_k^\alpha(r) \triangleq \Big\{ x \mid \langle \mathbf{1}, x \rangle \le r,\ 0 \le x_i \le \tfrac{1}{k} \langle \mathbf{1}, x \rangle\,,\ \forall i \Big\}, \tag{6.2}$$

$$\Delta_k^\beta(r) \triangleq \Big\{ x \mid \langle \mathbf{1}, x \rangle \le r,\ 0 \le x_i \le \tfrac{1}{k} r,\ \forall i \Big\}. \tag{6.3}$$

We let $\Delta_k^\alpha = \Delta_k^\alpha(1)$, $\Delta_k^\beta = \Delta_k^\beta(1)$, and note the relation $\Delta_k^\alpha \subset \Delta_k^\beta \subset \Delta$, where $\Delta = \Big\{ x \mid \langle \mathbf{1}, x \rangle \le 1,\ x_i \ge 0 \Big\}$ is the unit simplex and the inclusions are proper for $k > 1$, while for $k = 1$ all three sets coincide.

Proposition 6.1 (Prop. 5.2, page 85)**.** The convex conjugate of top-k SVM$^\alpha$ is

$$L^*(v) = \begin{cases} -\sum_{j \neq y} v_j & \text{if } \langle \mathbf{1}, v \rangle = 0 \text{ and } v^{\setminus y} \in \Delta_k^\alpha, \\ +\infty & \text{otherwise.} \end{cases}$$

The convex conjugate of top-k SVM$^\beta$ is defined in the same way, but with the set Δ_k^β instead of Δ_k^α.

[1] That does not imply that we get the same classifiers since we are minimizing a regularized sum of individually smoothed loss terms.

Note that the conjugates of both top-k SVM losses coincide and are equal to the conjugate of the SVM$^{\text{Multi}}$ loss with the exception of their effective domains, which are Δ_k^α, Δ_k^β, and Δ respectively. Recall that the effective domain of the conjugate loss is the feasible set for the dual variables. Therefore, as we move from SVM$^{\text{Multi}}$ to top-k SVM$^\beta$, to top-k SVM$^\alpha$, we introduce more and more constraints on the dual variables thus limiting the extent to which a single training example can influence the classifier.

Smooth top-k SVM. We apply the smoothing technique introduced above to top-k SVM$^\alpha$. Smoothing of top-k SVM$^\beta$ is done similarly, but the set $\Delta_k^\alpha(r)$ is replaced with $\Delta_k^\beta(r)$.

Proposition 6.2. Let $\gamma > 0$ be the smoothing parameter. The smooth top-k hinge loss (α) and its conjugate are

$$L_\gamma(a) = \tfrac{1}{\gamma}\Big(\langle (a+c)^{\setminus y}, p\rangle - \tfrac{1}{2}\|p\|^2\Big), \qquad \text{(top-}k\text{ SVM}_\gamma^\alpha)$$

$$L_\gamma^*(v) = \begin{cases} \frac{\gamma}{2}\|v^{\setminus y}\|^2 - \langle v^{\setminus y}, c^{\setminus y}\rangle & \text{if } \langle \mathbf{1}, v\rangle = 0,\ v^{\setminus y} \in \Delta_k^\alpha, \\ +\infty & \text{otherwise,} \end{cases}$$

where $p = \mathbf{proj}_{\Delta_k^\alpha(\gamma)}(a+c)^{\setminus y}$ is the Euclidean projection of $(a+c)^{\setminus y}$ onto $\Delta_k^\alpha(\gamma)$. Moreover, $L_\gamma(a)$ is $1/\gamma$-smooth.

Proof. We take the convex conjugate of the top-k hinge loss, which was derived in Proposition 5.2, and add a regularizer $\frac{\gamma}{2}\langle v, v\rangle$ to obtain the γ-strongly convex conjugate loss $L_\gamma^*(v)$. Note that since $v_y = -\sum_{j\neq y} v_j$ and $a_y = f_y(x) - f_y(x) = 0$, we only need to work with $(m-1)$-dimensional vectors where the yth coordinate is removed. The primal loss $L_\gamma(a)$, obtained as the convex conjugate of $L_\gamma^*(v)$, is $1/\gamma$-smooth due to a known result in convex analysis (Hiriart-Urruty and Lemaréchal, 2001) (see also (Shalev-Shwartz and Zhang, 2014, Lemma 2)). We now derive a formula to compute it based on the Euclidean projection onto the top-k simplex. By definition,

$$\begin{aligned} L_\gamma(a) = \sup_{v' \in \mathbb{R}^m} \{\langle a, v'\rangle - L_\gamma^*(v')\} &= \max_{v \in \Delta_k^\alpha(1)} \Big\{\langle a^{\setminus y}, v\rangle - \tfrac{\gamma}{2}\langle v, v\rangle + \langle v, c^{\setminus y}\rangle\Big\} \\ &= -\min_{v \in \Delta_k^\alpha(1)} \Big\{\tfrac{\gamma}{2}\langle v, v\rangle - \langle (a+c)^{\setminus y}, v\rangle\Big\} \\ &= -\tfrac{1}{\gamma}\min_{\frac{v}{\gamma} \in \Delta_k^\alpha(1)} \Big\{\tfrac{1}{2}\langle v, v\rangle - \langle (a+c)^{\setminus y}, v\rangle\Big\}. \end{aligned}$$

For the constraint $\frac{v}{\gamma} \in \Delta_k^\alpha(1)$, we have

$$\begin{aligned} \langle \mathbf{1}, v/\gamma\rangle \leq 1, &\qquad 0 \leq v_i/\gamma \leq \tfrac{1}{k}\langle \mathbf{1}, v/\gamma\rangle \iff \\ \langle \mathbf{1}, v\rangle \leq \gamma, &\qquad 0 \leq v_i \leq \tfrac{1}{k}\langle \mathbf{1}, v\rangle \iff v \in \Delta_k^\alpha(\gamma). \end{aligned}$$

The final expression follows from the fact that

$$\operatorname*{arg\,min}_{v \in \Delta_k^\alpha(\gamma)} \left\{ \tfrac{1}{2} \langle v, v \rangle - \langle (a+c)^{\setminus y}, v \rangle \right\} \equiv \operatorname*{arg\,min}_{v \in \Delta_k^\alpha(\gamma)} \| (a+c)^{\setminus y} - v \|^2 \equiv \mathbf{proj}_{\Delta_k^\alpha(\gamma)}(a+c)^{\setminus y}.$$

□

While there is no analytic formula for the top-k SVM$_\gamma^\alpha$ loss, it can be computed efficiently via the projection onto the top-k simplex (§ 5.4, page 94). We can also compute its gradient as

$$\nabla L_\gamma(a) = (1/\gamma) \big(\mathbf{I}_y - e_y \mathbf{1}_y^\top \big) \, \mathbf{proj}_{\Delta_k^\alpha(\gamma)}(a+c)^{\setminus y},$$

where $\mathbf{I}_y$ is the identity matrix w/o the yth column, e_y is the yth standard basis vector, and $\mathbf{1}_y$ is the $(m-1)$-dimensional vector of all ones. This follows from the definition of a, the fact that $L_\gamma(a)$ can be written as $\frac{1}{2\gamma}(\|x\|^2 - \|x-p\|^2)$ for $x = (a+c)^{\setminus y}$ and $p = \mathbf{proj}_{\Delta_k^\alpha(\gamma)}(x)$, and a known result

$$\nabla_x \tfrac{1}{2} \| x - \mathbf{proj}_C(x) \|^2 = x - \mathbf{proj}_C(x),$$

which holds for any closed convex set C, see Proposition A.5 (page 186).

Smooth multiclass SVM (SVM$_\gamma^{\text{Multi}}$). We also highlight an important special case of top-k SVM$_\gamma^\alpha$ that performed remarkably well in our experiments. It is a smoothed version of SVM$^{\text{Multi}}$ and is obtained with $k = 1$ and $\gamma > 0$.

Softmax conjugate. Before we introduce a top-k version of the softmax loss (LR$^{\text{Multi}}$), we need to recall its conjugate.

Proposition 6.3. The convex conjugate of the LR$^{\text{Multi}}$ loss is

$$L^*(v) = \begin{cases} \sum_{j \neq y} v_j \log v_j + (1 + v_y) \log(1 + v_y) & \text{if } \langle \mathbf{1}, v \rangle = 0 \text{ and } v^{\setminus y} \in \Delta, \\ +\infty & \text{otherwise,} \end{cases} \tag{6.4}$$

where $\Delta = \big\{ x \,|\, \langle \mathbf{1}, x \rangle \leq 1, \; x_j \geq 0 \big\}$ is the unit simplex.

Proof. Here, we use the notation $u \triangleq f(x)$ as we need to take special care of the differences $f_j(x) - f_y(x)$ when computing the conjugate. The softmax loss is

$$L(u) = \log \Big(\textstyle\sum_{j \in \mathcal{Y}} \exp(u_j - u_y) \Big) = \log \Big(\textstyle\sum_{j \in \mathcal{Y}} \exp(a_j) \Big),$$

where $a = H_y u$ as before and $H_y \triangleq \mathbf{I} - \mathbf{1} e_y^\top$. Define

$$\phi(u) \triangleq \log \Big(\textstyle\sum_{j \in \mathcal{Y}} \exp(u_j) \Big),$$

then $L(u) = \phi(H_y u)$ and the conjugate is computed similar to Lemma 5.2:

$$\begin{aligned} L^*(v) &= \sup\{\, \langle u, v \rangle - L(u) \,|\, u \in \mathbb{R}^m \} \\ &= \sup\{\, \langle u, v \rangle - \phi(H_y u) \,|\, u \in \mathbb{R}^m \} \\ &= \sup\{ \langle u^{\parallel}, v \rangle + \langle u^{\perp}, v \rangle - \phi(H_y u^{\perp}) \,|\, u^{\parallel} \in \operatorname{Ker} H_y, u^{\perp} \in \operatorname{Ker}^{\perp} H_y \}, \end{aligned}$$

where $\operatorname{Ker} H_y = \{u \,|\, H_y u = 0\} = \{t\mathbf{1} \,|\, t \in \mathbb{R}\}$ and $\operatorname{Ker}^{\perp} H_y = \{u \,|\, \langle \mathbf{1}, u \rangle = 0\}$. It follows that $L^*(v)$ can only be finite if $\langle u^{\parallel}, v \rangle = 0$, which implies

$$v \in \operatorname{Ker}^{\perp} H_y \iff \langle \mathbf{1}, v \rangle = 0.$$

Let $H_y^{\dagger}$ be the Moore-Penrose pseudoinverse of H_y. For a $v \in \operatorname{Ker}^{\perp} H_y$, we write

$$\begin{aligned} L^*(v) &= \sup\{ \langle H_y^{\dagger} H_y u^{\perp}, v \rangle - \phi(H_y u^{\perp}) \,|\, u^{\perp} \} \\ &= \sup\{ \langle z, (H_y^{\dagger})^{\top} v \rangle - \phi(z) \,|\, z \in \operatorname{Im} H_y \}, \end{aligned}$$

where $\operatorname{Im} H_y = \{H_y u \,|\, u \in \mathbb{R}^m\} = \{u \,|\, u_y = 0\}$. Using rank-1 update of the pseudoinverse (Petersen, Pedersen, et al., 2008, § 3.2.7), we have

$$(H_y^{\dagger})^{\top} = \mathbf{I} - e_y e_y^{\top} - \frac{1}{m} (\mathbf{1} - e_y) \mathbf{1}^{\top},$$

which together with $\langle \mathbf{1}, v \rangle = 0$ implies $(H_y^{\dagger})^{\top} v = v - v_y e_y$.

$$\begin{aligned} L^*(v) &= \sup\{ \langle u, v - v_y e_y \rangle - \phi(u) \,|\, u_y = 0 \} \\ &= \sup \Big\{ \langle u^{\setminus y}, v^{\setminus y} \rangle - \log \Big(1 + \textstyle\sum_{j \neq y} \exp(u_j) \Big) \Big\}. \end{aligned}$$

The function inside sup is concave and differentiable, hence the global optimum is at the critical point. Setting the partial derivatives to zero yields

$$v_j = \exp(u_j) / \Big(1 + \textstyle\sum_{j \neq y} \exp(u_j) \Big)$$

for $j \neq y$, from which we conclude, similar to (Shalev-Shwartz and Zhang, 2014, § 5.1), that $\langle \mathbf{1}, v \rangle \leq 1$ and $0 \leq v_j \leq 1$ for all $j \neq y$, i.e. $v^{\setminus y} \in \Delta$. Let

$$Z \triangleq \sum_{j \neq y} \exp(u_j),$$

we have at the optimum

$$u_j = \log(v_j) + \log(1 + Z), \quad \forall j \neq y.$$

Since $\langle \mathbf{1}, v \rangle = 0$, we also have that $v_y = -\sum_{j \neq y} v_j$, hence

$$\begin{aligned} L^*(v) &= \textstyle\sum_{j \neq y} u_j v_j - \log(1+Z) \\ &= \textstyle\sum_{j \neq y} v_j \log(v_j) + \log(1+Z)\Big(\sum_{j \neq y} v_j - 1\Big) \\ &= \textstyle\sum_{j \neq y} v_j \log(v_j) - \log(1+Z)(1+v_y). \end{aligned}$$

Summing v_j and using the definition of Z,

$$\textstyle\sum_{j \neq y} v_j = \sum_{j \neq y} e^{u_j} / \Big(1 + \sum_{j \neq y} e^{u_j}\Big) = Z/(1+Z).$$

Therefore,

$$1 + Z = 1/\Big(1 - \textstyle\sum_{j \neq y} v_j\Big) = 1/(1+v_y),$$

which finally yields

$$L^*(v) = \textstyle\sum_{j \neq y} v_j \log(v_j) + \log(1+v_y)(1+v_y),$$

if $\langle \mathbf{1}, v \rangle = 0$ and $v^{\setminus y} \in \Delta$ as stated in the proposition. □

Note that the conjugates of both the SVM$^{\text{Multi}}$ and the LR$^{\text{Multi}}$ losses share the same effective domain, the unit simplex Δ, and differ only in their functional form: a linear function for SVM$^{\text{Multi}}$ and a negative entropy for LR$^{\text{Multi}}$. While we motivated top-k SVM directly from the top-k error, we see that the only change compared to SVM$^{\text{Multi}}$ was in the effective domain of the conjugate loss. This suggests a general way to *construct novel losses* with specific properties by taking the conjugate of an existing loss function, and modifying its effective domain in a way that enforces the desired properties. The motivation for doing so comes from the interpretation of the dual variables as forces with which every training example pushes the decision surface in the direction given by the ground truth label. Therefore, by reducing the feasible set we can limit the maximal contribution of any given training example.

Top-k entropy. As hinted above, we first construct the conjugate of the top-k entropy loss (α) by taking the conjugate of LR$^{\text{Multi}}$ and replacing Δ in (6.4) with Δ_k^α, and then take the conjugate again to obtain the primal loss top-k Ent. A β version can be constructed using the set Δ_k^β instead.

Proposition 6.4. The top-k entropy loss is defined as

$$\begin{aligned} L(a) = \max\Big\{ &\langle a^{\setminus y}, x \rangle - (1-s)\log(1-s) \\ &- \langle x, \log x \rangle \mid x \in \Delta_k^\alpha,\ \langle \mathbf{1}, x \rangle = s \Big\}. \end{aligned} \tag{top-k Ent}$$

Moreover, we recover the LR$^{\text{Multi}}$ loss when $k = 1$.

Proof. The convex conjugate of the top-k entropy loss is

$$L^*(v) \triangleq \begin{cases} \sum_{j \neq y} v_j \log v_j + (1 + v_y) \log(1 + v_y) & \text{if } \langle \mathbf{1}, v \rangle = 0 \text{ and } v^{\setminus y} \in \Delta_k^\alpha, \\ +\infty & \text{otherwise.} \end{cases}$$

The (primal) top-k entropy loss is defined as the convex conjugate of the $L^*(v)$ above. We have

$$\begin{aligned} L(a) &= \sup\{ \langle a, v \rangle - L^*(v) \mid v \in \mathbb{R}^m \} \\ &= \sup\{ \langle a, v \rangle - \textstyle\sum_{j \neq y} v_j \log v_j - (1 + v_y) \log(1 + v_y) \mid \langle \mathbf{1}, v \rangle = 0,\ v^{\setminus y} \in \Delta_k^\alpha \} \\ &= \sup\{ \langle a^{\setminus y}, v^{\setminus y} \rangle - a_y \textstyle\sum_{j \neq y} v_j - \sum_{j \neq y} v_j \log v_j \\ &\qquad - (1 - \textstyle\sum_{j \neq y} v_j) \log(1 - \sum_{j \neq y} v_j) \mid v^{\setminus y} \in \Delta_k^\alpha \}. \end{aligned}$$

Note that $a_y = 0$, and hence the corresponding term vanishes. Finally, we let $x \triangleq v^{\setminus y}$ and $s \triangleq \sum_{j \neq y} v_j = \langle \mathbf{1}, x \rangle$.

Next, we discuss how this problem can be solved and show that it reduces to the softmax loss for $k = 1$. Let $a \triangleq a^{\setminus y}$ and consider an equivalent problem below.

$$L(a) = -\min \Big\{ \langle x, \log x \rangle + (1 - s) \log(1 - s) - \langle a, x \rangle \mid x \in \Delta_k^\alpha,\ \langle \mathbf{1}, x \rangle = s \Big\}. \tag{6.5}$$

The Lagrangian for (6.5) is

$$\begin{aligned} \mathcal{L}(x, s, t, \lambda, \mu, \nu) = \langle x, \log x \rangle + (1 - s) \log(1 - s) - \langle a, x \rangle \\ +t(\langle \mathbf{1}, x \rangle - s) + \lambda(s - 1) - \langle \mu, x \rangle + \left\langle \nu, x - \tfrac{s}{k} \mathbf{1} \right\rangle, \end{aligned}$$

where $t \in \mathbb{R}$ and $\lambda, \mu, \nu \geq 0$ are the dual variables. Computing the partial derivatives of $\mathcal{L}$ w.r.t. x_j and s, and setting them to zero, we obtain

$$\begin{aligned} \log x_j &= a_j - 1 - t + \mu_j - \nu_j, \quad \forall j \\ \log(1 - s) &= -1 - t - \tfrac{1}{k} \langle \mathbf{1}, \nu \rangle + \lambda. \end{aligned}$$

Note that $x_j = 0$ and $s = 1$ cannot satisfy the above conditions for any choice of the dual variables in $\mathbb{R}$. Therefore, $x_j > 0$ and $s < 1$, which implies $\mu_j = 0$ and $\lambda = 0$. The only constraint that might be active is $x_j \leq \frac{s}{k}$. Note, however, that in view of $x_j > 0$ it can only be active if either $k > 1$ or we have a one dimensional problem. We consider the case when this constraint is active below.

Consider x_j's for which $0 < x_j < \frac{s}{k}$ holds at the optimum. The complementary slackness conditions imply that the corresponding $\mu_j = \nu_j = 0$. Let $p \triangleq \langle \mathbf{1}, \nu \rangle$ and re-define t as $t \leftarrow 1 + t$. We obtain the simplified equations

$$\begin{aligned} \log x_j &= a_j - t, \\ \log(1 - s) &= -t - \tfrac{p}{k}. \end{aligned}$$

If $k = 1$, then $0 < x_j < s$ for all j in a multiclass problem as discussed above, hence also $p = 0$. We have

$$x_j = e^{a_j - t}, \qquad\qquad 1 - s = e^{-t},$$

where $t \in \mathbb{R}$ is to be found. Plugging that into the objective,

$$\begin{aligned} -L(a) &= \textstyle\sum_j (a_j - t) e^{a_j - t} - t e^{-t} - \sum_j a_j e^{a_j - t} = e^{-t} \Big[\sum_j (a_j - t) e^{a_j} - t - \sum_j a_j e^{a_j} \Big] \\ &= -t e^{-t} \Big[1 + \textstyle\sum_j e^{a_j} \Big] = -t \Big[e^{-t} + \sum_j e^{a_j - t} \Big] = -t \Big[1 - s + s \Big] = -t. \end{aligned}$$

To compute t, we note that

$$\textstyle\sum_j e^{a_j - t} = \langle \mathbf{1}, x \rangle = s = 1 - e^{-t},$$

from which we conclude

$$1 = \Big(1 + \textstyle\sum_j e^{a_j} \Big) e^{-t} \implies -t = -\log(1 + \sum_j e^{a_j}).$$

Taking into account the minus in front of the min in (6.5) and the definition of a, we finally recover the softmax loss

$$L(y, f(x)) = \log \Big(1 + \textstyle\sum_{j \neq y} \exp(f_j(x) - f_y(x)) \Big).$$

□

While there is no closed-form solution for the top-k Ent loss when $k > 1$, we can compute and optimize it efficiently as we discuss later in § 6.4.

Truncated top-k entropy. A major limitation of the softmax loss for top-k error optimization is that it cannot ignore the $(k - 1)$ highest scoring predictions. This can lead to a situation where the loss is high even though the top-k error is zero. To see that, let us rewrite the $\mathrm{LR}^{\mathrm{Multi}}$ loss as

$$L(y, f(x)) = \log \Big(1 + \textstyle\sum_{j \neq y} \exp(f_j(x) - f_y(x)) \Big). \tag{6.6}$$

If there is only a *single* j such that $f_j(x) - f_y(x) \gg 0$, then $L(y, f(x)) \gg 0$ even though $\mathrm{err}_2(y, f(x))$ is zero.

This problem is also present in all top-k hinge losses considered above and is an inherent limitation due to their convexity. The origin of the problem is the fact that ranking based losses (Usunier et al., 2009) are based on functions such as

$$\phi(f(x)) = (1/m) \textstyle\sum_{j \in \mathcal{Y}} \alpha_j f_{\pi_j}(x) - f_y(x).$$

The function ϕ is convex if the sequence (α_j) is monotonically nonincreasing (Boyd and Vandenberghe, 2004). This implies that convex ranking based losses have to put *more* weight on the highest scoring classifiers, while we would like to put *less* weight on them. To that end, we drop the first $(k - 1)$ highest scoring predictions from the sum in (6.6), sacrificing convexity of the loss, and define the truncated top-k entropy loss as follows

$$L(a) = \log\Big(1 + \textstyle\sum_{j \in \mathcal{J}_y^k} \exp(a_j)\Big), \qquad (\text{top-}k\ \text{Ent}_{\text{tr}})$$

where $\mathcal{J}_y^k$ are the indexes corresponding to the $(m-k)$ *smallest* components of $(f_j(x))_{j \neq y}$. This loss can be seen as a smooth version of the top-k error (6.1), as it is small whenever the top-k error is zero. We show a synthetic experiment in § 6.5.1, where the advantage of discarding the highest scoring classifier in top-k Ent$_{\text{tr}}$ becomes apparent.

6.2.3 Multilabel Methods

In this section, we introduce natural extensions of the classical multiclass methods discussed above to the setting where there is a *set* of ground truth labels $Y \subset \mathcal{Y}$ for each example x. We focus on the loss functions that produce a ranking of labels and optimize a multilabel loss $L : 2^{\mathcal{Y}} \times \mathbb{R}^m \to \mathbb{R}_+$. We let $u \triangleq f(x)$ and use a simplified notation $L(u) = L(Y, f(x))$. A more complete overview of multilabel classification methods is given in (Madjarov et al., 2012; Tsoumakas and Katakis, 2007; Zhang and Zhou, 2014).

Binary relevance (BR). Binary relevance is the standard one-vs-all scheme applied to multilabel classification. It is the default baseline for direct multilabel methods as it does not consider possible correlations between the labels.

Multilabel SVM. We follow the line of work by Crammer and Singer, (2003) and consider the multilabel SVM loss below:

$$\begin{aligned} L(u) &= \max_{y \in Y} \max_{j \in \bar{Y}} \max\{0, 1 + u_j - u_y\} \\ &= \max\{0, 1 + \max_{j \in \bar{Y}} u_j - \min_{y \in Y} u_y\}. \end{aligned} \qquad (\text{SVM}^{\text{ML}})$$

This method is also known as the *multiclass multilabel perceptron* (Fürnkranz et al., 2008) and the *separation ranking loss* (Guo and Schuurmans, 2011). It can be contrasted with another SVM$^{\text{Multi}}$ extension, the RankSVM of Elisseeff and Weston, (2001), which optimizes the *pairwise ranking loss*:

$$\tfrac{1}{|Y_i|\left|\bar{Y}_i\right|} \textstyle\sum_{(y,j) \in Y \times \bar{Y}} \max\{0, 1 + u_j - u_y\}.$$

Note that both the SVM$^{\text{ML}}$ that we consider and RankSVM avoid expensive enumeration of all the $2^{\mathcal{Y}}$ possible labellings by considering only pairwise label ranking. A principled large margin approach that accounts for all possible label interactions is structured output prediction (Tsochantaridis et al., 2005).

Multilabel SVM conjugate. Now, we compute the convex conjugate of the SVM$^{\text{ML}}$ loss which is used later to define a smooth multilabel SVM. Note that the SVM$^{\text{ML}}$ loss depends on the partitioning of $\mathcal{Y}$ into Y and $\bar{Y}$ for every given (x, Y) pair.

This is reflected in the definition of a set S_Y below, which is the effective domain of the conjugate:

$$S_Y \triangleq \Big\{x \,|\, -\textstyle\sum_{y\in Y} x_y = \sum_{j\in\bar{Y}} x_j \leq 1,\ x_y \leq 0,\ x_j \geq 0\Big\}.$$

In the multiclass setting, the set Y is singleton, therefore $x_y = -\sum_{j\in\bar{Y}} x_j$ has no degrees of freedom and we recover the unit simplex Δ over (x_j), as in (6.4). In the true multilabel setting, on the other hand, there is freedom to distribute the weight across all the classes in Y.

Proposition 6.5. The convex conjugate of the SVM$^{\text{ML}}$ loss is

$$L^*(v) = \begin{cases} -\sum_{j\in\bar{Y}} v_j & \text{if } v \in S_Y, \\ +\infty & \text{otherwise.} \end{cases} \tag{6.7}$$

Proof. We compute the convex conjugate of SVM$^{\text{ML}}$ as

$$L^*(v) = -\inf_{u\in\mathbb{R}^m} \{\max\{0, 1 + \max_{j\in\bar{Y}} u_j - \min_{y\in Y} u_y\} - \langle u, v\rangle\}.$$

When the infimum is attained, the conjugate can be computed by solving the following optimization problem, otherwise the conjugate is $+\infty$. The corresponding dual variables are given on the right.

$$\begin{aligned} \min_{u,\alpha,\beta,\xi}\ & \xi - \langle u, v\rangle & \\ & \xi \geq 1 + \beta - \alpha, & (\lambda \geq 0) \\ & \xi \geq 0, & (\mu \geq 0) \\ & \alpha \leq u_y, \quad \forall\, y \in Y, & (\nu_y \geq 0) \\ & \beta \geq u_j, \quad \forall\, j \in \bar{Y}. & (\eta_j \geq 0) \end{aligned}$$

The Lagrangian is given as

$$\begin{aligned} \mathcal{L}(u, \alpha, \beta, \xi, \lambda, \mu, \nu, \eta) = \xi - \langle u, v\rangle + \lambda(1 + \beta - \alpha - \xi) \\ -\mu\xi + \textstyle\sum_{y\in Y} \nu_y(\alpha - u_y) + \sum_{j\in\bar{Y}} \eta_j(u_j - \beta). \end{aligned}$$

Computing the partial derivatives and setting them to zero,

$$\begin{aligned} \partial_{u_y}\mathcal{L} &= -v_y - \nu_y, & \nu_y &= -v_y, & \forall\, y \in Y, \\ \partial_{u_j}\mathcal{L} &= -v_j + \eta_j, & \eta_j &= v_j, & \forall\, j \in \bar{Y}, \\ \partial_{\alpha}\mathcal{L} &= -\lambda + \langle \mathbf{1}, \nu\rangle, & \lambda &= \langle \mathbf{1}, \nu\rangle, & \\ \partial_{\beta}\mathcal{L} &= \lambda - \langle \mathbf{1}, \eta\rangle, & \lambda &= \langle \mathbf{1}, \eta\rangle, & \\ \partial_{\xi}\mathcal{L} &= 1 - \lambda - \mu, & \lambda &= 1 - \mu. & \end{aligned}$$

After a basic derivation, we arrive at the solution of the dual problem given by

$$\lambda = -\textstyle\sum_{y\in Y} v_y = \sum_{j\in\bar{Y}} v_j,$$

where v must be in the following feasible set S_Y:

$$\begin{aligned} S_Y \triangleq \Big\{ & v \in \mathbb{R}^m \mid -\textstyle\sum_{y\in Y} v_y = \sum_{j\in\bar{Y}} v_j \le 1, \\ & v_y \le 0,\ v_j \ge 0,\ \forall\, y \in Y,\ \forall\, j \in \bar{Y} \Big\}. \end{aligned}$$

To complete the proof, note that $L^*(v) = -\lambda$ if $v \in S_Y$. □

Note that when $|Y| = 1$, (6.7) naturally reduces to the conjugate of SVM$^{\text{Multi}}$ given in Proposition 6.1 with $k = 1$.

Smooth multilabel SVM. Here, we apply the smoothing technique, which worked very well for multiclass problems to the multilabel SVM$^{\text{ML}}$ loss.

As with the smooth top-k SVM, there is no analytic formula for the smoothed loss. However, we can both compute and optimize it within our framework by solving the Euclidean projection problem onto what we call a **bipartite simplex**. It is a convenient modification of the set S_Y above:

$$B(r) \triangleq \{(x, y) \mid \langle \mathbf{1}, x\rangle = \langle \mathbf{1}, y\rangle \le r, x \in \mathbb{R}^m_+, y \in \mathbb{R}^n_+\}. \tag{6.8}$$

Proposition 6.6. Let $\gamma > 0$ be the smoothing parameter. The smooth multilabel SVM loss and its conjugate are

$$L_\gamma(u) = \tfrac{1}{\gamma}\Big(\langle b, p\rangle - \tfrac{1}{2}\|p\|^2 + \big\langle \bar{b}, \bar{p}\big\rangle - \tfrac{1}{2}\|\bar{p}\|^2 \Big), \tag{SVM$^{\text{ML}}_\gamma$}$$

$$L^*_\gamma(v) = \begin{cases} \tfrac{1}{2}\Big(\sum_{y\in Y} v_y - \sum_{j\in\bar{Y}} v_j \Big) + \tfrac{\gamma}{2}\|v\|^2, & v \in S_Y, \\ +\infty, & \text{otherwise,} \end{cases}$$

where $(p, \bar{p}) = \mathbf{proj}_{B(\gamma)}(b, \bar{b})$ is the projection onto $B(\gamma)$ of $b = \big(\tfrac{1}{2} - u_y\big)_{y\in Y}$, $\bar{b} = \big(\tfrac{1}{2} + u_j\big)_{j\in\bar{Y}}$. $L_\gamma(u)$ is $1/\gamma$-smooth.

Proof. The convex conjugate of the SVM$^{\text{ML}}$ loss is

$$L^*(v) = \begin{cases} \sum_{y\in Y} v_y, & \text{if } v \in S_Y, \\ +\infty, & \text{otherwise.} \end{cases}$$

Before we add $\tfrac{\gamma}{2}\|v\|^2$, recall that $\sum_{y\in Y} v_y = -\sum_{j\in\bar{Y}} v_j$, and so

$$\sum_{y\in Y} v_y = \tfrac{1}{2}\Big(\textstyle\sum_{y\in Y} v_y - \sum_{j\in\bar{Y}} v_j \Big).$$

We use the average instead of an individual sum for symmetry and improved numerical stability. The smoothed conjugate loss is then

$$L^*_\gamma(v) = \begin{cases} \frac{1}{2}\Big(\sum_{y \in Y} v_y - \sum_{j \in \bar{Y}} v_j\Big) + \frac{\gamma}{2}\left\|v\right\|^2, & \text{if } v \in S_Y, \\ +\infty, & \text{otherwise.} \end{cases}$$

To derive the primal loss, we take the conjugate again:

$$\begin{aligned} L_\gamma(u) &= \sup_v \{\langle u, v\rangle - L^*_\gamma(v)\} \\ &= \max_{v \in S_Y} \Big\{ \langle u, v\rangle - \textstyle\sum_{y \in Y} v_y - \frac{\gamma}{2}\left\|v\right\|^2 \Big\} \\ &= -\min_{v \in S_Y} \Big\{ \frac{\gamma}{2}\left\|v\right\|^2 + \textstyle\sum_{y \in Y} v_y - \langle u, v\rangle \Big\} \\ &= -\frac{1}{\gamma} \min_{\frac{v}{\gamma} \in S_Y} \Big\{ \frac{1}{2}\left\|v\right\|^2 + \textstyle\sum_{y \in Y} v_y - \langle u, v\rangle \Big\} \\ &= -\frac{1}{\gamma} \min_{\frac{v}{\gamma} \in S_Y} \Big\{ \frac{1}{2}\left\|v\right\|^2 + \frac{1}{2}\Big(\sum_{y \in Y} v_y - \sum_{j \in \bar{Y}} v_j\Big) - \langle u, v\rangle \Big\} \\ &= -\frac{1}{\gamma} \min_{\frac{v}{\gamma} \in S_Y} \Big\{ \frac{1}{2}\left\|v\right\|^2 - \textstyle\sum_{y \in Y}(\frac{1}{2} - u_y)(-v_y) - \sum_{j \in \bar{Y}}(\frac{1}{2} + u_j)v_j \Big\}. \end{aligned}$$

Next, we define the following auxiliary variables:

$$\begin{aligned} x_j &= -v_j, & b_j &= \tfrac{1}{2} - u_j, & \forall j &\in Y, \\ y_j &= v_j, & \bar{b}_j &= \tfrac{1}{2} + u_j, & \forall j &\in \bar{Y}, \end{aligned}$$

and rewrite the smooth loss $L_\gamma(u)$ equivalently as

$$\begin{aligned} L_\gamma(u) = -\frac{1}{\gamma} \min_{x,y} \; & \frac{1}{2}\left\|x\right\|^2 - \langle x, b\rangle + \frac{1}{2}\left\|y\right\|^2 - \left\langle y, \bar{b}\right\rangle \\ & \langle \mathbf{1}, x\rangle = \langle \mathbf{1}, y\rangle \leq \gamma, \\ & x \geq 0, \; y \geq 0, \end{aligned}$$

which is the Euclidean projection onto the set $B(\gamma)$. □

Note that the smooth $\text{SVM}^{\text{ML}}_\gamma$ loss is a nice generalization of the multiclass smooth $\text{SVM}^{\text{Multi}}_\gamma$ loss, and we naturally recover the latter when Y is singleton. In § 6.4, we extend the variable fixing algorithm of Kiwiel, (2008b) and obtain an efficient method to compute Euclidean projections onto $B(r)$.

Multilabel cross-entropy. Here, we discuss an extension of the LR^{Multi} loss to multilabel learning. We use the softmax function to model the distribution over the class labels $p_y(x)$, which recovers the well-known multinomial logistic regression (Krishnapuram et al., 2005) and the maximum entropy (Yu et al., 2011) models.

Assume that all the classes given in the ground truth set Y are equally likely. We define an empirical distribution for a given (x, Y) pair as $\hat{p}_y = (1/\left|Y\right|)[\![y \in Y]\!]$, and model the conditional probability $p_y(x)$ via the softmax:

$$p_y(x) = (\exp u_y) / \Big(\textstyle\sum_{j \in \mathcal{Y}} \exp u_j \Big), \quad \forall\, y \in \mathcal{Y}.$$

The cross-entropy of the distributions $\hat{p}$ and $p(x)$ is given by

$$H(\hat{p}, p(x)) = -\tfrac{1}{|Y|} \sum_{y \in Y} \log \Big(\frac{\exp u_y}{\sum_j \exp u_j} \Big),$$

and the corresponding multilabel cross entropy loss is:

$$L(u) = \tfrac{1}{|Y|} \textstyle\sum_{y \in Y} \log \Big(\textstyle\sum_{j \in \mathcal{Y}} \exp(u_j - u_y) \Big). \tag{LR$^{\text{ML}}$}$$

Multilabel cross-entropy conjugate. Next, we compute the convex conjugate of the LR$^{\text{ML}}$ loss, which is used later in our optimization framework.

Proposition 6.7. The convex conjugate of the LR$^{\text{ML}}$ loss is

$$L^*(v) = \begin{cases} \sum_{y \in Y} (v_y + \frac{1}{k}) \log(v_y + \frac{1}{k}) + \sum_{j \in \bar{Y}} v_j \log v_j & \text{if } v \in D_Y, \\ +\infty & \text{otherwise.} \end{cases} \tag{6.9}$$

where $k = |Y|$ and D_Y is the effective domain defined as:

$$D_Y \triangleq \Big\{ v \mid \textstyle\sum_{y \in Y} (v_y + \frac{1}{k}) + \sum_{j \in \bar{Y}} v_j = 1,\ v_y + \frac{1}{k} \geq 0,\ v_j \geq 0,\ y \in Y,\ j \in \bar{Y} \Big\}.$$

Proof. The conjugate loss is given by $L^*(v) = \sup\{\langle u, v \rangle - L(u) \,|\, u \in \mathbb{R}^m\}$. Since $L(u)$ is smooth and convex in u, we compute the optimal u^* by setting the partial derivatives to zero, which leads to $v_j = \frac{\partial}{\partial u_j} L(u)$. We have

$$\tfrac{\partial}{\partial u_l} L(u) = \tfrac{1}{|Y|} \sum_{y \in Y} \frac{\partial_{u_l} \Big(\sum_j \exp(u_j - u_y) \Big)}{\sum_j \exp(u_j - u_y)},$$

$$\partial_{u_l} \Big(\textstyle\sum_j \exp(u_j - u_y) \Big) = \begin{cases} \exp(u_l - u_y), & l \neq y, \\ -\sum_{j \neq y} \exp(u_j - u_y), & l = y. \end{cases}$$

Therefore,

$$\tfrac{\partial}{\partial u_l} L(u) = \tfrac{1}{|Y|} \sum_{y \in Y} \frac{1}{\sum_j \exp u_j} \begin{cases} \exp u_l, & \text{if } l \neq y, \\ -\sum_{j \neq y} \exp u_j, & \text{if } l = y. \end{cases}$$

Let $Z \triangleq \sum_{j\in\mathcal{Y}} \exp u_j$, then

$$\tfrac{\partial}{\partial u_l} L(u) = \tfrac{1}{|Y|} \sum_{y\in Y} \frac{1}{Z} \begin{cases} \exp u_l, & \text{if } l \neq y, \\ \exp u_l - Z, & \text{if } l = y. \end{cases}$$

Let $k \triangleq |Y|$, we have

$$\begin{aligned} l \notin Y \Longrightarrow \qquad & \tfrac{\partial}{\partial u_l} L(u) = \tfrac{1}{k} \textstyle\sum_{y\in Y} \tfrac{1}{Z} \exp u_l = \tfrac{1}{Z} \exp u_l, \\ l \in Y \Longrightarrow \qquad & \tfrac{\partial}{\partial u_l} L(u) = \tfrac{1}{kZ}\Big(\exp u_l - Z + (k-1)\exp u_l \Big) = \tfrac{1}{Z}\exp u_l - \tfrac{1}{k}. \end{aligned}$$

Thus, for the supremum to be attained, we must have

$$v_j = \begin{cases} \tfrac{1}{Z}\exp u_j - \tfrac{1}{k}, & \text{if } j \in Y, \\ \tfrac{1}{Z}\exp u_j, & \text{if } j \in \bar{Y}, \end{cases} \tag{6.10}$$

which means $v_j \geq -\frac{1}{k}$ if $j \in Y$, and $v_j \geq 0$ otherwise. Moreover, we have

$$\langle \mathbf{1}, v \rangle = \textstyle\sum_{j\in Y} \left(\tfrac{1}{Z}\exp u_j - \tfrac{1}{k} \right) + \sum_{j\in\bar{Y}_i} \tfrac{1}{Z}\exp u_j = \tfrac{1}{Z}\sum_{j\in\mathcal{Y}} \exp u_j - 1 = 0$$

and

$$\begin{aligned} \textstyle\sum_{j\in Y} v_j &= \textstyle\sum_{j\in Y} \left(\tfrac{1}{Z}\exp u_j - \tfrac{1}{k} \right) \leq \tfrac{1}{Z}\sum_j \exp u_j - 1 = 0, \\ \textstyle\sum_{j\in\bar{Y}_i} v_j &= \textstyle\sum_{j\in\bar{Y}_i} \tfrac{1}{Z}\exp u_j \leq \tfrac{1}{Z}\sum_j \exp u_j = 1. \end{aligned}$$

Solving (6.10) for u, we get

$$u_j^* = \begin{cases} \log(v_j + \tfrac{1}{k}) + \log Z, & \text{if } j \in Y, \\ \log v_j + \log Z, & \text{otherwise.} \end{cases}$$

Plugging the optimal u^*, we compute the conjugate as

$$\begin{aligned} L^*(Y, v) &= \langle u^*, v \rangle - \tfrac{1}{|Y|} \textstyle\sum_{y\in Y} \log \Big(\sum_j \exp(u_j^* - u_y^*) \Big) \\ &= \textstyle\sum_{y\in Y} v_y \log(v_y + \tfrac{1}{k}) + \sum_{j\in\bar{Y}} v_j \log v_j + \sum_j v_j \log Z \\ &\quad - \tfrac{1}{k} \textstyle\sum_{y\in Y} (\log Z - u_y^*) \\ &= \textstyle\sum_{y\in Y} v_y \log(v_y + \tfrac{1}{k}) + \sum_{j\in\bar{Y}} v_j \log v_j + \tfrac{1}{k}\sum_{y\in Y} \log(v_y + \tfrac{1}{k}) \\ &= \textstyle\sum_{y\in Y} (v_y + \tfrac{1}{k}) \log(v_y + \tfrac{1}{k}) + \sum_{j\in\bar{Y}} v_j \log v_j, \end{aligned}$$

where $\langle \mathbf{1}, v \rangle = 0$ and

$$\begin{aligned} &\textstyle\sum_{y\in Y} v_y \leq 0, & v_y + \tfrac{1}{k} \geq 0,\ y \in Y, \\ &\textstyle\sum_{j\in\bar{Y}} v_j \leq 1, & v_j \geq 0,\ j \in \bar{Y}. \end{aligned}$$

This leads to the definition of the effective domain D_Y, since

$$0 = \langle \mathbf{1}, v \rangle = \textstyle\sum_{y \in Y} v_y + \sum_{j \in \bar{Y}} v_j = \sum_{y \in Y} (v_y + \frac{1}{k}) + \sum_{j \in \bar{Y}} v_j - 1.$$

□

The conjugates of the multilabel losses SVM^{ML} and LR^{ML} no longer share the same effective domain, which was the case for multiclass losses. However, we still recover the conjugate of the LR^{Multi} loss when Y is singleton.

6.3 Bayes Optimality and Top-k Calibration

This section is devoted to the theoretical analysis of multiclass losses in terms of their top-k performance. We establish the best top-k error in the Bayes sense, determine when a classifier achieves it, define the notion of top-k calibration, and investigate which loss functions possess this property.

Bayes optimality. Recall that the Bayes optimal zero-one loss in binary classification is simply the probability of the least likely class (Friedman et al., 2001). Here, we extend this notion to the top-k error (6.1) introduced in § 6.2.1 for multiclass classification and provide a description of top-k Bayes optimal classifier.

Lemma 6.1. The Bayes optimal top-k error at x is

$$\min_{g \in \mathbb{R}^m} \mathbb{E}_{Y \,|\, X}[\text{err}_k(Y, g) \,|\, X = x] = 1 - \textstyle\sum_{j=1}^{k} p_{\tau_j}(x),$$

where $p_{\tau_1}(x) \geq p_{\tau_2}(x) \geq \ldots \geq p_{\tau_m}(x)$.

A classifier f is top-k Bayes optimal at x if and only if

$$\big\{ y \,|\, f_y(x) \geq f_{\pi_k}(x) \big\} \subset \big\{ y \,|\, p_y(x) \geq p_{\tau_k}(x) \big\},$$

where $f_{\pi_1}(x) \geq f_{\pi_2}(x) \geq \ldots \geq f_{\pi_m}(x)$.

Proof. For any $g = f(x) \in \mathbb{R}^m$, let π be a permutation such that $g_{\pi_1} \geq g_{\pi_2} \geq \ldots \geq g_{\pi_m}$. The expected top-$k$ error at x is

$$\begin{aligned} \mathbb{E}_{Y \,|\, X}[\text{err}_k(Y, g) \,|\, X = x] &= \textstyle\sum_{y \in \mathcal{Y}} [\![g_{\pi_k} > g_y]\!] p_y(x) = \sum_{y \in \mathcal{Y}} [\![g_{\pi_k} > g_{\pi_y}]\!] p_{\pi_y}(x) \\ &= \textstyle\sum_{j=k+1}^{m} p_{\pi_j}(x) = 1 - \sum_{j=1}^{k} p_{\pi_j}(x). \end{aligned}$$

The error is minimal when $\sum_{j=1}^{k} p_{\pi_j}(x)$ is maximal, which corresponds to taking the k largest conditional probabilities $\sum_{j=1}^{k} p_{\tau_j}(x)$ and yields the Bayes optimal top-k error at x. Since the relative order within $\{p_{\tau_j}(x)\}_{j=1}^{k}$ is irrelevant for the top-k error, any classifier $f(x)$, for which the sets $\{\pi_1, \ldots, \pi_k\}$ and $\{\tau_1, \ldots, \tau_k\}$ coincide, is Bayes optimal.

Note that we assumed w.l.o.g. that there is a clear cut $p_{\tau_k}(x) > p_{\tau_{k+1}}(x)$ between the k most likely classes and the rest. In general, ties can be resolved arbitrarily as long as we can guarantee that the k largest components of $f(x)$ correspond to the classes (indexes) that yield the maximal sum $\sum_{j=1}^{k} p_{\pi_j}(x)$ and lead to top-k Bayes optimality. □

Another way to write the optimal top-k error is $\sum_{j=k+1}^{m} p_{\pi_j}(x)$, which naturally leads to an optimal prediction strategy according to the ranking of $p_y(x)$ in descending order. However, the description of a top-k Bayes optimal classifier reveals that optimality for any given k is better understood as a *partitioning*, rather than ranking, where the labels are split into $\pi_{1:k}$ and the rest, without any preference on the ranking in either subset. If, on the other hand, we want a classifer that is top-k Bayes optimal *for all* $k \geq 1$ *simultaneously*, a proper ranking according to $p_y(x)$ is both necessary and sufficient.

Top-k calibration. Optimization of the zero-one loss and the top-k error leads to hard combinatorial problems. Instead of tackling a combinatorial problem directly, an alternative is to use a convex surrogate loss which upper bounds the discrete error. Under mild conditions on the loss function (Bartlett et al., 2006; Tewari and Bartlett, 2007), an optimal classifier for the surrogate yields a Bayes optimal solution for the zero-one loss. Such loss functions are called *classification calibrated*, which is known in statistical learning theory as a necessary condition for a classifier to be universally Bayes consistent (Bartlett et al., 2006). We introduce now the notion of calibration for the top-k error.

Definition 6.1. A loss function $L : \mathcal{Y} \times \mathbb{R}^m \to \mathbb{R}_+$ is called **top-k calibrated** if for all possible data generating measures on $\mathcal{X} \times \mathcal{Y}$ and all $x \in \mathcal{X}$

$$\arg\min_{g \in \mathbb{R}^m} \mathbb{E}_{Y \mid X}[L(Y, g) \mid X = x] \subseteq \arg\min_{g \in \mathbb{R}^m} \mathbb{E}_{Y \mid X}[\mathrm{err}_k(Y, g) \mid X = x].$$

If a loss is *not* top-k calibrated, it implies that even in the limit of infinite data, one does not obtain a classifier with the Bayes optimal top-k error from Lemma 6.1. It is thus an important property, even though of an asymptotic nature. Next, we analyse which of the multiclass methods covered in § 6.2.2 are top-k calibrated.

6.3.1 Multiclass Top-k Calibration

In this section, we consider top-k calibration of the standard OVA scheme, established multiclass classification methods, and the proposed top-k $\mathrm{Ent}_{\mathrm{tr}}$ loss. First, we state a condition under which an OVA scheme is uniformly top-k calibrated, not only for $k = 1$, which corresponds to the standard zero-one loss, but *for all* $k \geq 1$ simultaneously. The condition is given in terms of the Bayes optimal classifier for each of the corresponding binary problems and with respect to a given loss function L, e.g. the hinge or logistic losses.

Lemma 6.2. The OVA reduction is top-k calibrated for any $1 \leq k \leq m$ if the Bayes optimal function of a convex margin-based loss L is a strictly monotonically increasing function of $p_y(x) = \Pr(Y = y \mid X = x)$ for every class $y \in \mathcal{Y}$.

Proof. Let the Bayes optimal classifier for the binary problem corresponding to a $y \in \mathcal{Y}$ have the form

$$f_y(x) = g\Big(\Pr(Y = y \mid X = x)\Big),$$

where g is a strictly monotonically increasing function. The ranking of f_y corresponds to the ranking of $p_y(x)$ and hence the OVA reduction is top-k calibrated for any $k \geq 1$. □

Next, we use Lemma 6.2 and the corresponding Bayes optimal classifiers to check if the one-vs-all schemes employing hinge and logistic regression losses are top-k calibrated.

Proposition 6.8. OVA SVM is not top-k calibrated.

Proof. First, we show that the Bayes optimal function for the binary hinge loss is

$$f^*(x) = 2[\![\Pr(Y = 1 \mid X = x) > \tfrac{1}{2}]\!] - 1.$$

We decompose the expected loss as

$$\mathbb{E}_{X,Y}[L(Y, f(X))] = \mathbb{E}_X[\mathbb{E}_{Y|X}[L(Y, f(x)) \mid X = x]].$$

Thus, one can compute the Bayes optimal classifier f^* pointwise by solving

$$\arg\min_{\alpha \in \mathbb{R}} \mathbb{E}_{Y|X}[L(Y, \alpha) \mid X = x],$$

for every $x \in \mathbb{R}^d$, which leads to the following problem

$$\arg\min_{\alpha \in \mathbb{R}} \ \max\{0, 1 - \alpha\} p_1(x) + \max\{0, 1 + \alpha\} p_{-1}(x),$$

where $p_y(x) \triangleq \Pr(Y = y \mid X = x)$. It is obvious that the optimal α^* is contained in $[-1, 1]$. We get

$$\arg\min_{-1 \leq \alpha \leq 1} \ (1 - \alpha) p_1(x) + (1 + \alpha) p_{-1}(x).$$

The minimum is attained at the boundary and we get

$$f^*(x) = \begin{cases} +1 & \text{if } p_1(x) > \frac{1}{2}, \\ -1 & \text{if } p_1(x) \leq \frac{1}{2}. \end{cases}$$

Therefore, the Bayes optimal classifier for the hinge loss is not a strictly monotonically increasing function of $p_1(x)$.

To show that OVA hinge is not top-k calibrated, we construct an example problem with 3 classes and $p_1(x) = 0.4$, $p_2(x) = p_3(x) = 0.3$. Note that for every class $y = 1, 2, 3$, the Bayes optimal binary classifier is -1, hence the predicted ranking of labels is arbitrary and may not produce the Bayes optimal top-k error. □

Proposition 6.9. OVA logistic regression is top-k calibrated.

Proof. First, we show that the Bayes optimal function for the logistic loss is

$$f^*(x) = \log\Big(\frac{p_1(x)}{1 - p_1(x)}\Big).$$

As above, the pointwise optimization problem is

$$\arg\min_{\alpha \in \mathbb{R}} \; \log(1 + \exp(-\alpha))p_1(x) + \log(1 + \exp(\alpha))p_{-1}(x).$$

The logistic loss is known to be convex and differentiable and thus the optimum can be computed via

$$\frac{-\exp(-\alpha)}{1 + \exp(-\alpha)} p_1(x) + \frac{\exp(\alpha)}{1 + \exp(\alpha)} p_{-1}(x) = 0.$$

Re-writing the first fraction we get

$$\frac{-1}{1 + \exp(\alpha)} p_1(x) + \frac{\exp(\alpha)}{1 + \exp(\alpha)} p_{-1}(x) = 0,$$

which can be solved as $\alpha^* = \log\Big(\frac{p_1(x)}{p_{-1}(x)}\Big)$ and leads to the formula for the Bayes optimal classifier stated above.

We check now that the function $\phi : (0, 1) \to \mathbb{R}$ defined as $\phi(x) = \log(\frac{x}{1-x})$ is strictly monotonically increasing.

$$\phi'(x) = \frac{1-x}{x}\Big(\frac{1}{1-x} + \frac{x}{(1-x)^2}\Big) = \frac{1-x}{x}\frac{1}{(1-x)^2} = \frac{1}{x(1-x)} > 0, \quad \forall x \in (0, 1).$$

The derivative is strictly positive on $(0, 1)$, which implies that ϕ is strictly monotonically increasing. The logistic loss, therefore, fulfills the conditions of Lemma 6.2 and is top-k calibrated for any $1 \leq k \leq m$. □

The hinge loss is not calibrated since the corresponding binary classifiers, being piecewise constant, are subject to degenerate cases that result in arbitrary rankings of classes. Surprisingly, the smoothing technique based on Moreau-Yosida regularization (§ 6.2.2) makes a smoothed loss more attractive not only from the optimization side, but also in terms of top-k calibration. Here, we show that a smooth binary hinge loss from (Shalev-Shwartz and Zhang, 2014) fulfills the conditions of Lemma 6.2 and leads to a top-k calibrated OVA scheme.

Proposition 6.10. OVA smooth SVM is top-k calibrated.

Proof. In order to derive the smooth hinge loss, we first compute the conjugate of the standard binary hinge loss,

$$\begin{aligned} L(\alpha) &= \max\{0, 1-\alpha\}, \\ L^*(\beta) &= \sup_{\alpha \in \mathbb{R}} \left\{ \alpha\beta - \max\{0, 1-\alpha\} \right\} = \begin{cases} \beta & \text{if } -1 \leq \beta \leq 0, \\ \infty & \text{otherwise.} \end{cases} \end{aligned} \tag{6.11}$$

The smoothed conjugate is

$$L^*_\gamma(\beta) = L^*(\beta) + \frac{\gamma}{2}\beta^2.$$

The corresponding primal smooth hinge loss is given by

$$L_\gamma(\alpha) = \sup_{-1 \leq \beta \leq 0} \left\{ \alpha\beta - \beta - \tfrac{\gamma}{2}\beta^2 \right\} = \begin{cases} 1 - \alpha - \frac{\gamma}{2} & \text{if } \alpha < 1 - \gamma, \\ \frac{(\alpha-1)^2}{2\gamma} & \text{if } 1 - \gamma \leq \alpha \leq 1, \\ 0, & \text{if } \alpha > 1. \end{cases} \tag{6.12}$$

$L_\gamma(\alpha)$ is convex and differentiable with the derivative

$$L'_\gamma(\alpha) = \begin{cases} -1 & \text{if } \alpha < 1 - \gamma, \\ \frac{\alpha-1}{\gamma} & \text{if } 1 - \gamma \leq \alpha \leq 1, \\ 0, & \text{if } \alpha > 1. \end{cases}$$

We compute the Bayes optimal classifier pointwise.

$$f^*(x) = \operatorname*{arg\,min}_{\alpha \in \mathbb{R}} \; L(\alpha)p_1(x) + L(-\alpha)p_{-1}(x).$$

Let $p \triangleq p_1(x)$, the optimal α^* is found by solving

$$L'(\alpha)p - L'(-\alpha)(1-p) = 0.$$

Case $0 < \gamma \leq 1$**.** Consider the case $1 - \gamma \leq \alpha \leq 1$,

$$\frac{\alpha - 1}{\gamma}p + (1-p) = 0 \quad \Longrightarrow \quad \alpha^* = 1 - \gamma\frac{1-p}{p}.$$

This case corresponds to $p \geq \frac{1}{2}$, which follows from the constraint $\alpha^* \geq 1 - \gamma$. Next, consider $\gamma - 1 \leq \alpha \leq 1 - \gamma$,

$$-p + (1-p) = 1 - 2p \neq 0,$$

unless $p = \frac{1}{2}$, which is already captured by the first case. Finally, consider $-1 \leq \alpha \leq \gamma - 1 \leq 1 - \gamma$. Then

$$-p - \frac{-\alpha - 1}{\gamma}(1-p) = 0 \quad \Longrightarrow \quad \alpha^* = -1 + \gamma \frac{p}{1-p},$$

where we have $-1 \leq \alpha^* \leq \gamma - 1$ if $p \leq \frac{1}{2}$. We obtain the Bayes optimal classifier for $0 < \gamma \leq 1$ as follows:

$$f^*(x) = \begin{cases} 1 - \gamma \frac{1-p}{p} & \text{if } p \geq \frac{1}{2}, \\ -1 + \gamma \frac{p}{1-p} & \text{if } p < \frac{1}{2}. \end{cases}$$

Note that while $f^*(x)$ is not a continuous function of $p = p_1(x)$ for $\gamma < 1$, it is still a strictly monotonically increasing function of p for any $0 < \gamma \leq 1$.

Case $\gamma > 1$. First, consider $\gamma - 1 \leq \alpha \leq 1$,

$$\frac{\alpha - 1}{\gamma} p + (1-p) = 0 \quad \Longrightarrow \quad \alpha^* = 1 - \gamma \frac{1-p}{p}.$$

From $\alpha^* \geq \gamma - 1$, we get the condition $p \geq \frac{\gamma}{2}$. Next, consider $1 - \gamma \leq \alpha \leq \gamma - 1$,

$$\frac{\alpha - 1}{\gamma} p - \frac{-\alpha - 1}{\gamma}(1-p) = 0 \quad \Longrightarrow \quad \alpha^* = 2p - 1,$$

which is in the range $[1-\gamma, \gamma-1]$ if $1 - \frac{\gamma}{2} \leq p \leq \frac{\gamma}{2}$. Finally, consider $-1 \leq \alpha \leq 1 - \gamma$,

$$-p - \frac{-\alpha - 1}{\gamma}(1-p) = 0 \quad \Longrightarrow \quad \alpha^* = -1 + \gamma \frac{p}{1-p},$$

where we have $-1 \leq \alpha^* \leq 1 - \gamma$ if $p \leq 1 - \frac{\gamma}{2}$. Overall, the Bayes optimal classifier for $\gamma > 1$ is

$$f^*(x) = \begin{cases} 1 - \gamma \frac{1-p}{p} & \text{if } p \geq \frac{\gamma}{2}, \\ 2p - 1 & \text{if } 1 - \frac{\gamma}{2} \leq p \leq \frac{\gamma}{2}, \\ -1 + \gamma \frac{p}{1-p} & \text{if } p < 1 - \frac{\gamma}{2}. \end{cases}$$

Note that f^* is again a strictly monotonically increasing function of $p = p_1(x)$. Therefore, for any $\gamma > 0$, the one-vs-all scheme with the smooth hinge loss (6.12) is top-k calibrated for all $1 \leq k \leq m$ by Lemma 6.2. □

An alternative to the OVA scheme with binary losses is to use a *multiclass* loss $L : \mathcal{Y} \times \mathbb{R}^m \to \mathbb{R}_+$ directly. First, we consider the multiclass hinge loss $\text{SVM}^{\text{Multi}}$, which is known to be *not* calibrated for the top-1 error (Tewari and Bartlett, 2007), and show that it is not top-k calibrated for any k.

Proposition 6.11. Multiclass SVM is not top-k calibrated.

Proof. Let $y \in \arg\max_{j \in \mathcal{Y}} p_j(x)$. Given any $c \in \mathbb{R}$, we will show that a Bayes optimal classifier $f^* : \mathbb{R}^d \to \mathbb{R}^m$ for the $\mathrm{SVM}^{\mathrm{Multi}}$ loss is

$$\begin{aligned} f^*_y(x) &= \begin{cases} c+1 & \text{if } \max_{j \in \mathcal{Y}} p_j(x) \geq \frac{1}{2}, \\ c & \text{otherwise}, \end{cases} \\ f^*_j(x) &= c, \; j \in \mathcal{Y} \setminus \{y\}. \end{aligned}$$

Let $g = f(x) \in \mathbb{R}^m$, then

$$\mathbb{E}_{Y|X}[L(Y,g) \,|\, X] = \sum_{l \in \mathcal{Y}} \max_{j \in \mathcal{Y}} \Big\{ [\![j \neq l]\!] + g_j - g_l \Big\} p_l(x).$$

Suppose that the maximum of $(g_j)_{j \in \mathcal{Y}}$ is not unique. In this case, we have

$$\max_{j \in \mathcal{Y}} \Big\{ [\![j \neq l]\!] + g_j - g_l \Big\} \geq 1, \; \forall\, l \in \mathcal{Y}$$

as the term $[\![j \neq l]\!]$ is always active. The best possible loss is obtained by setting $g_j = c$ for all $j \in \mathcal{Y}$, which yields an expected loss of 1. On the other hand, if the maximum is unique and is achieved by g_y, then

$$\max_{j \in \mathcal{Y}} \Big\{ [\![j \neq l]\!] + g_j - g_l \Big\} = \begin{cases} 1 + g_y - g_l & \text{if } l \neq y, \\ \max\Big\{0, \, \max_{j \neq y}\{1 + g_j - g_y\}\Big\} & \text{if } l = y. \end{cases}$$

As the loss only depends on the gap $g_y - g_l$, we can optimize this with $\beta_l = g_y - g_l$.

$$\begin{aligned} \mathbb{E}_{Y|X}[L(Y,g) \,|\, X = x] &= \sum_{l \neq y} (1 + g_y - g_l) p_l(x) + \max\Big\{0, \, \max_{l \neq y}\{1 + g_l - g_y\}\Big\} p_y(x) \\ &= \sum_{l \neq y} (1 + \beta_l) p_l(x) + \max\Big\{0, \, \max_{l \neq y}\{1 - \beta_l\}\Big\} p_y(x) \\ &= \sum_{l \neq y} (1 + \beta_l) p_l(x) + \max\{0, 1 - \min_{l \neq y} \beta_l\} p_y(x). \end{aligned}$$

As only the minimal β_l enters the last term, the optimum is achieved if all β_l are equal for $l \neq y$ (otherwise it is possible to reduce the first term without affecting the last term). Let $\alpha \triangleq \beta_l$ for all $l \neq y$. The problem becomes

$$\min_{\alpha \geq 0} \sum_{l \neq y} (1 + \alpha) p_l(x) + \max\{0, 1 - \alpha\} p_y(x) \equiv \min_{0 \leq \alpha \leq 1} \alpha(1 - 2p_y(x))$$

Let $p \triangleq p_y(x) = \Pr(Y = y \,|\, X = x)$. The solution is

$$\alpha^* = \begin{cases} 0 & \text{if } p < \frac{1}{2}, \\ 1 & \text{if } p \geq \frac{1}{2}, \end{cases}$$

and the associated risk is

$$\mathbb{E}_{Y|X}[L(Y,g) \,|\, X = x] = \begin{cases} 1 & \text{if } p < \frac{1}{2}, \\ 2(1-p) & \text{if } p \geq \frac{1}{2}. \end{cases}$$

If $p < \frac{1}{2}$, then the Bayes optimal classifier $f_j^*(x) = c$ for all $j \in \mathcal{Y}$ and any $c \in \mathbb{R}$. Otherwise, $p \geq \frac{1}{2}$ and

$$f_j^*(x) = \begin{cases} c+1 & \text{if } j = y, \\ c & \text{if } j \in \mathcal{Y} \setminus \{y\}. \end{cases}$$

Moreover, we have that the Bayes risk at x is

$$\mathbb{E}_{Y|X}[L(Y, f^*(x)) \,|\, X = x] = \min\{1, 2(1-p)\} \leq 1.$$

It follows, that the multiclass hinge loss is not (top-1) classification calibrated at any x where $\max_{y \in \mathcal{Y}} p_y(x) < \frac{1}{2}$ as its Bayes optimal classifier reduces to a constant. Moreover, even if $p_y(x) \geq \frac{1}{2}$ for some y, the loss is not top-k calibrated for $k \geq 2$ as the predicted order of the remaining classes need not be optimal. □

Tewari and Bartlett, (2007) provide a general framework to study classification calibration that is applicable to a large family of multiclass methods. However, their characterization of calibration is derived in terms of the properties of the convex hull of $\{(L(1,f), \ldots, L(m,f)) \,|\, f \in \mathcal{F}\}$, which might be difficult to verify in practice. In contrast, our proofs of Propositions 6.11 and 6.12 are stratightforward and based on direct derivation of the corresponding Bayes optimal classifiers for the SVM$^{\text{Multi}}$ and the LR$^{\text{Multi}}$ losses respectively.

Proposition 6.12. Multiclass softmax loss is top-k calibrated.

Proof. The multiclass softmax loss is (top-1) calibrated for the zero-one error in the following sense. If

$$f^*(x) \in \operatorname*{arg\,min}_{g \in \mathbb{R}^m} \mathbb{E}_{Y|X}[L(Y,g) \,|\, X = x],$$

then for some $\alpha > 0$ and all $y \in \mathcal{Y}$

$$f_y^*(x) = \begin{cases} \log(\alpha \, p_y(x)) & \text{if } p_y(x) > 0, \\ -\infty & \text{otherwise,} \end{cases}$$

which implies

$$\operatorname*{arg\,max}_{y \in \mathcal{Y}} f_y^*(x) = \operatorname*{arg\,max}_{y \in \mathcal{Y}} \Pr(Y = y \,|\, X = x).$$

We now prove this result and show that it also generalizes to top-k calibration for $k > 1$. Using the identity

$$L(y, g) = \log\left(\textstyle\sum_{j \in \mathcal{Y}} e^{g_j - g_y}\right) = \log\left(\textstyle\sum_{j \in \mathcal{Y}} e^{g_j}\right) - g_y$$

and the fact that $\sum_{y \in \mathcal{Y}} p_y(x) = 1$, we write for a $g \in \mathbb{R}^m$

$$\mathbb{E}_{Y|X}[L(Y, g) \,|\, X = x] = \sum_{y \in \mathcal{Y}} L(y, g) p_y(x) = \log\Big(\sum_{y \in \mathcal{Y}} e^{g_y}\Big) - \sum_{y \in \mathcal{Y}} g_y p_x(y).$$

As the loss is convex and differentiable, we get the global optimum by computing a critical point. We have

$$\frac{\partial}{\partial g_j} \mathbb{E}_{Y|X}[L(Y, g) \,|\, X = x] = \frac{e^{g_j}}{\sum_{y \in \mathcal{Y}} e^{g_y}} - p_j(x) = 0$$

for $j \in \mathcal{Y}$. We note that the critical point is not unique as multiplication $g \to \kappa g$ leaves the equation invariant for any $\kappa > 0$. One can verify that $e^{g_j} = \alpha p_j(x)$ satisfies the equations for any $\alpha > 0$. This yields a solution

$$f_y^*(x) = \begin{cases} \log(\alpha p_y(x)) & \text{if } p_y(x) > 0, \\ -\infty & \text{otherwise,} \end{cases}$$

for any fixed $\alpha > 0$. We note that f_y^* is a strictly monotonically increasing function of the conditional class probabilities. Therefore, it preserves the ranking of $p_y(x)$ and implies that f^* is top-k calibrated for any $1 \leq k \leq m$. □

The implicit reason for top-k calibration of the OVA schemes and the softmax loss is that one can estimate the probabilities $p_y(x)$ from the Bayes optimal classifier. Loss functions which allow this are called *proper*. We refer to (Reid and Williamson, 2010a) and references therein for a detailed discussion.

We have established that the OVA logistic regression and the softmax loss are top-k calibrated for any k, so why should we be interested in defining new loss functions for the top-k error? The reason is that calibration is an asymptotic property since the Bayes optimal functions are obtained by *pointwise* minimization of $\mathbb{E}_{Y|X}[L(Y, f(x)) \,|\, X = x]$ at every $x \in \mathcal{X}$. The picture changes if we use linear classifiers, since they cannot be minimized independently at each point. Indeed, the Bayes optimal classifiers, in general, cannot be realized by linear functions.

Furthermore, convexity of the softmax and multiclass hinge losses leads to phenomena where $\mathrm{err}_k(y, f(x)) = 0$, but $L(y, f(x)) \gg 0$. We discussed this issue § 6.2.2 and motivated modifications of the above losses for the top-k error. Next, we show that one of the proposed top-k losses is also top-k calibrated.

Proposition 6.13. The truncated top-k entropy loss is top-s calibrated for any $k \leq s \leq m$.

Proof. Given any $g = f(x) \in \mathbb{R}^m$, let π be a permutation such that $g_{\pi_1} \geq g_{\pi_2} \geq \ldots \geq g_{\pi_m}$. Then, we have

$$\mathcal{J}_y = \begin{cases} \{\pi_{k+1}, \ldots, \pi_m\} & \text{if } y \in \{\pi_1, \ldots, \pi_{k-1}\}, \\ \{\pi_k, \ldots, \pi_m\} \setminus \{y\} & \text{if } y \in \{\pi_k, \ldots, \pi_m\}. \end{cases}$$

Therefore, the expected loss at x can be written as

$$\begin{aligned} \mathbb{E}_{Y|X}[L(Y,g) \,|\, X = x] &= \textstyle\sum_{y \in \mathcal{Y}} L(y,g)\, p_y(x) \\ &= \textstyle\sum_{r=1}^{k-1} \log \Big(1 + \sum_{j=k+1}^m e^{g_{\pi_j} - g_{\pi_r}}\Big)\, p_{\pi_r}(x) + \sum_{r=k}^m \log \Big(\sum_{j=k}^m e^{g_{\pi_j} - g_{\pi_r}}\Big)\, p_{\pi_r}(x). \end{aligned}$$

Note that the sum inside the logarithm does not depend on g_{π_r} for $r < k$. Therefore, a Bayes optimal classifier will have $g_{\pi_r} = +\infty$ for all $r < k$ as then the first sum vanishes.

Let $p \triangleq (p_y(x))_{y \in \mathcal{Y}}$ and $q \triangleq (L(y,g))_{y \in \mathcal{Y}}$, then

$$q_{\pi_1} = \ldots = q_{\pi_{k-1}} = 0 \leq q_{\pi_k} \leq \ldots \leq q_{\pi_m}$$

and we can re-write the expected loss as

$$\mathbb{E}_{Y|X}[L(Y,g) \,|\, X = x] = \langle p, q \rangle = \langle p_\pi, q_\pi \rangle \geq \langle p_\tau, q_\pi \rangle\,,$$

where $p_{\tau_1} \geq p_{\tau_2} \geq \ldots \geq p_{\tau_m}$ and we used the rearrangement inequality. Therefore, the expected loss is minimized when π and τ coincide (up to a permutation of the first $k-1$ elements), which already establishes top-s calibration for all $s \geq k$.

We can also derive a Bayes optimal classifier following the proof of Proposition 6.12. We have

$$\begin{aligned} \mathbb{E}_{Y|X}[L(Y,g) \,|\, X = x] &= \textstyle\sum_{r=k}^m \log \Big(\sum_{j=k}^m e^{g_{\tau_j} - g_{\tau_r}}\Big)\, p_{\tau_r}(x) \\ &= \textstyle\sum_{r=k}^m \Big(\log \Big(\sum_{j=k}^m e^{g_{\tau_j}}\Big) - g_{\tau_r}\Big)\, p_{\tau_r}(x). \end{aligned}$$

A critical point is found by setting partial derivatives to zero for $y \in \{\tau_k, \ldots, \tau_m\}$,

$$\frac{e^{g_y}}{\sum_{j=k}^m e^{g_{\tau_j}}} \textstyle\sum_{r=k}^m p_{\tau_r}(x) = p_y(x).$$

We let $g_y = -\infty$ if $p_y(x) = 0$, and obtain finally

$$g^*_{\tau_j} = \begin{cases} +\infty & \text{if } j < k, \\ \log \big(\alpha p_{\tau_j}(x)\big) & \text{if } j \geq k \text{ and } p_{\tau_j}(x) > 0, \\ -\infty & \text{if } j \geq k \text{ and } p_{\tau_j}(x) = 0, \end{cases}$$

as a Bayes optimal classifier for any $\alpha > 0$.

Note that g^* preserves the ranking of $p_y(x)$ for all y in $\{\tau_k, \ldots, \tau_m\}$, hence, it is top-$s$ calibrated for all $s \geq k$. □

Top-k calibration of the remaining top-k losses is an open problem, which is complicated by the absence of a closed-form expression for most of them.

6.4 Optimization Framework

This section is mainly devoted to efficient optimization of the multiclass and multilabel methods from § 6.2 within the stochastic dual coordinate ascent (SDCA) framework of Shalev-Shwartz and Zhang, (2013b). A portion of the material presented here has been covered already in § 5.3 (page 88). However, we make a few important extensions in this section: (i) we no longer assume that multiclass loss functions are j-compatible; (ii) we use the Lambert W function for entropic projections; (iii) we cover multiclass and multilabel methods in a unified framework.

The core reason for efficiency of the optimization scheme is the ability to formulate variable updates in terms of projections onto the effective domain of the conjugate loss, which, in turn, can be solved in time $O(m \log m)$ or faster. These projections fall into a broad area of nonlinear resource allocation (Patriksson and Strömberg, 2015), where we already have a large selection of specialized algorithms. For example, we use an algorithm of Kiwiel, (2008b) for $\text{SVM}^{\text{Multi}}$ and top-k SVM^{β}, and contribute analogous algorithms for the remaining losses. In particular, we propose an entropic projection algorithm based on the Lambert W function for the LR^{Multi} loss, and a variable fixing algorithm for projecting onto the bipartite simplex (6.8) for the SVM^{ML}. We also discuss how the proposed loss functions that do not have a closed-form expression can be evaluated efficiently, and perform a runtime comparison against FISTA (Beck and Teboulle, 2009) using the SPAMS optimization toolbox of Mairal et al., (2010).

- In § 6.4.1, we state the primal and Fenchel dual optimization problems, and introduce the Lambert W function.
- In § 6.4.2, we consider SDCA update steps and loss computation for multiclass methods, as well as present our runtime evaluation experiments.
- In § 6.4.3, we cover multilabel optimization and present our algorithm for the Euclidean projection onto the bipartite simplex.

6.4.1 Technical Background

We briefly recall the main facts about the SDCA framework (Shalev-Shwartz and Zhang, 2013b), Fenchel duality (Borwein and Lewis, 2000), and the Lambert W function (Corless et al., 1996).

The primal and dual problems. Let $X \in \mathbb{R}^{d \times n}$ be the matrix of training examples $x_i \in \mathbb{R}^d$, $K = X^\top X$ the corresponding Gram matrix, $W \in \mathbb{R}^{d \times m}$ the matrix of primal variables, $A \in \mathbb{R}^{m \times n}$ the matrix of dual variables, and $\lambda > 0$ the regularization parameter. The primal and Fenchel dual (§ A.2, page 183) objective functions are given as

$$\begin{aligned} P(W) &= +\frac{1}{n} \sum_{i=1}^{n} L\left(y_i, W^\top x_i\right) + \frac{\lambda}{2} \mathbf{tr}\left(W^\top W\right), \\ D(A) &= -\frac{1}{n} \sum_{i=1}^{n} L^*\left(y_i, -\lambda n a_i\right) - \frac{\lambda}{2} \mathbf{tr}\left(AKA^\top\right), \end{aligned} \tag{6.13}$$

where L^* is the convex conjugate of the loss L and y_i is interpreted as a set Y_i if L is a multilabel loss.

SDCA proceeds by sampling a dual variable $a_i \in \mathbb{R}^m$, which corresponds to a training example $x_i \in \mathbb{R}^d$, and modifying it to achieve maximal increase in the dual objective $D(A)$ while keeping other dual variables fixed. Several sampling strategies can be used, e.g. (Qu et al., 2015), but we use a simple scheme where the set of indexes is randomly shuffled before every epoch and then all a_i's are updated sequentially. The algorithm terminates when the relative duality gap $(P(W) - D(A))/P(W)$ falls below a predefined $\varepsilon > 0$, or the computational budget is exhausted, in which case we still have an estimate of suboptimality via the duality gap.

Since the algorithm operates entirely on the dual variables and the prediction scores $f(x_i)$, it is directly applicable to training both linear $f(x_i) = W^\top x_i$ as well as nonlinear $f(x_i) = AK_i$ classifiers (K_i being the ith column of the Gram matrix K). When $d \ll n$, which is often the case in our experiments, and we are training a linear classifier, then it is less expensive to maintain the primal variables $W = XA^\top$ and compute the dot products $W^\top x_i$ in $\mathbb{R}^d$. In that case, whenever a_i is updated, we perform a rank-1 update of W.

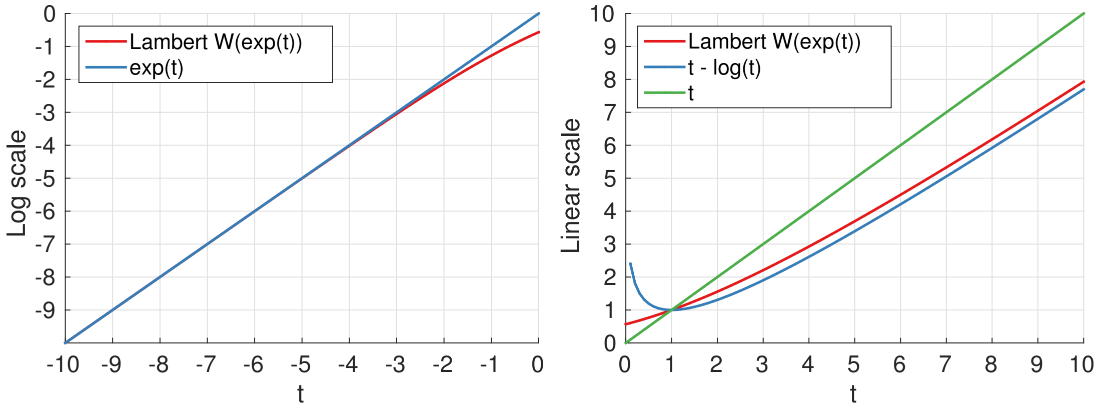
The SDCA update step $a_i \leftarrow \max_{a_i} D(A)$ turns out to be equivalent to the proximal operator[2] of a certain function and can also be seen as a projection onto the effective domain of L^*.

The conjugate loss. An important ingredient in the SDCA framework is the conjugate loss L^*. We show that for all multiclass loss functions that we consider the fact that they depend on the differences $f_j(x) - f_y(x)$ enforces a certain constraint on the conjugate function.

Lemma 6.3. Let $H_y = \mathbf{I} - \mathbf{1} e_y^\top$ and let $L(u) = \phi(H_y u)$ for a loss $\phi : \mathbb{R}^m \to \mathbb{R}_+$, then $L^*(v) = +\infty$ unless $\langle \mathbf{1}, v \rangle = 0$.

Proof. The proof follows directly from Lemma 5.2 and is partially reproduced in the proof of Proposition 6.3. This generalized lemma drops the constraint of y-compatibility as the latter does not apply to the softmax loss. □

[2] The **proximal operator**, or the **proximal map**, of a function f is defined by $\mathbf{prox}_f(v) = \arg\min_x \left(f(x) + \frac{1}{2}\|x - v\|^2\right)$.



(a) $V(t) \approx e^t$ for $t \ll 0$. **(b)** $V(t) \approx t - \log t$ for $t \gg 0$.

Figure 6.1.: Behavior of the Lambert W function of the exponent $(V(t) = W(e^t))$. **(a)** Log scale plot with $t \in (-10, 0)$. **(b)** Linear scale plot with $t \in (0, 10)$.

This has an implication that we need to enforce $\langle \mathbf{1}, a_i \rangle = 0$ during optimization, which translates into $a_{y_i} = -\sum_{j \neq y_i} a_j$ for all multiclass losses. Therefore, the update steps are actually performed on the $(m-1)$-dimensional vectors obtained by removing the coordinate a_{y_i}. For multilabel losses, we have $\sum_{y \in Y_i} a_y = -\sum_{j \in \bar{Y}_i} a_j$, which leads to the definition of the bipartite simplex (6.8) and a specialized projection algorithm.

Lambert W function. The Lambert W function is defined as the inverse of the mapping $w \mapsto we^w$. It is widely used in many fields of computer science (Corless et al., 1996; Fukushima, 2013; Veberič, 2012), and can often be recognized in nonlinear equations involving the exp and the log functions. Taking logarithms on both sides of the defining equation $z = We^W$, we get $\log z = W(z) + \log W(z)$. Therefore, if we are given an equation of the form $x + \log x = t$ for some $t \in \mathbb{R}$, we can directly "solve" it in closed-form as $x = W(e^t)$. The crux of the problem is that the function $V(t) \triangleq W(e^t)$ is transcendental (Fukushima, 2013) just like the logarithm and the exponent. There exist highly optimized implementations for the latter and we argue that the same can be done for the Lambert W function. In fact, there is already some work on this topic (Fukushima, 2013; Veberič, 2012), which we also employ in our implementation.

To develop intuition about the function $V(t) = W(e^t)$, which is the Lambert W function of the exponent, we look at how it behaves for different values of t. An illustration is provided in Figure 6.1. One can see directly from the equation $x + \log x = t$ that the behavior of $x = V(t)$ changes dramatically depending on whether t is a large positive or a large negative number. In the first case, the linear part dominates the logarithm and the function is approximately linear; a better approximation is $x(t) \approx t - \log t$, when $t \gg 1$. In the second case, the function behaves like an exponent e^t. To see this, we write $x = e^t e^{-x}$ and note that $e^{-x} \approx 1$ when $t \ll 0$, therefore, $x(t) \approx e^t$, if $t \ll 0$.

To compute $V(t)$, we use these approximations as initial points in a 5-th order Householder method (Householder, 1970). A *single* iteration is already sufficient to get full float precision and at most two iterations are needed for double, which makes the function $V(t)$ an attractive tool for computing entropic projections.

6.4.2 Multiclass Methods

In this section, we cover optimization of the multiclass methods from § 6.2.2 within the SDCA framework. We discuss how to efficiently compute the smoothed losses that were introduced via conjugation and do not have a closed-form expression. Finally, we evaluate SDCA convergence in terms of runtime and show that smoothing with Moreau-Yosida regularization leads to significant improvements in speed.

As mentioned in § 6.4.1 above, the core of the SDCA algorithm is the update step $a_i \leftarrow \arg\max_{a_i} D(A)$. Even the primal objective $P(W)$ is only computed for the duality gap and could conceivably be omitted if the certificate of optimality is not required. Next, we focus on how the updates are computed for the different multiclass methods.

SDCA update: $\mathrm{SVM}^{\mathrm{OVA}}$**,** $\mathrm{LR}^{\mathrm{OVA}}$**.** SDCA updates for the binary hinge and logistic losses are covered in (Hsieh et al., 2008) and (Shalev-Shwartz and Zhang, 2014). We highlight that the $\mathrm{SVM}^{\mathrm{OVA}}$ update has a closed-form expression that leads to scalable training of linear SVMs (Hsieh et al., 2008), and is implemented in LibLinear (Fan et al., 2008).

SDCA update: $\mathrm{SVM}^{\mathrm{Multi}}$**,** $\mathrm{LR}^{\mathrm{Multi}}$**,** $\mathrm{SVM}^{\mathrm{Multi}}_{\gamma}$**.** Although $\mathrm{SVM}^{\mathrm{Multi}}$ is also covered in (Shalev-Shwartz and Zhang, 2014), they use a different algorithm based on sorting, while we do a case distinction (§ 5.4.3, page 98). First, we solve an easier continuous quadratic knapsack problem using a variable fixing algorithm of Kiwiel, (2008b) which does not require sorting. This corresponds to enforcing the equality constraint in the simplex and generally already gives the optimal solution. The computation is also fast: we observe linear time complexity in practice, as shown in Figure 5.3. For the remaining hard cases, however, we fall back to sorting and use a scheme similar to (Shalev-Shwartz and Zhang, 2014). In our experience, performing the case distinction offers significant time savings.

For the $\mathrm{SVM}^{\mathrm{Multi}}$ and $\mathrm{SVM}^{\mathrm{Multi}}_{\gamma}$, note that they are special cases of top-k $\mathrm{SVM}^{\alpha}_{\gamma}$ and top-k $\mathrm{SVM}^{\beta}_{\gamma}$ with $k = 1$, as well as $\mathrm{LR}^{\mathrm{Multi}}$ is a special case of top-k Ent.

SDCA update: top-k $\mathrm{SVM}^{\alpha/\beta}$**,** top-$k$ $\mathrm{SVM}^{\alpha/\beta}_{\gamma}$**.** Here, we consider the update step for the smooth top-k $\mathrm{SVM}^{\alpha}_{\gamma}$ loss. The non-smooth version is directly recovered by setting $\gamma = 0$, while the update for top-k $\mathrm{SVM}^{\beta}_{\gamma}$ is derived similarly using the set Δ^{β}_{k} in (6.14) instead of Δ^{α}_{k}.

We show that performing the update step is equivalent to projecting a certain vector b, computed from the prediction scores $f(x_i) = W^{\top} x_i$, onto the effective domain of L^*, the top-k simplex, with an added regularization $\rho \langle \mathbf{1}, x \rangle^2$, which biases the solution to be orthogonal to $\mathbf{1}$.

Proposition 6.14. Let L and L^* in (6.13) be respectively the top-k SVM$^\alpha_\gamma$ loss and its conjugate as in Proposition 6.2. The dual variables a_i corresponding to (x_i, y_i) are updated as:

$$\begin{cases} a_i^{\setminus y_i} &= -\arg\min_{x \in \Delta^\alpha_k(1/(\lambda n))} \left\{ \|x - b\|^2 + \rho \langle \mathbf{1}, x\rangle^2 \right\}, \\ a_{y_i,i} &= -\sum_{j \neq y_i} a_{j,i}, \end{cases} \tag{6.14}$$

where $b = \frac{1}{\langle x_i, x_i\rangle + \gamma\lambda n}\left(q^{\setminus y_i} + (1 - q_{y_i})\mathbf{1}\right)$, $q = W^\top x_i - \langle x_i, x_i\rangle a_i$, and $\rho = \frac{\langle x_i, x_i\rangle}{\langle x_i, x_i\rangle + \gamma\lambda n}$.

Proof. We follow the proof of Proposition 5.5 (page 92). Choose an $i \in \{1, \dots, n\}$ and update a_i to maximize

$$-\tfrac{1}{n} L^*(y_i, -\lambda n a_i) - \tfrac{\lambda}{2}\, \mathbf{tr}\left(AKA^\top\right).$$

For the nonsmooth top-k hinge loss, it was shown in Proposition 5.5 that

$$L^*(y_i, -\lambda n a_i) = \langle c, \lambda n(a_i - a_{y_i,i} e_{y_i})\rangle$$

if $-\lambda n(a_i - a_{y_i,i} e_{y_i}) \in \Delta^\alpha_k$ and $+\infty$ otherwise. Now, for the smoothed loss, we add regularization and obtain

$$-\tfrac{1}{n}\left(\tfrac{\gamma}{2}\|-\lambda n(a_i - a_{y_i,i} e_{y_i})\|^2 + \langle c, \lambda n(a_i - a_{y_i,i} e_{y_i})\rangle\right)$$

with $-\lambda n(a_i - a_{y_i,i} e_{y_i}) \in \Delta^\alpha_k$. Using $c = \mathbf{1} - e_{y_i}$ and $\langle \mathbf{1}, a_i\rangle = 0$, it simplifies to

$$-\frac{\gamma n \lambda^2}{2}\left\|a_i^{\setminus y_i}\right\|^2 + \lambda a_{y_i,i},$$

and the feasibility constraint can be re-written as

$$-a_i^{\setminus y_i} \in \Delta^\alpha_k(\tfrac{1}{\lambda n}), \qquad a_{y_i,i} = \langle \mathbf{1}, -a_i^{\setminus y_i}\rangle.$$

For the regularization term $\mathbf{tr}\left(AKA^\top\right)$, we have

$$\mathbf{tr}\left(AKA^\top\right) = K_{ii}\langle a_i, a_i\rangle + 2\sum_{j \neq i} K_{ij}\langle a_i, a_j\rangle + \text{const}.$$

We let $q = \sum_{j \neq i} K_{ij} a_j = AK_i - K_{ii} a_i$ and $x = -a_i^{\setminus y_i}$:

$$\begin{aligned} \langle a_i, a_i\rangle &= \langle \mathbf{1}, x\rangle^2 + \langle x, x\rangle, \\ \langle q, a_i\rangle &= q_{y_i}\langle \mathbf{1}, x\rangle - \langle q^{\setminus y_i}, x\rangle. \end{aligned}$$

Now, we plug everything together and multiply with $-2/\lambda$.

$$\min_{x \in \Delta^\alpha_k(\frac{1}{\lambda n})} \gamma\lambda n\|x\|^2 - 2\langle \mathbf{1}, x\rangle + 2\Big(q_{y_i}\langle \mathbf{1}, x\rangle - \langle q^{\setminus y_i}, x\rangle\Big) + K_{ii}\Big(\langle \mathbf{1}, x\rangle^2 + \langle x, x\rangle\Big).$$

Collecting the corresponding terms finishes the proof. □

We solve (6.14) using the algorithm for computing a (biased) projection onto the top-k simplex, which we introduced in § 5.4.3 (page 98), with a minor modification of b and ρ. Similarly, the update step for the top-k SVM$_\gamma^\beta$ loss is solved using a (biased) continuous quadratic knapsack problem, which we discussed in § 5.4.4.

Smooth top-k hinge losses converge significantly faster than their nonsmooth variants as we show in the scaling experiments below. This can be explained by the theoretical results of Shalev-Shwartz and Zhang, (2014), where they also had similar observations for the smoothed binary hinge loss.

SDCA update: top-k Ent**.** Finally, we derive an optimization problem for the proposed top-k entropy loss.

Proposition 6.15. Let L in (6.13) be the top-k Ent loss and L^* be its convex conjugate as in (6.4) with Δ replaced by Δ_k^α. The dual variables a_i corresponding to (x_i, y_i) are updated as:

$$\begin{cases} a_i^{\setminus y_i} = -\frac{1}{\lambda n} \operatorname*{arg\,min}_{x \in \Delta_k^\alpha} \Big\{ \frac{\alpha}{2}(\langle x, x\rangle + s^2) - \langle b, x\rangle + \langle x, \log x\rangle \\ \qquad\qquad + (1-s)\log(1-s) \,|\, s = \langle \mathbf{1}, x\rangle \Big\}, \\ a_{y_i,i} = -\sum_{j \neq y_i} a_{j,i}, \end{cases} \tag{6.15}$$

where $\alpha = \frac{\langle x_i, x_i\rangle}{\lambda n}$, $b = q^{\setminus y_i} - q_{y_i}\mathbf{1}$, $q = W^\top x_i - \langle x_i, x_i\rangle a_i$.

Proof. Let $v \triangleq -\lambda n a_i$ and $y = y_i$. Using Proposition 6.3,

$$L^*(v) = \textstyle\sum_{j\neq y} v_j \log v_j + (1 + v_y)\log(1 + v_y),$$

where $\langle \mathbf{1}, v\rangle = 0$ and $v^{\setminus y} \in \Delta_k^\alpha$. Let $x \triangleq v^{\setminus y}$ and $s \triangleq -v_y$. We have $s = \langle \mathbf{1}, x\rangle$ and from $\mathbf{tr}\left(AKA^\top\right)$ we get

$$K_{ii}(\langle x, x\rangle + s^2)/(\lambda n)^2 - 2\Big\langle q^{\setminus y} - q_y\mathbf{1}, x\Big\rangle/(\lambda n),$$

where $q = \sum_{j\neq i} K_{ij} a_j = AK_i - K_{ii}a_i$. Finally, we plug everything together as in Proposition 6.14. □

Problems (6.14) and (6.15) have similar structure, but the latter is considerably more difficult to solve due to the presence of logarithms. We propose to tackle this problem using the function $V(t)$ introduced in § 6.4.1 above.

Our algorithm is an instance of the variable fixing scheme with the following steps: (i) partition the variables into disjoint sets and compute an auxiliary variable t from the optimality conditions; (ii) compute the values of the variables using t and verify them against a set of constraints (e.g. an upper bound in the top-k simplex); (iii) if there are no violated constraints, we have computed the solution, and otherwise examine the next partitioning.

There can be at most k partitionings that we need to consider for Δ_k^α and Δ_k^β. To see this, let $x \in \Delta_k^\alpha$ be a feasible point for (6.15), and define the subsets

$$U \triangleq \{j \,|\, x_j = \tfrac{s}{k}\}, \qquad M \triangleq \{j \,|\, x_j < \tfrac{s}{k}\}. \tag{6.16}$$

Clearly, $|U| \leq k$ must hold, and $|U| = k$ we consider as a degenerate fall back case. Therefore, we are interested in the k partitions when $0 \leq |U| < k$. Due to monotonicity in the optimality conditions, one can show that U always corresponds to the largest elements b_j of the vector being projected. Hence, we start with an empty U and add indexes of the largest b_j's until the solution is found.

Next, we show how to actually compute t and x, given a candidate partition into U and M.

Proposition 6.16. Let x^* be the solution of (6.15) and let the sets U and M be defined for the given x^* as in (6.16), then

$$x^*_j = \min\left\{\tfrac{1}{\alpha}V(b_j - t), \tfrac{s}{k}\right\}, \quad \forall j,$$

and the variables s, t satisfy the nonlinear system

$$\begin{cases} \alpha(1-\rho)s - \sum_{j \in M} V(b_j - t) = 0, \\ (1-\rho)t + V^{-1}\Big(\alpha(1-s)\Big) - \rho V^{-1}(\tfrac{\alpha s}{k}) + A - \alpha = 0, \end{cases} \tag{6.17}$$

where $\rho \triangleq \frac{|U|}{k}$, $A \triangleq \frac{1}{k}\sum_{j \in U} b_j$, V^{-1} is the inverse of V.

Moreover, if U is empty, then $x^*_j = \frac{1}{\alpha}V(b_j - t)$ for all j, and t can be found from

$$V(\alpha - t) + \textstyle\sum_j V(b_j - t) = \alpha. \tag{6.18}$$

Proof. The Lagrangian of (6.15) is given by

$$\begin{aligned} \mathcal{L}(x, s, t, \lambda, \mu, \nu) = \tfrac{\alpha}{2}(\langle x, x\rangle + s^2) - \langle b, x\rangle + \langle x, \log x\rangle + (1-s)\log(1-s) \\ + t(\langle \mathbf{1}, x\rangle - s) + \lambda(s-1) - \langle \mu, x\rangle + \left\langle \nu, x - \tfrac{s}{k}\mathbf{1}\right\rangle, \end{aligned}$$

where $t \in \mathbb{R}$, $\lambda, \mu, \nu \geq 0$ are the dual variables. Computing partial derivatives of $\mathcal{L}$ w.r.t. x_j and s, and setting them to zero, we obtain

$$\begin{aligned} \alpha x_j + \log x_j &= b_j - 1 - t + \mu_j - \nu_j, \quad \forall j, \\ \alpha(1-s) + \log(1-s) &= \alpha - 1 - t - \lambda - \tfrac{1}{k}\langle \mathbf{1}, \nu\rangle, \quad \forall j. \end{aligned}$$

Note that only $x_j > 0$ and $s < 1$ satisfy the above constraints, which implies $\mu_j = 0$ and $\lambda = 0$. We re-write the above as

$$\begin{aligned} &\alpha x_j + \log(\alpha x_j) = b_j - 1 - t + \log\alpha - \nu_j, \\ &\alpha(1-s) + \log\Big(\alpha(1-s)\Big) = \alpha - 1 - t + \log\alpha - \tfrac{\langle \mathbf{1}, \nu\rangle}{k}. \end{aligned}$$

These equations correspond to the Lambert W function of the exponent, $V(t) = W(e^t)$, discussed in § 6.4.1. Let $p \triangleq \langle \mathbf{1}, \nu\rangle$ and re-define $t \leftarrow 1 + t - \log\alpha$.

$$\begin{aligned} \alpha x_j &= W\Big(\exp(b_j - t - \nu_j)\Big), \\ \alpha(1-s) &= W\Big(\exp(\alpha - t - \tfrac{p}{k})\Big). \end{aligned}$$

Finally, we obtain the following system:

$$\begin{aligned} \alpha x_j &= V(b_j - t - \nu_j), \quad \forall j \\ \alpha(1-s) &= V(\alpha - t - \tfrac{p}{k}), \\ s &= \langle \mathbf{1}, x \rangle, \quad p = \langle \mathbf{1}, \nu \rangle . \end{aligned}$$

Note that $V(t)$ is a strictly monotonically increasing function, therefore, it is invertible and we can write

$$\begin{aligned} b_j - t - \nu_j &= V^{-1}(\alpha x_j), \\ \alpha - t - \tfrac{p}{k} &= V^{-1}\big(\alpha(1-s)\big). \end{aligned}$$

Next, we use the definition of the sets U and M,

$$\begin{aligned} s = \langle \mathbf{1}, x \rangle &= \textstyle\sum_U \frac{s}{k} + \sum_M \frac{1}{\alpha} V(b_j - t), \\ p = \langle \mathbf{1}, \nu \rangle &= \textstyle\sum_U b_j - |U| \Big(t + V^{-1}(\frac{\alpha s}{k})\Big). \end{aligned}$$

Let $\rho \triangleq \frac{|U|}{k}$ and $A \triangleq \frac{1}{k} \sum_U b_j$, we get

$$\begin{aligned} (1-\rho)s &= \tfrac{1}{\alpha} \textstyle\sum_M V(b_j - t), \\ \tfrac{p}{k} &= A - \rho\Big(t + V^{-1}(\tfrac{\alpha s}{k})\Big). \end{aligned}$$

Finally, we eliminate p and obtain the system:

$$\begin{aligned} &\alpha(1-\rho)s - \textstyle\sum_M V(b_j - t) = 0, \\ &(1-\rho)t + V^{-1}\big(\alpha(1-s)\big) - \rho V^{-1}(\tfrac{\alpha s}{k}) + A - \alpha = 0. \end{aligned}$$

Moreover, when U is empty, it simplifies into a single equation

$$V(\alpha - t) + \textstyle\sum_M V(b_j - t) = \alpha.$$

□

We solve (6.17) using the Newton method (Nocedal and Wright, 2006), while for (6.18) we use a 4-th order Householder method (Householder, 1970) with a faster convergence rate. The latter is particularly attractive, since the set U can be assumed empty for $k = 1$, i.e. for the $\mathrm{LR}^{\mathrm{Multi}}$ loss, and is often also empty for the general top-k Ent loss. As both methods require the derivatives of $V(t)$, we note that $\partial_t V(t) = V(t)/(1 + V(t))$ (Corless et al., 1996), which means that the derivatives come at no additional cost. Finally, we note that $V^{-1}(v) = v + \log v$.

Loss computation: $\mathrm{SVM}_\gamma^{\mathrm{Multi}}$**, top-**$k$ $\mathrm{SVM}_\gamma^{\alpha/\beta}$**.** Here, we discuss how to evaluate smoothed losses that do not have a closed-form expression for the primal loss. Recall that the smooth top-k $\mathrm{SVM}_\gamma^\alpha$ loss is given by

$$L_\gamma(a) = \tfrac{1}{\gamma}\Big(\langle (a+c)^{\setminus y}, p \rangle - \tfrac{1}{2}\|p\|^2\Big),$$

where $a_j = f_j(x) - f_y(x)$, $c_j = 1 - [\![y = j]\!]$ for all $j \in \mathcal{Y}$, and $p = \mathbf{proj}_{\Delta_k^\alpha(\gamma)}(a+c)^{\setminus y}$ is the Euclidean projection of $(a+c)^{\setminus y}$ onto $\Delta_k^\alpha(\gamma)$. We described an $O(m \log m)$ algorithm to compute the projection p in § 5.4.3. For the special case $k = 1$, i.e. the $\text{SVM}_\gamma^{\text{Multi}}$ loss, the algorithm is particularly efficient and exhibits essentially linear scaling in practice. Moreover, since we only need the dot products with p in $L_\gamma(a)$, we exploit its special structure, $p = \min\{\max\{l, b-t\}, u\}$ with $b = (a+c)^{\setminus y}$, and avoid explicit computation of p. The same procedure is done for top-k SVM_γ^β.

Loss computation: top-k Ent**.** Next, we discuss how to evaluate the top-k Ent loss that was defined via the conjugate of the softmax loss as

$$\max_{x \in \Delta_k^\alpha,\, s = \langle \mathbf{1}, x \rangle} \Big\{ \langle a^{\setminus y}, x \rangle - (1-s)\log(1-s) - \langle x, \log x \rangle \Big\}. \tag{6.19}$$

Note that (6.19) is similar to (6.15) and we use a similar variable fixing scheme, as described above. However, this problem is much easier: the auxiliary variables s and t are computed directly without having to solve a nonlinear system, and their computation does not involve the $V(t)$ function.

Proposition 6.17. Let x^* be the solution of (6.19) and let the sets U and M be defined for the given x^* as in (6.16), then

$$x_j^* = \min\Big\{ \exp(a_j - t), \tfrac{s}{k} \Big\}, \quad \forall\, j,$$

and the variables s, t are computed from

$$\begin{cases} s = 1/(1+Q), \\ t = \log Z + \log(1+Q) - \log(1-\rho), \end{cases} \tag{6.20}$$

where $\rho \triangleq \frac{|U|}{k}$, $A \triangleq \frac{1}{k}\sum_{j \in U} a_j$, $Z \triangleq \sum_{j \in M} \exp a_j$, and

$$Q \triangleq (1-\rho)^{(1-\rho)} / (k^\rho Z^{(1-\rho)} \exp A).$$

The top-k Ent loss is then computed as

$$L(a) = (A + (1-\rho)t - \rho \log(\tfrac{s}{k}))s - (1-s)\log(1-s).$$

Moreover, if U is empty, then $x_j^* = \exp(a_j - t)$ for all j, and we recover the softmax loss LR^{Multi} as

$$L(a) = t = \log(1+Z) = \log(1 + \textstyle\sum_j \exp a_j).$$

Proof. We continue the derivation started in the proof of Propostion 6.4. First, we write the system that follows directly from the KKT conditions.

$$\begin{aligned} x_j &= \min\{\exp(a_j - t), \tfrac{s}{k}\}, \quad \forall j, \\ \nu_j &= \max\{0, a_j - t - \log(\tfrac{s}{k})\}, \quad \forall j, \\ 1 - s &= \exp(-t - \tfrac{p}{k}), \\ s &= \langle \mathbf{1}, x \rangle, \quad p = \langle \mathbf{1}, \nu \rangle . \end{aligned} \tag{6.21}$$

Next, we define the two index sets U and M as follows

$$U \triangleq \{j \,|\, x_j = \tfrac{s}{k}\}, \qquad M \triangleq \{j \,|\, x_j < \tfrac{s}{k}\}.$$

Note that the set U contains at most k indexes corresponding to the largest components of a_j. Now, we proceed with finding a t that solves (6.21). Let $\rho \triangleq \frac{|U|}{k}$. We eliminate p as

$$p = \sum_j \nu_j = \sum_U a_j - |U| \Big(t + \log(\tfrac{s}{k})\Big) \quad \Longrightarrow \quad \tfrac{p}{k} = \tfrac{1}{k} \sum_U a_j - \rho\Big(t + \log(\tfrac{s}{k})\Big).$$

Let $Z \triangleq \sum_M \exp a_j$, we write for s

$$s = \sum_j x_j = \sum_U \tfrac{s}{k} + \sum_M \exp(a_j - t) = \rho s + \exp(-t) \sum_M \exp a_j = \rho s + \exp(-t) Z.$$

We conclude that

$$(1-\rho)s = \exp(-t)Z \quad \Longrightarrow \quad t = \log Z - \log\Big((1-\rho)s\Big).$$

Let $A \triangleq \frac{1}{k} \sum_U a_j$. We further write

$$\begin{aligned} \log(1-s) &= -t - \tfrac{p}{k} = -t - A + \rho\Big(t + \log(\tfrac{s}{k})\Big) \\ &= \rho \log(\tfrac{s}{k}) - A - (1-\rho)\Big[\log Z - \log\Big((1-\rho)s\Big)\Big], \end{aligned}$$

which yields the following equation for s

$$\log(1-s) - \rho(\log s - \log k) + A + (1-\rho)\Big[\log Z - \log(1-\rho) - \log s\Big] = 0.$$

Therefore,

$$\begin{aligned} &\log(1-s) - \log s + \rho \log k + A + (1-\rho)\log Z - (1-\rho)\log(1-\rho) = 0, \\ &\log\left(\frac{1-s}{s}\right) = \log\left(\frac{(1-\rho)^{(1-\rho)} \exp(-A)}{k^\rho Z^{(1-\rho)}}\right). \end{aligned}$$

We finally get

$$\begin{aligned} s &= 1/(1+Q), \\ Q &\triangleq (1-\rho)^{(1-\rho)}/(k^\rho Z^{(1-\rho)} e^A). \end{aligned}$$

We note that: *a)* Q is readily computable once the sets U and M are fixed; and *b)* $Q = 1/Z$ if $k = 1$ since $\rho = A = 0$ in that case. This yields the formula for t as

$$t = \log Z + \log(1 + Q) - \log(1 - \rho).$$

As a sanity check, we note that we again recover the softmax loss for $k = 1$, since $t = \log Z + \log(1 + 1/Z) = \log(1 + Z) = \log(1 + \sum_j \exp a_j)$.

To verify that the computed s and t are compatible with the choice of the sets U and M, we check if this holds:

$$\begin{aligned} \exp(a_j - t) &\geq \tfrac{s}{k}, \quad && \forall j \in U, \\ \exp(a_j - t) &\leq \tfrac{s}{k}, \quad && \forall j \in M, \end{aligned}$$

which is equivalent to

$$\max_M a_j \leq \log(\tfrac{s}{k}) + t \leq \min_U a_j.$$

To compute the actual loss (6.19), we have

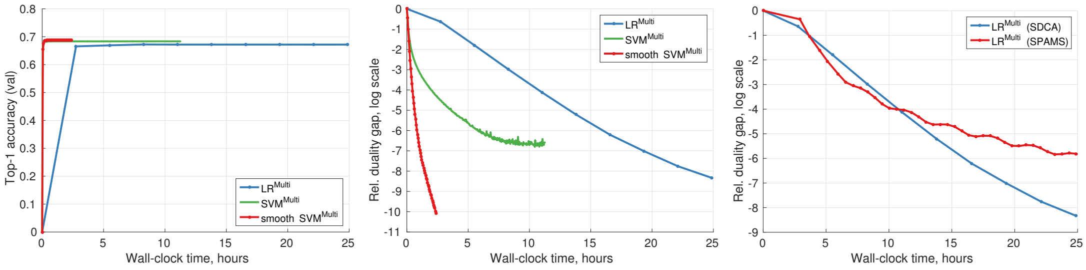
$$\begin{aligned} &\langle a, x \rangle - \langle x, \log x \rangle - (1 - s)\log(1 - s) \\ &= \sum_U a_j \tfrac{s}{k} + \sum_M a_j \exp(a_j - t) - \sum_U \tfrac{s}{k} \log(\tfrac{s}{k}) \\ &- \sum_M (a_j - t)\exp(a_j - t) - (1 - s)\log(1 - s) \\ &= As - \rho s \log(\tfrac{s}{k}) + t\exp(-t)Z - (1 - s)\log(1 - s) \\ &= As - \rho s \log(\tfrac{s}{k}) + (1 - \rho)st - (1 - s)\log(1 - s). \end{aligned}$$

□

As before, we only need to examine at most k partitions U, adding the next maximal a_j to U until there are no violated constraints. Therefore, the overall complexity of the procedure to compute the top-k Ent loss is $O(km)$.

The efficiency of the outlined approach for optimizing the top-k Ent loss crucially depends on fast computation of $V(t)$ in the SDCA update. Our implementation was able to scale to large datasets as we show next.

Runtime evaluation. Figure 6.2 compares the wall-clock training time of $\text{SVM}^{\text{Multi}}$ with a smoothed $\text{SVM}^{\text{Multi}}_{\gamma}$ and the LR^{Multi} objectives. We plot the validation accuracy (6.2a) and the relative duality gap (6.2b) versus time for the best performing models on the ImageNet 2012 benchmark. We obtain substantial improvement of the convergence rate for the smooth $\text{SVM}^{\text{Multi}}_{\gamma}$ compared to the non-smooth baseline. Moreover, we see that the top-1 accuracy saturates after a few passes over the training data, which justifies the use of a fairly loose stopping criterion (we use $\varepsilon = 10^{-3}$). For the LR^{Multi} loss, the cost of each epoch is significantly higher compared to $\text{SVM}^{\text{Multi}}$, which is due to the difficulty of solving (6.15). This suggests that the smooth top-1 SVM^{α}_{1} loss can offer competitive performance (see § 6.5) at a lower training cost.



(a) Top-1 accuracy vs. time (our solvers).

(b) Relative duality gap vs. time (our solvers).

(c) Relative duality gap vs. time (ours, SPAMS).

Figure 6.2.: SDCA convergence on ImageNet 2012. **(a-b)** Convergence of the $\mathrm{LR}^{\mathrm{Multi}}$, $\mathrm{SVM}^{\mathrm{Multi}}$, and smooth $\mathrm{SVM}_{\gamma}^{\mathrm{Multi}}$ methods. **(c)** SDCA vs. FISTA as implemented in the SPAMS toolbox of Mairal et al., (2010).

Finally, we also compare our implementation of $\mathrm{LR}^{\mathrm{Multi}}$ (marked SDCA in 6.2c) with the SPAMS optimization toolbox (Mairal et al., 2010), which provides an efficient implementation of FISTA (Beck and Teboulle, 2009). We note that the rate of convergence of SDCA is competitive with FISTA for $\epsilon \geq 10^{-4}$ and is noticeably better for $\epsilon < 10^{-4}$. We conclude that our approach for training the $\mathrm{LR}^{\mathrm{Multi}}$ model is competitive with the state-of-the-art, and faster computation of $V(t)$ can lead to a further speedup.

6.4.3 Multilabel Methods

This section covers optimization of the multilabel objectives introduced in § 6.2.3. First, we reduce computation of the SDCA update step and evaluation of the smoothed loss $\mathrm{SVM}_{\gamma}^{\mathrm{ML}}$ to the problem of computing the Euclidean projection onto what we called the bipartite simplex $B(r)$, see Eq. (6.8). Next, we contribute a novel variable fixing algorithm for computing that projection. Finally, we discuss SDCA optimization of the multilabel cross-entropy loss $\mathrm{LR}^{\mathrm{ML}}$.

SDCA update: $\mathrm{SVM}^{\mathrm{ML}}$, $\mathrm{SVM}_{\gamma}^{\mathrm{ML}}$. Here, we discuss optimization of the smoothed $\mathrm{SVM}_{\gamma}^{\mathrm{ML}}$ loss. The update step for the nonsmooth counterpart is recovered by setting $\gamma = 0$.

Proposition 6.18. Let L and L^* in (6.13) be respectively the $\mathrm{SVM}_{\gamma}^{\mathrm{ML}}$ loss and its conjugate as in Proposition 6.6. The dual variables $a \triangleq a_i$ corresponding to the training pair (x_i, Y_i) are updated as $(a_y)_{y \in Y_i} = p$ and $(a_j)_{j \in \bar{Y}_i} = -\bar{p}$, where

$$(p, \bar{p}) = \mathbf{proj}_{B(1/\lambda n)}(b, \bar{b}),$$

$b = \rho\left(\frac{1}{2} - q_y\right)_{y \in Y_i}$, $\bar{b} = \rho\left(\frac{1}{2} + q_j\right)_{j \in \bar{Y}_i}$, $q = W^\top x_i - \langle x_i, x_i \rangle\, a_i$, and $\rho = \frac{1}{\langle x_i, x_i \rangle + \gamma \lambda n}$.

Proof. We update the dual variables $a \triangleq a_i \in \mathbb{R}^m$ corresponding to the training example (x_i, Y_i) by solving the following optimization problem.

$$\max_{a \in \mathbb{R}^m} \; -\frac{1}{n} L^*_\gamma(Y_i, -\lambda n a) - \frac{\lambda}{2} \mathbf{tr}(AKA^\top),$$

where $\lambda > 0$ is a regularization parameter. Equivalently, we can divide both the primal and the dual objectives by λ and use $C \triangleq \frac{1}{\lambda n} > 0$ as the regularization parameter instead. The optimization problem becomes

$$\max_{a \in \mathbb{R}^m} \; -CL^*\Big(Y_i, -\frac{1}{C} a\Big) - \frac{1}{2} \mathbf{tr}(AKA^\top). \tag{6.22}$$

Note that

$$\mathbf{tr}(AKA^\top) = K_{ii} \langle a, a \rangle + 2 \sum_{j \neq i} K_{ij} \langle a_j, a \rangle + \text{const},$$

where the const does not depend on a. We ignore that constant in the following derivation and also define an auxiliary vector $q \triangleq \sum_{j \neq i} K_{ij} a_j = AK_i - K_{ii} a_i$. Plugging the conjugate from Proposition 6.6 into (6.22), we obtain

$$\begin{aligned} \max_{a \in \mathbb{R}^m} \; & -C\Big(\frac{1}{2C}\Big(-\textstyle\sum_{y \in Y_i} a_y + \sum_{j \in \bar{Y}_i} a_j\Big) + \frac{\gamma}{2C^2} \|a\|^2\Big) - (1/2)\Big(K_{ii} \|a\|^2 + 2 \langle q, a \rangle\Big) \\ \text{s.t.} \; & -\tfrac{1}{C} a \in S_{Y_i} \end{aligned}$$

We re-write the constraint $-\frac{1}{C} a \in S_{Y_i}$ as

$$\begin{aligned} &\textstyle\sum_{y \in Y_i} a_y = -\sum_{j \in \bar{Y}_i} a_j \leq C \\ &a_y \geq 0, \; \forall\, y \in Y_i; \quad a_j \leq 0, \; \forall\, j \in \bar{Y}_i; \end{aligned}$$

and switch to the equivalent minimization problem below.

$$\begin{aligned} \min_{a \in \mathbb{R}^m} \; & \tfrac{1}{2}\Big(K_{ii} + \tfrac{\gamma}{C}\Big) \|a\|^2 - \tfrac{1}{2} \textstyle\sum_{y \in Y_i} a_y - \tfrac{1}{2} \sum_{j \in \bar{Y}_i} (-a_j) + \langle q, a \rangle \\ & \textstyle\sum_{y \in Y_i} a_y = \sum_{j \in \bar{Y}_i} -a_j \leq C \\ & a_y \geq 0, \; \forall\, y \in Y_i; \quad -a_j \geq 0, \; \forall\, j \in \bar{Y}_i. \end{aligned}$$

Note that

$$-\tfrac{1}{2} \textstyle\sum_{y \in Y_i} a_y - \tfrac{1}{2} \sum_{j \in \bar{Y}_i} (-a_j) + \langle q, a \rangle = -\sum_{y \in Y_i} (\tfrac{1}{2} - q_y) a_y - \sum_{j \in \bar{Y}_i} (\tfrac{1}{2} + q_j)(-a_j),$$

and let us define

$$\begin{aligned} x &\triangleq (a_y)_{y \in Y_i} \in \mathbb{R}^{|Y_i|}, & b &\triangleq \tfrac{1}{K_{ii} + \gamma / C} (\tfrac{1}{2} - q_y)_{y \in Y_i} \in \mathbb{R}^{|Y_i|}, \\ y &\triangleq (-a_j)_{j \in \bar{Y}_i} \in \mathbb{R}^{|\bar{Y}_i|}, & \bar{b} &\triangleq \tfrac{1}{K_{ii} + \gamma / C} (\tfrac{1}{2} + q_j)_{j \in \bar{Y}_i} \in \mathbb{R}^{|\bar{Y}_i|}. \end{aligned}$$

The final projection problem for the update step is

$$\begin{aligned} \min_{x,y} \quad & \tfrac{1}{2}\left\|x - b\right\|^2 + \tfrac{1}{2}\left\|y - \bar{b}\right\|^2 \\ & \langle \mathbf{1}, x \rangle = \langle \mathbf{1}, y \rangle \leq C \\ & x \geq 0, \quad y \geq 0. \end{aligned} \tag{6.23}$$

□

Let us make two remarks regarding optimization of the multilabel SVM. First, we see that the update step involves exactly the same projection that was used in Proposition 6.6 to define the smoothed $\text{SVM}^{\text{ML}}_{\gamma}$ loss, with the difference in the vectors being projected and the radius of the bipartite simplex. Therefore, we can use the same projection algorithm both during optimization as well as when computing the loss. And second, even though SVM^{ML} reduces to $\text{SVM}^{\text{Multi}}$ when Y_i is singleton, the derivation of the smoothed loss and the projection algorithm proposed below for the bipartite simplex are substantially different from what we proposed in the multiclass setting. Most notably, the treatment of the dimensions in Y_i and $\bar{Y}_i$ is now symmetric.

Loss computation: $\text{SVM}^{\text{ML}}_{\gamma}$**.** The smooth multilabel SVM loss $\text{SVM}^{\text{ML}}_{\gamma}$ is

$$L_{\gamma}(u) = \tfrac{1}{\gamma}\Big(\langle b, p \rangle - \tfrac{1}{2}\left\|p\right\|^2 + \left\langle \bar{b}, \bar{p} \right\rangle - \tfrac{1}{2}\left\|\bar{p}\right\|^2 \Big),$$

where $b = \left(\frac{1}{2} - u_y\right)_{y \in Y}$, $\bar{b} = \left(\frac{1}{2} + u_j\right)_{j \in \bar{Y}}$, $u = f(x)$, and $(p, \bar{p}) = \mathbf{proj}_{B(\gamma)}(b, \bar{b})$. Below, we propose an efficient variable fixing algorithm to compute the Euclidean projection onto $B(\gamma)$. We also note that we can use the same trick that we used for top-k $\text{SVM}^{\alpha}_{\gamma}$ and exploit the special form of the projection to avoid explicit computation of p and $\bar{p}$.

Euclidean projection onto the bipartite simplex $B(\rho)$**.** The optimization problem that we seek to solve is:

$$\begin{aligned} (p, \bar{p}) = \mathop{\arg\min}_{x \in \mathbb{R}^m_+,\, y \in \mathbb{R}^n_+} \quad & \tfrac{1}{2}\left\|x - b\right\|^2 + \tfrac{1}{2}\left\|y - \bar{b}\right\|^2 \\ & \langle \mathbf{1}, x \rangle = \langle \mathbf{1}, y \rangle \leq \rho. \end{aligned} \tag{6.24}$$

This problem has been considered by Shalev-Shwartz and Singer, (2006), who proposed a breakpoint searching algorithm based on sorting, as well as by Liu and Ye, (2009), who formulated it as a root finding problem that is solved via bisection. Next, we contribute a novel variable fixing algorithm that is inspired by the algorithm of Kiwiel, (2008b) for the continuous quadratic knapsack problem (a.k.a. projection onto simplex).

1. *Initialization.* Define the sets $I_x = \{1, \dots, m\}$, $L_x = \{\}$, $I_y = \{1, \dots, n\}$, $L_y = \{\}$, and solve the independent subproblems below using (Kiwiel, 2008b).

$$p = \operatorname*{arg\,min}_{x \in \mathbb{R}^m_+} \left\{ \tfrac{1}{2} \left\| x - b \right\|^2 \mid \langle \mathbf{1}, x \rangle = \rho \right\},$$

$$\bar{p} = \operatorname*{arg\,min}_{y \in \mathbb{R}^n_+} \left\{ \tfrac{1}{2} \left\| x - \bar{b} \right\|^2 \mid \langle \mathbf{1}, y \rangle = \rho \right\}.$$

   Let t' and s' be the resulting optimal thresholds, such that $p = \max\{0, b - t'\}$ and $\bar{p} = \max\{0, \bar{b} - s'\}$. If $t' + s' \geq 0$, then $(p, \bar{p})$ is the solution to (6.24); stop.

2. *Restricted subproblem.* Compute t as

$$t = \left(\textstyle\sum_{I_x} b_j - \sum_{I_y} \bar{b}_j \right) / (|I_x| + |I_y|),$$

   and let $x_j(t) = b_j - t$, $y_j(t) = \bar{b}_j + t$.

3. *Feasibility check.* Compute

$$\Delta_x = \textstyle\sum_{I^L_x} (b_j - t), \text{ where } I^L_x = \{ j \in I_x \,|\, x_j(t) \leq 0 \},$$
$$\Delta_y = \textstyle\sum_{I^L_y} (\bar{b}_j + t), \text{ where } I^L_y = \{ j \in I_y \,|\, y_j(t) \leq 0 \}.$$

4. *Stopping criterion.* If $\Delta_x = \Delta_y$, then the solution to (6.24) is given by $p = \max\{0, b - t\}$ and $\bar{p} = \max\{0, \bar{b} + t\}$; stop.

5. *Variable fixing.* If $\Delta_x > \Delta_y$, update $I_x \leftarrow I_x \setminus I^L_x$, $L_x \leftarrow L_x \cup I^L_x$. If $\Delta_x < \Delta_y$, update $I_y \leftarrow I_y \setminus I^L_y$, $L_y \leftarrow L_y \cup I^L_y$. Go to step 2.

Proposition 6.19. The algorithm above solves (6.24).

Proof. We sketch the main parts of the proof to show correctness of the algorithm. A complete derivation would follow the proof given in Kiwiel, 2008b.

The Lagrangian for the optimization problem (6.24) is

$$\mathcal{L}(x, y, t, s, \lambda, \mu, \nu) = \tfrac{1}{2} \left\| x - b \right\|^2 + \tfrac{1}{2} \left\| y - \bar{b} \right\|^2$$
$$+ t(\langle \mathbf{1}, x \rangle - r) + s(\langle \mathbf{1}, y \rangle - r) + \lambda(r - \rho) - \langle \mu, x \rangle - \langle \nu, y \rangle,$$

and it leads to the following KKT conditions

$$\begin{aligned} x_j &= b_j - t + \mu_j, & \mu_j x_j &= 0, & \mu_j &\geq 0, \\ y_k &= \bar{b}_k - s + \nu_k, & \nu_k y_k &= 0, & \nu_k &\geq 0, \\ \lambda &= t + s, & \lambda(r - \rho) &= 0, & \lambda &\geq 0. \end{aligned} \tag{6.25}$$

If $\rho = 0$, the solution is trivial. Assume $\rho > 0$ and let

$$x(t) = \max\{0, b - t\}, \qquad y(s) = \max\{0, \bar{b} - s\},$$

where t, s are the dual variables from (6.25) and we have

$$(t+s)(r-\rho)=0, \qquad t+s\geq 0, \qquad 0\leq r\leq \rho.$$

We define index sets for x as

$$I_x=\{j\,|\,b_j-t>0\}, \qquad L_x=\{j\,|\,b_j-t\leq 0\}, \qquad m_x=\left|I_x\right|,$$

and similar sets I_y, L_y for y. Solving a reduced subproblem

$$\min\{\tfrac{1}{2}\left\|x-b\right\|^2 \mid \langle \mathbf{1},x\rangle = r\},$$

for t and a similar problem for s, yields

$$t=\tfrac{1}{m_x}\Big(\textstyle\sum_{j\in I_x} b_j - r\Big), \qquad s=\tfrac{1}{m_y}\Big(\textstyle\sum_{j\in I_y} \bar{b}_j - r\Big). \tag{6.26}$$

We consider two cases: $r=\rho$ and $r<\rho$. If $r=\rho$, then we have two variables t and s to optimize over, but the optimization problem (6.24) decouples into two simplex projection problems which can be solved independently.

$$\begin{aligned}&\min\{\tfrac{1}{2}\left\|x-b\right\|^2 \mid \langle \mathbf{1},x\rangle = \rho,\ x_j\geq 0\},\\ &\min\{\tfrac{1}{2}\left\|y-\bar{b}\right\|^2 \mid \langle \mathbf{1},y\rangle = \rho,\ y_j\geq 0\}.\end{aligned} \tag{6.27}$$

Let t' and s' be solutions to the independent problems (6.27). If $t'+s'\geq 0$, we have that the KKT conditions (6.25) are fulfilled and we have, therefore, the solution to the original problem (6.24). Otherwise, we have that the optimal $t^*+s^*>t'+s'$ and so at least one of the two variables must increase. Let $t^*>t'$, then $\langle \mathbf{1},x(t^*)\rangle < \langle \mathbf{1},x(t')\rangle = \rho$, therefore $r^*<\rho$.

If $r<\rho$, then $t+s=0$. We eliminate s, which leads to

$$\tfrac{1}{m_x}\Big(\textstyle\sum_{j\in I_x} b_j - r\Big) = -\tfrac{1}{m_y}\Big(\textstyle\sum_{j\in I_y} \bar{b}_j - r\Big).$$

This can now be solved for r as

$$r=\Big(m_y\textstyle\sum_{I_x} b_j + m_x\sum_{I_y}\bar{b}_j\Big)/(m_x+m_y). \tag{6.28}$$

One can verify that $r<\rho$ if r is computed by (6.28) and $t'+s'<0$. Plugging (6.28) into (6.26), we get

$$t=\Big(\textstyle\sum_{I_x} b_j - \sum_{I_y}\bar{b}_j\Big)/(m_x+m_y). \tag{6.29}$$

One can further verify that $t>t'$ and $-t>s'$, where t is computed by (6.29), t', s' are computed by (6.26) with $r=\rho$, and $t'+s'<0$. Therefore, if $x_j(t')=0$ for some $j\in L_x(t')$, then $x_j(t)=0$, and so $L_x(t')\subset L_x(t)$. The variables that were fixed to the lower bound while solving (6.27) with $r=\rho$ remain fixed when considering $r<\rho$. □

Dimension d	10^3	10^4	10^5	10^6	10^7
Sorting based (Shalev-Shwartz and Singer, 2006)	0.07	0.56	6.92	85.56	1364.94
Variable fixing (ours)	0.02	0.15	1.48	16.46	169.81
Improvement factor	3.07	3.79	4.69	5.20	8.04

Table 6.2.: Runtime (in seconds) for solving 1000 projection problems onto $B(\rho)$ with $\rho = 10$ and $m = n = d/2$, see Eq. (6.24). The data is i.i.d. $\mathcal{N}(0, \mathbf{I})$.

The proposed algorithm is easy to implement, does not require sorting, and scales well in practice, as demonstrated by our experiments in § 6.5.3.

Runtime evaluation. We also compare the runtime of the proposed variable fixing algorithm and the sorting based algorithm of Shalev-Shwartz and Singer, (2006). We perform no comparison to (Liu and Ye, 2009) as their code is not available. Furthermore, the algorithms that we consider are exact, while the method of Liu and Ye, (2009) is approximate and its runtime is dependent on the required precision. The experimental setup is the same as in § 5.5.1 (page 100), and our results are reported in Table 6.2.

We observe consistent improvement in runtime over the sorting based implementation, and we use our algorithm to train $\text{SVM}_\gamma^{\text{ML}}$ in further experiments.

SDCA update: LR^{ML}. Finally, we discuss optimization of the multilabel cross-entropy loss LR^{ML}. We show that the corresponding SDCA update step is equivalent to a certain entropic projection problem, which we propose to tackle using the $V(t)$ function introduced above.

Proposition 6.20. Let L and L^* in (6.13) be respectively the LR^{ML} loss and its conjugate from Proposition 6.7. The dual variables $a \triangleq a_i$ corresponding to the training pair (x_i, Y_i) are updated as $(a_y)_{y \in Y_i} = -\frac{1}{\lambda n}\big(p - \frac{1}{k}\big)$ and $(a_j)_{j \in \bar{Y}_i} = -\frac{1}{\lambda n}\bar{p}$, where

$$\begin{aligned}(p, \bar{p}) = \operatorname*{arg\,min}_{x \geq 0,\, y \geq 0} \; & \tfrac{\alpha}{2}\|x - b\|^2 + \langle x, \log x \rangle + \tfrac{\alpha}{2}\|y - \bar{b}\|^2 + \langle y, \log y \rangle\,, \\ \text{s.t.} \;\; & \langle \mathbf{1}, x \rangle + \langle \mathbf{1}, y \rangle = 1,\end{aligned} \tag{6.30}$$

$k = |Y_i|$, $\alpha = \frac{\langle x_i, x_i \rangle}{\lambda n}$, $b = \big(\frac{1}{\alpha} q_j + \frac{1}{k}\big)_{j \in Y_i}$, $\bar{b} = \big(\frac{1}{\alpha} q_j\big)_{j \in \bar{Y}_i}$, and $q = W^\top x_i - \langle x_i, x_i \rangle\, a_i$.

Moreover, the solution of (6.30) is given by

$$p_j = \tfrac{1}{\alpha} V(\alpha b_j - t),\ \forall\, j, \qquad\qquad \bar{p}_j = \tfrac{1}{\alpha} V(\alpha \bar{b}_j - t),\ \forall\, j,$$

where t is computed from

$$\textstyle\sum_{j \in Y_i} V(q_j + \frac{\alpha}{k} - t) + \sum_{j \in \bar{Y}_i} V(q_j - t) = \alpha. \tag{6.31}$$

Proof. Let $q \triangleq \sum_{j \neq i} K_{ij} a_j = AK_i - K_{ii} a_i$ and $C \triangleq \frac{1}{\lambda n}$, as before. We solve

$$\max_{a \in \mathbb{R}^m} \; -CL^*\Big(Y_i, -\frac{1}{C}a\Big) - \tfrac{1}{2}\Big(K_{ii}\|a\|^2 + 2\langle q, a\rangle\Big).$$

Let x and y be defined as

$$\begin{cases} x = \big(-\tfrac{1}{C}a_j + \tfrac{1}{k}\big)_{j \in Y_i} \\ y = \big(-\tfrac{1}{C}a_j\big)_{j \in \bar{Y}_i}, \end{cases} \qquad \Longrightarrow \qquad \begin{cases} a_j = -C\big(x_j - \tfrac{1}{k}\big), \\ a_j = -Cy_j. \end{cases}$$

We have that

$$K_{ii}\|a\|^2 + 2\langle q, a\rangle = K_{ii}C^2\Big(\Big\|x - \tfrac{1}{k}\mathbf{1}\Big\|^2 + \|y\|^2\Big) - 2C\Big(\Big\langle q_{Y_i}, x - \tfrac{1}{k}\mathbf{1}\Big\rangle + \langle q_{\bar{Y}_i}, y\rangle\Big).$$

Ignoring the constant terms and switching the sign, we obtain

$$\begin{aligned} \min_{x \geq 0,\, y \geq 0} \quad & \langle x, \log x\rangle + \tfrac{1}{2}K_{ii}C\|x\|^2 - K_{ii}C\tfrac{1}{k}\langle \mathbf{1}, x\rangle - \Big\langle q_{Y_i}, x\Big\rangle \\ & \langle y, \log y\rangle + \tfrac{1}{2}K_{ii}C\|y\|^2 - \Big\langle q_{\bar{Y}_i}, y\Big\rangle \\ \text{s.t.} \quad & \langle \mathbf{1}, x\rangle + \langle \mathbf{1}, y\rangle = 1 \end{aligned}$$

Let $\alpha \triangleq K_{ii}C$ and define

$$b_j = \tfrac{1}{\alpha}q_j + \tfrac{1}{k},\ j \in Y_i, \qquad\qquad \bar{b}_j = \tfrac{1}{\alpha}q_j,\ j \in \bar{Y}_i.$$

The final proximal problem for the update step is given as

$$\begin{aligned} \min_{x \geq 0,\, y \geq 0} \quad & \langle x, \log x\rangle + \tfrac{\alpha}{2}\|x - b\|^2 + \langle y, \log y\rangle + \tfrac{\alpha}{2}\Big\|y - \bar{b}\Big\|^2 \\ & \langle \mathbf{1}, x\rangle + \langle \mathbf{1}, y\rangle = 1. \end{aligned}$$

Next, we discuss how to solve (6.30). The Lagrangian for this problem is

$$\begin{aligned} \mathcal{L}(x, y, \lambda, \mu, \nu) = {} & \langle x, \log x\rangle + \tfrac{\alpha}{2}\|x - b\|^2 + \langle y, \log y\rangle + \tfrac{\alpha}{2}\|y - \bar{b}\|^2 \\ & + \lambda(\langle \mathbf{1}, x\rangle + \langle \mathbf{1}, y\rangle - 1) - \langle \mu, x\rangle - \langle \nu, y\rangle\,. \end{aligned}$$

Setting the partial derivatives to zero, we obtain

$$\begin{aligned} \log x_j + \alpha x_j &= \alpha b_j - \lambda - 1 + \mu_j, \\ \log y_j + \alpha y_j &= \alpha \bar{b}_j - \lambda - 1 + \nu_j. \end{aligned}$$

We $x_j > 0$ and $y_j > 0$, which implies $\mu_j = 0$ and $\nu_j = 0$.

$$\begin{aligned} \log(\alpha x_j) + \alpha x_j &= \alpha b_j - \lambda - 1 + \log \alpha, \\ \log(\alpha y_j) + \alpha y_j &= \alpha \bar{b}_j - \lambda - 1 + \log \alpha. \end{aligned}$$

Let $t \triangleq \lambda + 1 - \log \alpha$, we have

$$\begin{aligned}\alpha x_j &= W(\exp(\alpha b_j - t)) = V(\alpha b_j - t),\\ \alpha y_j &= W(\exp(\alpha \bar{b}_j - t)) = V(\alpha \bar{b}_j - t),\end{aligned}$$

where W is the Lambert W function. Let

$$g(t) = \sum_{j \in Y_i} V(b_j + \tfrac{\alpha}{k} - t) + \sum_{j \in \bar{Y}_i} V(b_j - t) - \alpha,$$

then the optimal t^* is the root of $g(t) = 0$, which corresponds to the constraint $\langle \mathbf{1}, x \rangle + \langle \mathbf{1}, y \rangle = 1$. □

We use a 4-th order Householder method (Householder, 1970) to solve (6.31), similar to the top-k Ent loss above. Solving the nonlinear equation in t is the main computational challenge when updating the dual variables. However, as this procedure does not require iteration over the index partitions, it is generally faster than optimization of the top-k Ent loss.

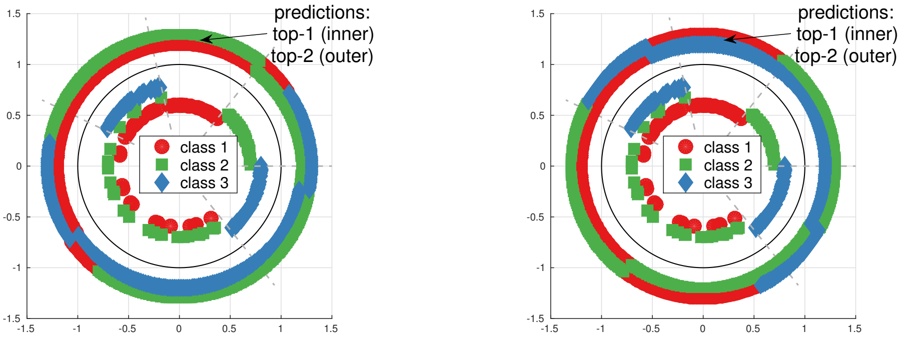
6.5 Experiments

This section provides a broad array of experiments on 24 different datasets comparing multiclass and multilabel performance of the 13 loss functions from § 6.2. We look at different aspects of empirical evaluation: performance on synthetic and real data, use of handcrafted features and the features extracted from a ConvNet, targeting a specific performance measure and being generally competitive over a range of metrics.

- In § 6.5.1, we show on synthetic data that the top-k Ent_{tr} loss targeting the top-2 error outperforms all competing methods by a large margin.
- In § 6.5.2, we focus on evaluating top-k performance of multiclass methods on 11 real-world benchmark datasets including ImageNet and Places.
- In § 6.5.3, we cover multilabel classification in two groups of experiments: (i) a comparative study following Madjarov et al., (2012) on 10 popular multilabel datasets; (ii) image classification on Pascal VOC and MS COCO in a novel setting contrasting multiclass, top-k, and multilabel methods.

6.5.1 Synthetic Example

In this section, we demonstrate in a synthetic experiment that our proposed top-2 losses outperform the top-1 losses when the aim is optimal top-2 performance. The dataset with three classes is shown in the inner circle of Figure 6.3.



(a) top-1 SVM$_1^\alpha$ test accuracy (top-1, 2): 65.7%, 81.3%

(b) top-2 Ent$_{\text{tr}}$ test accuracy (top-1, 2): 29.4%, 96.1%

Figure 6.3.: Synthetic data in $\mathbb{R}^2$ (color markers inside of the black circle) and visualization of top-1 and top-2 predictions (outside of the circle).
(a) top-1 SVM$_1^\alpha$ optimizes the top-1 error which increases its top-2 error.
(b) top-2 Ent$_{\text{tr}}$ ignores the top-1 and optimizes directly the top-2 error.

Sampling scheme. First, we generate samples in $[0, 7] \subset \mathbb{R}$ which is subdivided into 5 segments. All segments have unit length, except for the 4-th segment which has length 3. We sample in each of the 5 segments according to the following distribution: $(0, 1, .4, .3, 0)$ for class 1; $(1, 0, .1, .7, 0)$ for class 2; $(0, 0, .5, 0, 1)$ for class 3. Finally, the data is rescaled to $[0, 1]$ and mapped onto the unit circle.

Samples of different classes are plotted next to each other for better visibility as there is significant class overlap. We visualize top-1/2 predictions with two colored circles outside of the black circle. We sample 200/200/200K points for training/validation/test and tune $C = 1/(\lambda n)$ in the range 2^{-18} to 2^{18}. Results are shown in Table 6.3.

Circle (synthetic)

Method	Top-1	Top-2	Method	Top-1	Top-2
SVM$^{\text{OVA}}$	54.3	85.8	top-1 SVM$_1$	**65.7**	83.9
LR$^{\text{OVA}}$	54.7	81.7	top-2 SVM$_{0/1}$	54.4 / 54.5	87.1 / 87.0
SVM$^{\text{Multi}}$	58.9	89.3	top-2 Ent	54.6	87.6
LR$^{\text{Multi}}$	54.7	81.7	top-2 Ent$_{\text{tr}}$	58.4	**96.1**

Table 6.3.: Top-k accuracy (%) on synthetic data. **Left:** Baseline methods.
Right: Top-k SVM (nonsmooth / smooth) and the top-k entropy losses.

In each column, we provide the results for the model (as determined by the hyperparameter C) that optimizes the corresponding top-k accuracy. First, we note that all top-1 baselines perform similar in top-1 performance, except for SVM$^{\text{Multi}}$ and top-1 SVM$_1$ which show better results. Next, we see that our top-2 losses improve the top-2 accuracy and the improvement is most significant for the nonconvex top-2 Ent$_{\text{tr}}$ loss, which is close to the optimal solution for this dataset. This is because top-2 Ent$_{\text{tr}}$ provides a tight bound on the top-2 error and ignores

Dataset	m	n	d	Dataset	m	n	d
ALOI (Rocha and Goldenstein, 2014)	1K	54K	128	**Indoor 67** (Quattoni and Torralba, 2009)	67	5354	4K
Caltech 101 Silhouettes (Swersky et al., 2012)	101	4100	784	**Letter** (Hsu and Lin, 2002)	26	10.5K	16
CUB (Wah et al., 2011)	202	5994	4K	**News 20** (Lang, 1995)	20	15.9K	16K
Flowers (Nilsback and Zisserman, 2008)	102	2040	4K	**Places 205** (Zhou et al., 2014)	205	2.4M	4K
FMD (Sharan et al., 2009)	10	500	4K	**SUN 397** (Xiao et al., 2010)	397	19.9K	4K
ImageNet 2012 (Russakovsky et al., 2015)	1K	1.3M	4K				

Table 6.4.: Statistics of multiclass classification benchmarks (m: # classes, n: # training examples, d: # feature dimensions).

the top-1 errors in the loss. Unfortunately, similar significant improvements are not observed on the real-world datasets that we tried. This might be due to the high dimension of the feature spaces, which yields well separable problems.

6.5.2 Multiclass Experiments

The goal of this section is to provide an extensive empirical evaluation of the loss functions from § 6.2.2 in terms of top-k performance. To that end, we compare multiclass and top-k methods on 11 datasets ranging in size (500 to 2.4M training examples, 10 to 1000 classes), problem domain (vision, non-vision), and granularity (scene, object, and fine-grained classification). The statistics of the datasets is given in Table 6.4. We also report the reference performance from the literature, and use the encoding scheme given in Table 6.5 in the interest of space.

C	(Cimpoi et al., 2015)	**HL**	(Hsu and Lin, 2002)
RA	(Razavian et al., 2014)	**R**	(Rennie, 2001)
RG	(Rocha and Goldenstein, 2014)	**S**	(Swersky et al., 2012)
SZ	(Simonyan and Zisserman, 2015)	**WG**	(Wang et al., 2015a)

Table 6.5.: Encoding of the reference methods for the results from the literature.

Solvers. We use LibLinear (Fan et al., 2008) for the one-vs-all baselines SVM$^{\text{OVA}}$ and LR$^{\text{OVA}}$; and our code from Chapter 5 for top-k SVM. We extended the latter to support the smooth top-k SVM$_\gamma$ and the top-k Ent losses. The multiclass baselines SVM$^{\text{Multi}}$ and LR$^{\text{Multi}}$ correspond respectively to top-1 SVM and top-1 Ent. For the nonconvex top-k Ent$_{\text{tr}}$, we use the LR$^{\text{Multi}}$ solution as an initial point and perform gradient descent with line search (Nocedal and Wright, 2006). We cross-validate hyper-parameters in the range 10^{-5} to 10^3, extending it when the optimal value is at the boundary.

	ALOI				Letter				News 20				Caltech 101 Sil			
Reference:	Top-1: 93 ± 1.2 **RG**				Top-1: 97.98 **HL**				Top-1: 86.9 **R**				62.1	79.6	83.4	**S**
Top-k:	1	3	5	10	1	3	5	10	1	3	5	10	1	3	5	10
SVM$^{\text{OVA}}$	82.4	89.5	91.5	93.7	63.0	82.0	88.1	94.6	84.3	95.4	97.9	**99.5**	61.8	76.5	80.8	86.6
LR$^{\text{OVA}}$	86.1	93.0	94.8	96.6	68.1	86.1	90.6	96.2	84.9	96.3	97.8	99.3	63.2	80.4	84.4	89.4
SVM$^{\text{Multi}}$	90.0	95.1	96.7	98.1	76.5	89.2	93.1	97.7	85.4	94.9	97.2	99.1	62.8	77.8	82.0	86.9
LR$^{\text{Multi}}$	89.8	95.7	97.1	98.4	75.3	90.3	94.3	98.0	84.5	96.4	98.1	**99.5**	63.2	**81.2**	85.1	89.7
top-3 SVM	89.2	95.5	97.2	98.4	74.0	91.0	94.4	97.8	85.1	96.6	98.2	99.3	63.4	79.7	83.6	88.3
top-5 SVM	87.3	95.6	97.4	98.6	70.8	**91.5**	95.1	98.4	84.3	96.7	98.4	99.3	63.3	80.0	84.3	88.7
top-10 SVM	85.0	95.5	97.3	**98.7**	61.6	88.9	96.0	99.6	82.7	96.5	98.4	99.3	63.0	80.5	84.6	89.1
top-1 SVM$_1$	**90.6**	95.5	96.7	98.2	**76.8**	89.9	93.6	97.6	**85.6**	96.3	98.0	99.3	**63.9**	80.3	84.0	89.0
top-3 SVM$_1$	89.6	95.7	97.3	98.4	74.1	90.9	94.5	97.9	85.1	96.6	98.4	99.4	63.3	80.1	84.0	89.2
top-5 SVM$_1$	87.6	95.7	**97.5**	98.6	70.8	**91.5**	95.2	98.6	84.5	96.7	98.4	99.4	63.3	80.5	84.5	89.1
top-10 SVM$_1$	85.2	95.6	97.4	**98.7**	61.7	89.1	95.9	**99.7**	82.9	96.5	98.4	**99.5**	63.1	80.5	84.8	89.1
top-3 Ent	89.0	95.8	97.2	98.4	73.0	90.8	94.9	98.5	84.7	96.6	98.3	99.4	63.3	81.1	85.0	89.9
top-5 Ent	87.9	95.8	97.2	98.4	69.7	90.9	95.1	98.8	84.3	**96.8**	**98.6**	99.4	63.2	80.9	85.2	89.9
top-10 Ent	86.0	95.6	97.3	98.5	65.0	89.7	**96.2**	99.6	82.7	96.4	98.5	99.4	62.5	80.8	**85.4**	90.1
top-3 Ent$_{\text{tr}}$	89.3	**95.9**	97.3	98.5	63.6	91.1	95.6	98.8	83.4	96.4	98.3	99.4	60.7	81.1	85.2	**90.2**
top-5 Ent$_{\text{tr}}$	87.9	95.7	97.3	98.6	50.3	87.7	96.1	99.4	83.2	96.0	98.2	99.4	58.3	79.8	85.2	**90.2**
top-10 Ent$_{\text{tr}}$	85.2	94.8	97.1	98.5	46.5	80.9	93.7	99.6	82.9	95.7	97.9	99.4	51.9	78.4	84.6	**90.2**

Table 6.6.: Top-k accuracy evaluation. We compare the OVA and multiclass baselines with the top-k SVM$^{\alpha}$ from Chapter 5, as well as the proposed smooth top-k SVM$^{\alpha}_{\gamma}$, top-k Ent, and the nonconvex top-k Ent$_{\text{tr}}$.

Features. For ALOI, Letter, and News 20 datasets, we use the features provided by the LibSVM (Chang and Lin, 2011) datasets. For ALOI, we randomly split the data into equally sized training and test sets preserving class distributions. The Letter dataset comes with a separate validation set, which we used for model selection only. For News20, we use PCA to reduce dimensionality of sparse features from 62060 to 15478 preserving all non-singular PCA components[3].

For Caltech101 Silhouettes, we use the features and the train/val/test splits provided by Swersky et al., (2012).

For CUB, Flowers, FMD, and ImageNet 2012, we use MatConvNet (Vedaldi and Lenc, 2015) to extract the outputs of the last fully connected layer of the VGGNet-16 model (Simonyan and Zisserman, 2015).

For Indoor 67, SUN 397, and Places 205, we perform the same feature extraction, but use the VGGNet-16 model of Wang et al., (2015a) trained on Places 205.

Discussion. The results are given in Tables 6.6, 6.7, and we can make several interesting observations. First, while the OVA schemes perform quite similar to the multiclass approaches (OVA LR vs. softmax, OVA SVM vs. multiclass SVM), which confirms earlier observations in (Akata et al., 2014; Rifkin and Klautau,

[3] Our SDCA-based solvers are designed for dense inputs.

	Indoor 67				**CUB**				**Flowers**				**FMD**		
Top-1:	82.0 **WG**				62.8 **C**				86.8 **RA**				77.4 **C**		
Top-k:	1	3	5	10	1	3	5	10	1	3	5	10	1	3	5
SVM$^{\text{OVA}}$	81.9	94.3	96.5	98.0	60.6	77.1	83.4	89.9	82.0	91.7	94.3	96.8	77.4	92.4	96.4
LR$^{\text{OVA}}$	82.0	94.9	97.2	98.7	62.3	80.5	87.4	93.5	82.6	92.2	94.8	97.6	79.6	94.2	**98.2**
SVM$^{\text{Multi}}$	82.5	**95.4**	97.3	99.1	61.0	79.2	85.7	92.3	82.5	92.2	94.8	96.4	77.6	93.8	97.2
LR$^{\text{Multi}}$	82.4	95.2	**98.0**	99.1	62.3	81.7	87.9	**93.9**	82.9	92.4	95.1	97.8	79.0	94.6	97.8
top-3 SVM	81.6	95.1	97.7	99.0	61.3	80.4	86.3	92.5	81.9	92.2	95.0	96.1	78.8	94.6	97.8
top-5 SVM	79.9	95.0	97.7	99.0	60.9	81.2	87.2	92.9	81.7	92.4	95.1	97.8	78.4	94.4	97.6
top-10 SVM	78.4	95.1	97.4	99.0	59.6	81.3	87.7	93.4	80.5	91.9	95.1	97.7			
top-1 SVM$_1$	**82.6**	95.2	97.6	99.0	61.9	80.2	86.9	93.1	**83.0**	92.4	95.1	97.6	78.6	93.8	98.0
top-3 SVM$_1$	81.6	95.1	97.8	99.0	61.9	81.1	86.6	93.2	82.5	92.3	95.2	97.7	79.0	94.4	98.0
top-5 SVM$_1$	80.4	95.1	97.8	99.1	61.3	81.3	87.4	92.9	82.0	**92.5**	95.1	97.8	79.4	94.4	97.6
top-10 SVM$_1$	78.3	95.1	97.5	99.0	59.8	81.4	87.8	93.4	80.6	91.9	95.1	97.7			
top-3 Ent	81.4	**95.4**	97.6	**99.2**	**62.5**	81.8	87.9	**93.9**	82.5	92.0	**95.3**	97.8	**79.8**	94.8	98.0
top-5 Ent	80.3	95.0	97.7	99.0	62.0	**81.9**	88.1	93.8	82.1	92.2	95.1	**97.9**	79.4	94.4	98.0
top-10 Ent	79.2	95.1	97.6	99.0	61.2	81.6	**88.2**	93.8	80.9	92.1	95.0	97.7			
top-3 Ent$_{\text{tr}}$	79.8	95.0	97.5	99.1	62.0	81.4	87.6	93.4	82.1	92.2	95.2	97.6	78.4	**95.4**	**98.2**
top-5 Ent$_{\text{tr}}$	76.4	94.3	97.3	99.0	61.4	81.2	87.7	93.7	81.4	92.0	95.0	97.7	77.2	94.0	97.8
top-10 Ent$_{\text{tr}}$	72.6	92.8	97.1	98.9	59.7	80.7	87.2	93.4	77.9	91.1	94.3	97.3			

	SUN 397				**Places 205**				**ImageNet 2012**			
Reference:	Top-1: 66.9 **WG**				60.6		88.5	**WG**	76.3		93.2	**SZ**
Top-k:	1	3	5	10	1	3	5	10	1	3	5	10
SVM$^{\text{Multi}}$	65.8	85.1	90.8	95.3	58.4	78.7	84.7	89.9	68.3	82.9	87.0	91.1
LR$^{\text{Multi}}$	**67.5**	**87.7**	**92.9**	**96.8**	59.0	**80.6**	**87.6**	**94.3**	67.2	83.2	87.7	92.2
top-3 SVM	66.5	86.5	91.8	95.9	58.6	80.3	87.3	93.3	68.2	84.0	88.1	92.1
top-5 SVM	66.3	87.0	92.2	96.3	58.4	80.5	87.4	94.0	67.8	**84.1**	88.2	92.4
top-10 SVM	64.8	87.2	92.6	96.6	58.0	80.4	87.4	**94.3**	67.0	83.8	88.3	**92.6**
top-1 SVM$_1$	67.4	86.8	92.0	96.1	**59.2**	80.5	87.3	93.8	**68.7**	83.9	88.0	92.1
top-3 SVM$_1$	67.0	87.0	92.2	96.2	58.9	80.5	**87.6**	93.9	68.2	**84.1**	88.2	92.3
top-5 SVM$_1$	66.5	87.2	92.4	96.3	58.5	80.5	87.5	94.1	67.9	**84.1**	**88.4**	92.5
top-10 SVM$_1$	64.9	87.3	92.6	96.6	58.0	80.4	87.5	**94.3**	67.1	83.8	88.3	**92.6**
top-3 Ent	67.2	**87.7**	**92.9**	**96.8**	58.7	**80.6**	**87.6**	94.2	66.8	83.1	87.8	92.2
top-5 Ent	66.6	**87.7**	**92.9**	**96.8**	58.1	80.4	87.4	94.2	66.5	83.0	87.7	92.2
top-10 Ent	65.2	87.4	92.8	**96.8**	57.0	80.0	87.2	94.1	65.8	82.8	87.6	92.1

Table 6.7.: Top-k accuracy evaluation. We compare the OVA and multiclass baselines with the top-k SVM$^\alpha$ from Chapter 5, as well as the proposed smooth top-k SVM$^\alpha_\gamma$, top-k Ent, and the nonconvex top-k Ent$_{\text{tr}}$.

	Places 205				ImageNet 2012			
Top-k:	Top-1	Top-3	Top-5	Top-10	Top-1	Top-3	Top-5	Top-10
LR$^{\text{Multi}}$	59.97	81.39	88.17	94.59	68.60	84.29	88.66	92.83
top-3 SVM$_1$ (FT)	60.73	82.09	88.58	94.56	71.66	86.63	90.55	94.17
top-5 SVM$_1$ (FT)	**60.88**	**82.18**	**88.78**	94.75	71.60	86.67	90.56	94.23
top-3 Ent$_{\text{tr}}$ (FT)	60.51	81.86	88.69	94.78	71.41	86.80	90.77	94.35
top-5 Ent$_{\text{tr}}$ (FT)	60.48	81.66	88.66	94.80	71.20	86.57	90.75	**94.38**
LR$^{\text{Multi}}$ (FT)	60.73	82.07	88.71	**94.82**	**72.11**	**87.08**	**90.88**	**94.38**

Table 6.8.: Top-k accuracy, as reported by Caffe (Jia et al., 2014), after fine-tuning (FT) for approximately one epoch on Places and 3 epochs on ImageNet. The first line (LR$^{\text{Multi}}$) is the reference performance w/o fine-tuning.

2004), the OVA schemes performed worse on ALOI and Letter. Thus, we generally recommend the multiclass losses instead of the OVA schemes.

Comparing the softmax loss and multiclass SVM, we see that there is no clear winner in top-1 performance, but softmax consistently outperforms multiclass SVM in top-k performance for $k > 1$. This might be due to the strong property of softmax being top-k calibrated for all k. Note that this trend is uniform across all datasets, in particular, also for the ones where the features are not coming from a ConvNet. Both the smooth top-k SVM and the top-k entropy losses perform slightly better than softmax if one compares specific top-k errors. However, the good performance of the truncated top-k entropy loss on synthetic data did not transfer to the real world datasets.

Fine-tuning experiments. We also performed a number of fine-tuning experiments where the original network was trained further for 1-3 epochs with the smooth top-k hinge and the truncated top-k entropy losses[4]. The motivation was to see if the full end-to-end training would be more beneficial compared to training just the classifier. Results are reported in Table 6.8. We should note that the setting is now slightly different: there is no feature extraction step with the MatConvNet and there is a non-regularized bias term in Caffe (Jia et al., 2014). We see that the top-k specific losses are able to improve the performance compared to the reference model, and that, on Places 205, the smooth top-5 SVM$_1$ loss achieves the best top-1..5 performance. However, in this set of experiments, we also observed similar improvements when fine-tuning with the standard softmax loss, which achieves the best performance on ImageNet 2012. Further training beyond 3 epochs did not change the results significantly.

Conclusion. We see that a safe choice for multiclass problems seems to be the LR$^{\text{Multi}}$ loss as it yields reasonably good results in all top-k errors. A competitive alternative is the smooth SVM$_\gamma^{\text{Multi}}$ loss which can be faster to train (see the runtime experiments in § 6.4.2). If one wants to optimize directly for a top-k error at the

[4] Code: https://github.com/mlapin/caffe/tree/topk

Dataset	m	n	d	l_c	Dataset	m	n	d	l_c
bibtex (Katakis et al., 2008)	159	5K	2K	2.40	**enron** (Klimt and Yang, 2004)	53	1K	1K	3.38
bookmarks (Katakis et al., 2008)	208	60K	2K	2.03	**mediamill** (Snoek et al., 2006)	101	31K	120	4.38
corel5k (Duygulu et al., 2002)	374	4.5K	499	3.52	**medical** (Read et al., 2009)	45	645	1.5K	1.25
delicious (Tsoumakas et al., 2008)	983	13K	500	19.02	**scene** (Boutell et al., 2004)	6	1.2K	294	1.07
emotions (Trohidis et al., 2008)	6	391	72	1.87	**yeast** (Elisseeff and Weston, 2001)	14	1.5K	103	4.24
VOC 2007 (Everingham et al., 2010)	20	5K	2K	1.46	**MS COCO** (Lin et al., 2014)	80	83K	2K	2.91

Table 6.9.: Statistics of multilabel benchmarks (m: # classes, n: # training examples, d: # feature dimensions, l_c: label cardinality).

cost of a higher top-1 error, then further improvements are possible using either the smooth top-k SVM or the top-k entropy losses.

6.5.3 Multilabel Experiments

The aim of this section is threefold. First, we establish competitive performance of our multilabel classification methods from § 6.2.3 comparing them to the top 3 methods from an extensive experimental study by Madjarov et al., (2012) on 10 multilabel benchmark datasets of varying scale and complexity. Next, we discuss an interesting learning setting when top-k classification methods emerge as a transition step between multiclass and multilabel approaches. Finally, we evaluate multiclass, top-k, and multilabel classification methods on Pascal VOC 2007 (Everingham et al., 2010) and the more challenging Microsoft COCO (Lin et al., 2014) image classification benchmarks.

Multilabel classification. Here, establish a solid baseline to evaluate our implementation of the multilabel SVM^{ML}, smooth $\text{SVM}^{\text{ML}}_{\gamma}$, and the LR^{ML} methods. We follow the work of Madjarov et al., (2012) who provide a clear description of the evaluation protocol and an extensive experimental comparison of 12 multilabel classification methods on 11 datasets reporting 16 performance metrics. We limit our comparison to the 3 best performing methods from their study, namely: (i) the random forest of predicting clustering trees (Kocev et al., 2007), (ii) the hierarchy of multilabel classifiers (Tsoumakas et al., 2008), and (iii) the binary relevance method using SVM^{OVA}. We report results on 10 datasets as there was an issue with the published train/test splits on the remaining benchmark[5]. The datasets vary greatly in size and **label cardinality** (the average number of labels per example),

[5] See https://github.com/tsoumakas/mulan/issues/4 for details.

	bibtex							**bookmarks**						
Method	RLoss	HLoss	Acc	SAcc	F_1^{mic}	F_1^{mac}	F_1^{inst}	RLoss	HLoss	Acc	SAcc	F_1^{mic}	F_1^{mac}	F_1^{inst}
RF-PCT	0.093	0.013	16.6	9.8	23.0	5.5	21.2	0.104	0.009	20.4	18.9	23.6	10.1	21.3
HOMER	0.255	0.014	33.0	16.5	42.9	26.6	42.6	-	-	-	-	-	-	-
BR (RBF)	0.068	**0.012**	34.8	**19.4**	45.7	30.7	43.3	-	-	-	-	-	-	-
LR^{ML}	**0.053**	0.013	30.9	14.2	42.5	35.0	38.8	0.079	0.009	22.5	16.5	29.5	21.7	27.0
SVM^{ML}	0.094	0.013	28.6	13.2	40.6	31.5	36.1	0.140	0.009	24.0	19.8	27.5	18.4	26.7
SVM$^{ML}_{\gamma}$	0.073	0.013	31.4	16.2	43.5	33.5	39.3	0.091	0.009	28.0	20.7	34.0	22.6	32.4
LR^{ML} (RBF)	0.054	0.013	33.8	14.6	45.4	31.8	42.0	**0.072**	0.009	25.1	19.7	33.1	24.6	29.0
SVM^{ML} (RBF)	0.067	0.013	36.2	19.0	**46.5**	37.1	**44.6**	0.103	0.009	30.3	22.9	35.8	26.0	34.9
SVM$^{ML}_{\gamma}$ (RBF)	0.067	**0.012**	**36.6**	18.4	**46.5**	**37.2**	**44.6**	0.079	**0.008**	**31.8**	**23.0**	**38.0**	**28.0**	**36.7**

	corel5k							**delicious**						
Method	RLoss	HLoss	Acc	SAcc	F_1^{mic}	F_1^{mac}	F_1^{inst}	RLoss	HLoss	Acc	SAcc	F_1^{mic}	F_1^{mac}	F_1^{inst}
RF-PCT	0.117	**0.009**	0.9	0.0	1.8	0.4	1.4	0.106	**0.018**	14.6	0.7	24.8	8.3	24.4
HOMER	0.352	0.012	17.9	0.2	27.5	3.6	28.0	0.379	0.022	20.7	0.1	33.9	10.3	34.3
BR (RBF)	0.117	0.017	3.0	0.0	5.9	2.1	4.7	0.114	**0.018**	13.6	0.4	23.4	9.6	23.0
LR^{ML}	**0.101**	**0.009**	17.5	0.0	27.1	6.4	27.3	0.123	0.019	11.6	0.3	21.4	10.9	19.5
SVM^{ML}	0.205	**0.009**	9.9	0.8	18.5	5.0	17.5	0.184	0.019	6.9	0.2	11.1	6.6	12.2
SVM$^{ML}_{\gamma}$	0.174	**0.009**	18.8	1.0	29.4	5.9	26.3	0.163	0.019	14.9	0.3	27.1	12.1	23.6
LR^{ML} (RBF)	**0.101**	**0.009**	18.0	1.0	28.5	6.0	27.8	**0.096**	0.019	22.1	1.5	37.2	12.4	34.5
SVM^{ML} (RBF)	0.107	**0.009**	18.1	**1.8**	28.8	6.7	27.2	0.137	**0.018**	17.8	**1.7**	32.4	16.7	26.4
SVM$^{ML}_{\gamma}$ (RBF)	0.105	**0.009**	**19.3**	**1.8**	**30.2**	**6.8**	**28.8**	0.099	**0.018**	**23.1**	1.6	**39.0**	**18.2**	**35.7**

	emotions							**enron**						
Method	RLoss	HLoss	Acc	SAcc	F_1^{mic}	F_1^{mac}	F_1^{inst}	RLoss	HLoss	Acc	SAcc	F_1^{mic}	F_1^{mac}	F_1^{inst}
RF-PCT	**0.151**	**0.189**	51.9	**30.7**	67.2	65.0	61.1	0.079	0.046	41.6	13.1	53.7	12.2	55.2
HOMER	0.297	0.361	47.1	16.3	58.8	57.0	61.4	0.183	0.051	47.8	14.5	59.1	16.7	**61.3**
BR (RBF)	0.246	0.257	36.1	12.9	50.9	44.0	46.9	0.084	**0.045**	44.6	14.9	56.4	14.3	58.2
LR^{ML}	0.186	0.239	**53.6**	22.8	66.9	66.6	64.0	0.074	0.055	38.5	7.8	53.0	21.9	50.4
SVM^{ML}	0.217	0.238	50.4	23.3	63.4	65.2	63.9	0.136	0.055	38.9	10.5	50.3	21.6	50.9
SVM$^{ML}_{\gamma}$	0.178	0.230	54.0	23.3	**67.3**	**66.7**	**65.5**	0.095	0.050	42.8	10.5	56.2	23.2	54.9
LR^{ML} (RBF)	0.225	0.266	47.2	19.3	61.1	62.0	58.4	**0.070**	0.047	46.3	13.0	58.4	20.3	57.9
SVM^{ML} (RBF)	0.186	0.224	53.0	21.3	65.5	64.3	64.1	0.090	0.047	46.6	15.0	58.1	26.8	58.4
SVM$^{ML}_{\gamma}$ (RBF)	0.187	0.224	49.3	21.3	65.5	64.2	61.1	0.076	0.047	**48.6**	**16.1**	**59.5**	**26.9**	59.9

Table 6.10.: Multilabel classification results (part 1). The best 3 methods from the study by Madjarov et al., (2012) are compared to our multilabel methods. RF-PCT: random forest of predicting clustering trees (Kocev et al., 2007); HOMER: hierarchy of multilabel classifiers (Tsoumakas et al., 2008); BR: binary relevance method using SVM^{OVA}. HOMER and all the methods marked with (RBF) use an RBF kernel. The threshold δ for our methods is chosen by cross validation.

	mediamill							medical						
Method	RLoss	HLoss	Acc	SAcc	F_1^{mic}	F_1^{mac}	F_1^{inst}	RLoss	HLoss	Acc	SAcc	F_1^{mic}	F_1^{mac}	F_1^{inst}
RF-PCT	0.047	**0.029**	44.1	12.2	56.3	11.2	**58.9**	0.024	0.014	59.1	53.8	69.3	20.7	61.6
HOMER	0.177	0.038	41.3	5.3	55.3	7.3	57.9	0.090	**0.012**	71.3	61.0	77.3	28.2	76.1
BR (RBF)	0.061	0.032	40.3	8.0	53.3	5.6	55.7	**0.021**	0.077	20.6	0.0	34.3	36.1	32.8
LR^{ML}	**0.042**	0.033	41.2	7.8	54.8	17.1	54.4	0.024	0.013	68.7	56.9	76.2	35.1	75.2
SVM^{ML}	0.102	0.034	35.6	7.9	47.2	16.5	49.2	0.026	0.013	72.9	**62.8**	78.4	34.8	**77.5**
$\text{SVM}^{\text{ML}}_{\gamma}$	0.058	0.032	41.8	8.4	56.1	17.7	54.7	0.023	**0.012**	**73.1**	60.2	78.7	**36.7**	77.4
LR^{ML} (RBF)	**0.042**	0.033	42.0	10.0	56.2	21.8	53.3	0.031	0.016	64.9	46.5	72.6	28.0	71.4
SVM^{ML} (RBF)	0.072	0.031	43.3	11.8	57.6	25.9	55.3	0.027	**0.012**	72.5	61.7	**78.9**	36.6	76.6
$\text{SVM}^{\text{ML}}_{\gamma}$ (RBF)	0.046	**0.029**	**46.6**	**13.3**	**61.0**	**27.1**	58.7	0.027	**0.012**	72.5	61.6	**78.9**	36.6	77.2

	scene							yeast						
Method	RLoss	HLoss	Acc	SAcc	F_1^{mic}	F_1^{mac}	F_1^{inst}	RLoss	HLoss	Acc	SAcc	F_1^{mic}	F_1^{mac}	F_1^{inst}
RF-PCT	0.072	0.094	54.1	51.8	66.9	65.8	55.3	0.167	0.197	47.8	15.2	61.7	32.2	61.4
HOMER	0.119	0.082	**71.7**	**66.1**	**76.4**	**76.8**	74.5	0.205	0.207	55.9	21.3	67.3	44.7	**68.7**
BR (RBF)	**0.060**	**0.079**	68.9	63.9	76.1	76.5	71.4	0.164	0.190	52.0	19.0	65.2	39.2	65.0
LR^{ML}	0.081	0.120	58.3	39.4	67.2	68.4	66.3	0.352	0.264	36.0	8.4	48.3	44.4	48.0
SVM^{ML}	0.082	0.114	60.7	46.0	68.7	69.5	67.7	0.424	0.280	31.3	5.8	46.7	42.9	45.6
$\text{SVM}^{\text{ML}}_{\gamma}$	0.081	0.114	60.4	44.1	68.7	69.5	67.6	0.366	0.261	35.6	9.5	46.6	44.7	47.3
LR^{ML} (RBF)	0.068	0.096	63.6	54.3	72.0	73.0	70.5	0.160	0.193	55.1	19.0	67.5	47.1	66.9
SVM^{ML} (RBF)	0.069	0.088	69.1	58.0	75.1	75.9	**75.4**	0.159	0.188	**56.2**	**21.6**	68.2	48.1	66.7
$\text{SVM}^{\text{ML}}_{\gamma}$ (RBF)	0.064	0.088	68.0	58.0	74.7	75.2	75.0	**0.157**	**0.187**	**56.2**	19.8	**68.4**	**48.2**	67.0

Table 6.11.: Multilabel classification results (part 2); continuation of Table 6.10.

as can be seen in Table 6.9. Further details about each of the datasets can be found in (Madjarov et al., 2012).

We follow closely the evaluation protocol of Madjarov et al., (2012) except for the selection of the cut-off threshold δ (see § 6.2.1 for definition). Following Read et al., (2009), Madjarov et al. choose δ by matching label cardinality between the training and test data. While it is fast and easy to compute, that approach has two drawbacks: (i) being an instance of transductive learning, the method requires re-computation of δ every time test data changes; (ii) the choice of δ is not tuned to any performance measure and is likely to be suboptimal. In our experiments (not reported here), we observed generally comparable, but slightly lower results compared to when δ is tuned as discussed next.

Instead, Koyejo et al., (2015) recently showed that a consistent classifier is obtained when one computes δ by optimizing a given performance measure on a hold-out validation set. While there are at most mn distinct values of δ that would need to be considered, we limit the search to the grid $\{-10^{(-5.9:.2:1)}, 0, 10^{(-5.9:.2:1)}\}$ of 71 values. Following Madjarov et al., (2012), we use 10-fold cross-validation to select $C = 1/(\lambda n)$, the RBF kernel parameter $\theta = 1/(2\sigma^2)$, and the threshold δ, as described above. We use rather large and fine-grained grids both for C (from 2^{-20} to 2^5) and θ (from 2^{-15} to 2^3). The smoothing parameter is always set $\gamma = 1$.

Tables 6.10 and 6.11 present our experimental results. We report 7 performance metrics previously introduced in § 6.2.1 and tune the hyper-parameters for each metric individually. All metrics, except the rank loss and the hamming loss, are given in percents. Since the RF-PCT method did not use the RBF kernel in (Madjarov et al., 2012), we also report results with the linear kernel for our methods in the middle section of each table.

Overall, experimental results indicate competitive performance of our methods across all datasets and evaluation measures. Specifically, we highlight that the smooth $\text{SVM}_{\gamma}^{\text{ML}}$ with the RBF kernel yields the best performance in 38 out of 70 cases. On the two largest datasets, **bookmarks** and **delicious**, where the previous methods even struggled to complete training, we are able to achieve significant performance improvements both in rank loss as well as in partition-based measures. Finally, we note that while the previous methods show rather large variability in performance, all three of our multilabel methods tend to be more stable and show results that are concentrated around the best performing method in each case.

Multiclass to multilabel

Collecting ground truth annotation is hard. Even when the annotation is simply an image level tag, providing a *consistent* and *exhaustive* list of labels for every image in the training set would require significant effort. It is much easier to provide a weaker form of annotation where only a single prominent object is tagged. An interesting question is then whether it is still possible to train multilabel classifiers from multiclass annotation. And if so, how large is the performance gap compared to methods trained with full multilabel annotation? In the following, we set to explore that setting and answer the questions above.

We also note that top-k classification emerges naturally as an intermediate step between multiclass and multilabel learning. Recall that top-k loss functions operate in the multiclass setting where there is a single label per example, but that label is hard to guess correctly on the first attempt. One could imagine that the example is actually associated with k labels, but only a single label is revealed in the annotation. Therefore, it is also interesting to see if our top-k loss functions can offer an advantage over the classic multiclass losses in this setting.

To evaluate the multiclass, top-k, and multilabel loss functions on a common task, we choose two multilabel image classification benchmarks: Pascal VOC 2007 and Microsoft COCO. Multilabel methods are trained using full image level annotation (i.e. all class labels, but no bounding boxes or segmentation), while multiclass and top-k methods are trained using a *single* label per image. Both datasets offer object level bounding box annotations which can be used to estimate relative sizes of objects in the scene. For multiclass training, we only keep the label of the largest object, which is our proxy to estimating the prominent object in the image. All methods are evaluated using full annotation at test time. Note that except for pruning the training labels, we do not use bounding boxes anywhere during training or testing.

Experimental setup. We use 5K images for training and 5K for testing on Pascal VOC 2007, and 83K for training and 40K for testing on the MS COCO validation set. We split the training data in half for parameter tuning, and re-train on the full set for testing. We tune the regularization parameter $C = 1/(\lambda n)$ in the range from 2^{-20} to 2^{15}, and the top-k parameter k in the range $\{2, 3, 4, 5\}$. For the partition-based measures, we also tune the threshold δ in the range $[0.1, 10]$ with 100 equally spaced points. That range was chosen by observing the distribution of δ as computed by matching the label cardinality between training and test data. All parameters are tuned for each method and performance metric individually.

To isolate the effect of loss functions on classifier training from feature learning, we follow the classic approach of extracting features as a pre-processing step and then train our classifiers on the fixed image representation. We use our own implementation of SDCA based solvers for all of the methods considered in this section. That offers strong convergence guarantees due to (i) convexity of the objective and (ii) having the duality gap as the stopping criterion.

Our feature extraction pipeline is fairly common and follows the steps outlined by Simonyan and Zisserman, (2015) and Wei and Hoai, (2016). We compute multiple feature vectors per image. Every original image is resized isotropically so that the smallest side is equal to $Q \in \{256, 384, 512\}$ pixels, and then horizontal flips are added for a total of 6 images at 3 scales. We use MatConvNet (Vedaldi and Lenc, 2015) and apply the ResNet-152 model (He et al., 2016) which has been pre-trained on ImageNet. We extract features from the pool5 layer and obtain about 500 feature vectors of dimension 2048 per image on Pascal VOC (the exact number depends on the size of the original image). To reduce computational costs on COCO, we increase the stride of that layer to 2 for $Q \in \{384, 512\}$, which yields about 140 feature vectors per image and a total of $n = 12$M training examples. Unlike Wei and Hoai, (2016), we do not compute an additional global descriptor and also perform no normalization. Our preliminary experiments showed no advantage in doing so, and we keep the pipeline close to the original ResNet network.

Every feature vector can be mapped to a region in the original image. For training, we simply replicate the same image labels effectively increasing the size of the training set. At test time, we obtain a single ranking of class labels per image by max pooling the scores for each class. We follow this basic setup, but note that a $1-2\%$ improvement is possible with a more sophisticated aggregation of information from the different image regions, e.g., as done by Wei and Hoai, (2016) and Zhao et al., (2016).

Pascal VOC 2007. Here, we discuss the results presented in Tables 6.12 and 6.14. We start with the first table which reports the standard VOC evaluation measure, the mean AP, on the Pascal dataset. First, we compare top-1 (multiclass) and top-k classification methods. As before, although the differences are small, we see consistent improvements in each of the three groups: LR^{Multi} to top-k Ent, $\text{SVM}^{\text{Multi}}$ to top-k SVM^{β}, and $\text{SVM}^{\text{Multi}}_{\gamma}$ to top-k $\text{SVM}^{\beta}_{\gamma}$. The best top-1 method is $\text{SVM}^{\text{Multi}}$ with 89.3% mAP, which is outperformed by top-k SVM^{β} reporting the overall best multiclass result of 89.5% mAP.

Labels	Method	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	prsn	plant	sheep	sofa	train	tv	mAP
multi-class	LR$^{\text{Multi}}$	99.2	95.0	92.5	92.3	61.9	86.6	93.4	95.8	55.3	85.8	82.0	92.1	97.2	91.5	93.2	70.8	82.1	82.6	97.8	81.1	86.4
	top-k Ent	99.1	**96.0**	92.3	95.4	62.1	89.2	93.9	95.3	58.5	88.1	72.8	94.2	97.3	93.8	93.0	67.9	87.7	83.4	97.6	**85.3**	87.1
	SVM$^{\text{Multi}}$	**99.5**	94.0	**97.0**	**96.8**	62.1	**93.4**	**94.6**	**97.5**	65.0	89.9	**85.1**	**97.4**	**97.8**	**95.5**	93.7	**71.0**	90.2	84.4	**98.7**	82.3	89.3
	top-k SVM$^{\alpha}$	99.3	95.5	94.7	95.5	61.5	91.9	**94.6**	97.4	**66.7**	89.0	80.8	97.1	97.7	95.4	**95.3**	70.7	90.2	84.3	98.5	84.8	89.0
	top-k SVM$^{\beta}$	99.4	95.5	96.0	95.9	63.5	92.6	**94.6**	97.4	66.1	**90.2**	84.1	97.1	**97.8**	**95.5**	95.0	70.9	**91.7**	**84.6**	98.5	83.9	**89.5**
	SVM$_{\gamma}^{\text{Multi}}$	99.4	95.4	95.0	95.5	64.3	91.9	94.4	97.0	64.0	90.0	84.7	96.1	97.7	94.8	94.2	70.6	89.7	**84.6**	98.3	83.3	89.0
	top-k SVM$_{\gamma}^{\alpha}$	99.3	**96.0**	93.2	95.0	63.6	90.7	94.3	97.0	62.4	89.3	79.7	96.0	97.6	95.0	95.0	70.2	89.5	84.4	98.3	83.9	88.5
	top-k SVM$_{\gamma}^{\beta}$	99.3	95.6	94.7	95.2	**64.4**	91.8	94.5	97.1	65.1	89.8	84.2	96.3	97.7	94.9	94.8	70.6	89.7	**84.6**	98.4	84.0	89.1
multi-label	LR$^{\text{ML}}$	98.8	94.2	92.3	90.6	56.6	83.3	92.1	95.8	65.0	85.3	84.0	93.9	96.5	93.6	92.5	69.4	83.8	81.2	97.7	78.2	86.2
	SVM$^{\text{ML}}$	99.5	**96.5**	**97.5**	**96.7**	**71.8**	**93.6**	**95.3**	**97.8**	**79.3**	**92.0**	87.6	**98.4**	98.2	**96.6**	97.9	**73.1**	**93.3**	**83.8**	**98.7**	**88.5**	**91.8**
	SVM$_{\gamma}^{\text{ML}}$	**99.6**	96.3	97.1	96.4	69.5	93.3	94.9	97.5	76.7	91.3	**88.0**	98.0	**98.3**	**96.6**	**98.1**	72.6	93.2	83.7	98.6	88.0	91.4

Table 6.12.: Pascal VOC 2007 classification results. Evaluation of multiclass, top-k, and multilabel classification methods. Methods in the "multiclass" section above use only a *single* label per image, while methods in the "multilabel" section use all annotated labels. Please see the section **Multiclass to multilabel** for further details on the learning setting.

Labels	Method	mAP	P@1	P@2	P@3	P@5	R@1	R@3	R@5	R@10	RLoss	HLoss	Acc	SAcc	F_1^{mic}	F_1^{mac}	F_1^{inst}
multi-class	$\mathrm{LR}^{\mathrm{Multi}}$	54.6	92.6	66.9	52.2	37.0	44.8	65.8	73.9	82.9	0.066	0.028	43.4	15.9	52.4	41.1	55.8
	$\mathrm{SVM}^{\mathrm{Multi}}$	54.2	92.8	66.9	51.9	36.6	44.9	65.6	73.4	83.5	0.057	**0.025**	48.3	20.7	55.6	43.3	60.1
	top-k SVM^{α}	58.3	92.8	68.1	53.2	**37.5**	44.9	66.8	**74.8**	83.7	0.054	**0.025**	48.8	20.8	56.8	44.4	59.9
	top-k SVM^{β}	59.0	**93.2**	**68.4**	53.2	**37.5**	**45.0**	**66.9**	74.7	**84.0**	**0.053**	**0.025**	**49.7**	**21.2**	**57.5**	44.5	**61.3**
	$\mathrm{SVM}_{\gamma}^{\mathrm{Multi}}$	58.1	93.0	67.7	52.8	37.2	**45.0**	66.5	74.3	83.6	0.056	**0.025**	48.9	20.2	56.5	**44.6**	60.6
	top-k $\mathrm{SVM}_{\gamma}^{\alpha}$	58.4	92.8	68.1	53.2	**37.5**	44.9	66.8	**74.8**	83.7	0.055	**0.025**	48.4	20.4	57.0	44.5	60.3
	top-k $\mathrm{SVM}_{\gamma}^{\beta}$	**59.1**	**93.2**	**68.4**	**53.3**	37.4	**45.0**	**66.9**	74.7	83.7	0.054	**0.025**	49.3	20.9	57.2	44.4	60.9
multi-label	$\mathrm{LR}^{\mathrm{ML}}$	58.2	92.8	76.3	61.8	44.2	44.9	75.5	84.6	**93.2**	0.021	0.030	43.7	16.9	52.9	49.6	55.6
	$\mathrm{SVM}^{\mathrm{ML}}$	63.0	92.1	72.9	57.4	40.6	44.1	70.8	78.9	89.1	0.040	0.024	49.5	25.6	58.3	50.6	60.5
	$\mathrm{SVM}_{\gamma}^{\mathrm{ML}}$	**71.0**	**95.7**	**79.5**	**63.4**	**44.6**	**46.2**	**77.1**	**85.3**	**93.2**	**0.020**	**0.021**	**57.4**	**29.8**	**65.5**	**58.9**	**67.9**

Table 6.13.: MS COCO multilabel classification results. Methods in the "multiclass" section use only a *single* label per image, while methods in the "multilabel" section use all annotated labels. Please see the section **Multiclass to multilabel** for further details on the learning setting, and § 6.2.1 for details on the evaluation measures.

Method	R@1	R@3	R@5	RLoss	HLoss	Acc	SAcc	F_1^{mic}	F_1^{mac}	F_1^{inst}
LR^{Multi}	76.2	94.3	98.0	0.016	0.029	73.2	54.4	78.1	75.0	80.0
top-k Ent	76.1	94.3	97.8	0.016	0.038	48.1	44.0	59.8	72.1	61.7
$\text{SVM}^{\text{Multi}}$	76.5	94.1	97.6	0.017	0.025	76.0	60.6	80.3	77.5	81.9
top-k SVM^{α}	76.4	94.6	98.0	0.015	0.025	76.6	61.3	81.4	77.9	81.7
top-k SVM^{β}	**76.8**	**95.2**	98.1	**0.014**	**0.024**	**77.3**	**62.0**	82.0	**78.6**	**83.1**
$\text{SVM}_{\gamma}^{\text{Multi}}$	76.6	94.7	98.0	0.015	0.025	75.9	60.0	81.2	78.3	82.3
top-k $\text{SVM}_{\gamma}^{\alpha}$	76.4	95.0	**98.2**	0.015	0.025	76.2	59.3	81.2	76.7	81.5
top-k $\text{SVM}_{\gamma}^{\beta}$	76.7	**95.2**	98.1	**0.014**	**0.024**	76.7	60.5	**82.2**	78.1	82.8
LR^{ML}	76.5	96.3	98.9	0.010	0.027	75.2	59.8	81.3	77.4	80.7
SVM^{ML}	**78.0**	96.8	98.9	**0.008**	0.019	81.6	69.8	85.8	81.9	86.1
$\text{SVM}_{\gamma}^{\text{ML}}$	77.9	**97.3**	**99.1**	**0.008**	**0.018**	**82.4**	**70.8**	**86.8**	**83.0**	**86.4**

Table 6.14.: Pascal VOC 2007 multilabel classification results.

Next, we look at the performance gap between multiclass and multilabel settings. The best mAP of 91.8% is achieved by the multilabel SVM, SVM^{ML}, which exploits full annotation to boost its performance. However, the gap of just above 2% suggests a non-trivial trade-off between the additional annotation effort and the resulting classification performance. One limitation of the results on VOC 2007 is the relatively low label cardinality of only 1.5 labels per image. The picture changes on MS COCO where the label cardinality is about 3 labels per image.

Comparing the smooth and nonsmooth losses, we see that nonsmooth loss functions tend to perform better on this dataset. Moreover, SVM seems to perform significantly better than softmax. While this is a somewhat surprising result, it has been observed previously, e.g. with the R-CNN detector (Girshick, 2015; Lenc and Vedaldi, 2015), and with deeply-supervised CNNs (Lee et al., 2015a), even though their comparison was to OVA SVM.

Finally, we note that the current state of the art classification results on VOC 2007 are reported in (Wang et al., 2016b; Wei and Hoai, 2016; Zhao et al., 2016). Our 91.8% mAP of SVM^{ML} matches exactly the result of LSSVM-Max by Wei and Hoai, (2016), which operates in the setting closest to ours in terms of image representation and the learning architecture. Their proposed PRSVM method performs additional inference (as opposed to the simple max pooling that we use) and achieves 92.9% mAP. Multiscale orderless pooling by Zhao et al., (2016) is directly comparable to our setting and yields 90.8% mAP. Performing inference on the extracted image regions, they too report around 93% mAP, while additionally exploiting bounding box annotations boosts the performance to 93.7%.

While mAP is the established performance measure on Pascal VOC datasets, it does not evaluate how well a method captures inter-class correlations since the AP is computed for each class independently. To address this limitation, we also report a number of multilabel performance metrics from § 6.2.1 in Table 6.14. The best performing method in the multiclass category is again top-k SVM^{β}, but the improvement over the baseline $\text{SVM}^{\text{Multi}}$ is more pronounced. Furthermore, the smooth $\text{SVM}^{\text{ML}}_{\gamma}$ now clearly outperforms its nonsmooth counterpart also significantly increasing the gap between multiclass and multilabel methods.

MS COCO. Table 6.13 presents our results on the MS COCO benchmark. The general trend is similar to that observed on VOC 2007: top-k methods tend to outperform top-1 multiclass baselines, but are outperformed by multilabel methods that exploit full annotation. However, the differences between the methods are more meaningful on this dataset. In particular, smooth top-k $\text{SVM}^{\beta}_{\gamma}$ achieves 59.1% mAP, which is a 1% improvement over $\text{SVM}^{\text{Multi}}_{\gamma}$, while multilabel $\text{SVM}^{\text{ML}}_{\gamma}$ boosts the performance to 71%. The improvement of over 10% highlights the value of multilabel annotation, even though this result is subject to the bias of our label selection procedure for multiclass methods: small objects may have not been repesented well. That class imbalance could be also the reason for relatively poor mAP performance of $\text{SVM}^{\text{Multi}}$ and LR^{Multi} methods in these experiments.

The current state of the art classification results on COCO are reported by Zhao et al., (2016). A comparable architecture achieved 69.7% mAP, while performing inference on the multiple regions per image and exploiting the bounding box annotations boosted the performance to 73% mAP.

Looking at multilabel evaluation measures, we can also make a few interesting observations. First, the rank loss seems to correlate well with the other performance measures, which is good since that is the metric that our loss functions are designed to optimize. Second, strong performance at P@1 suggests that a single guess is generally sufficient to guess *a* correct label. However, due to high class imbalance this result is not too impressive and is humbled by the performance of R@k: even 10 attempts may not suffice to guess all relevant labels. The difficulty of properly ranking the less represented classes is also highlighted by the relatively low accuracy and subset accuracy results, although the latter metric may be too stringent for a large scale benchmark.

6.6 Conclusion

Modern large scale image classification benchmarks are subject to the problem of increased *class ambiguity* to the extent that the classical error rate becomes too stringent and may not be an adequate performance measure. In Chapter 5, we proposed to consider the top-k error as the target performance measure instead, and formulated the corresponding *top-k SVM* algorithm.

In this chapter, we continued the study of class ambiguity and carried out an extensive experimental study of multiclass, top-k, and multilabel performance optimization. We observed that the softmax loss and the smooth hinge loss are competitive across all top-k errors and should be considered the primary candidates in practice. Our new surrogate losses include the *smooth top-k hinge* loss and the *top-k entropy* loss. These novel methods can further improve the top-k results, especially if one is targeting a particular top-k error as the performance measure, or if the training examples are multilabel in nature.

We have also considered an interesting transition from multiclass to top-k to multilabel classification, and observed that effective multilabel classifiers can be trained from single label annotation. Our results indicate, in particular, that the classical multilabel SVM is competitive in mAP on Pascal VOC 2007, however, the proposed *smooth multilabel SVM* outperforms the competing methods in other metrics on Pascal VOC, and in all metrics on MS COCO.

Finally, we would like to highlight our optimization schemes for top-k Ent, top-k SVM_γ, and $\mathrm{SVM}_\gamma^{\mathrm{ML}}$, which include the softmax loss and multiclass, multilabel SVM as special cases. Our optimization algorithms are based on the SDCA framework of Shalev-Shwartz and Zhang, (2013b) and use efficient projection subroutines which are of independent interest.

7 Conclusions, Insights, and Perspectives

In this chapter, we present a brief summary of the thesis highlighting some of the conclusions and insights, and offer an outlook for further research directions.

We started the dissertation with an intriguing problem: how do we teach a machine to see? We discussed that the statistical learning theory offers a sound platform to build algorithms that learn from data. From the computer vision perspective, we understood the specific representations of data and attempted to capture invariances to certain irrelevant transformations. Furthermore, we formalized a concrete perceptual task of being able to classify and categorize images. Having reviewed the related work, we identified and contributed towards overcoming two major challenges in the automated learning approach: limited amount and ambiguity in the training data.

7.1 Conclusions and Insights

In this section, we draw conclusions from our results and offer some insights which could be useful to a practitioner or, perhaps, even inspire further research. The section is organized into two parts following the structure of the thesis.

Learning with Limited Training Data

In Part I, which covers learning with limited training data, we considered two frameworks that incorporate additional prior knowledge: learning with privileged information (Chapter 3) and multitask learning (Chapter 4). Below, we summarize some of the main insights and conclusions related to that line of research.

Privileged Information is related to instance weighting, or importance weighting, and, therefore, can be interpreted as a way to introduce guidance on the difficulty of the individual training examples.

SVM+ realizes an interesting constraint that forces it to concentrate more weight on the difficult, even misclassified, examples. This is an artifact of the formulation that offers an *upper* bound on the loss on the given example, which is based on the assumption that the loss in the decision space should not exceed the loss in the correcting space.

Weight Learning is demonstrated to be both beneficial and hard in practice. We obtained substantial performance improvements when the instance weights were learned on a large validation sample, which proves the existence of weighting schemes that perform significantly better than uniform weighting. That could motivate further research to find such schemes, in particular, exploiting the privileged information. On the other hand, we observed that adjusting the loss function, or, even more generally, learning a loss function, can lead to severe overfitting and has to be carefully controlled.

Multitask Learning is an effective method to exploit task relatedness and it generally outperforms single task learning where every task is learned independently. In our experiments, we observed consistent, although limited, improvements over the OVA baseline. Moreover, multitask learning is scalable to modern datasets and feature vectors if an efficient optimization scheme is used.

Learning with Class Ambiguity

In Part II, we introduced the problem of class ambiguity and recognized the top-k error as an appropriate target performance measure. We proposed top-k multiclass SVM (Chapter 5) as a suitable learning algorithm and developed an efficient optimization scheme based on SDCA. Furthermore, we proposed smooth top-k SVM, top-k versions of the softmax loss, and smooth multilabel SVM along with the corresponding optimization algorithms (Chapter 6). Our theoretical analysis and extensive empirical evaluation leads to the following insights and conclusions.

Top-k Error is an interesting performance measure that offers nontrivial tradeoffs depending on the target value of k. In particular, our proposed top-k SVM and top-k entropy methods demonstrate consistent improvements in top-k error for $k > 1$ at the cost of a higher top-1 error, which reveals a "diagonal" pattern in the results table. However, there is no direct correspondence between the k in the loss and the k' in the performance metric, which suggests that the value of k should be cross-validated for the target performance measure.

Softmax Loss and smooth multiclass SVM demonstrate surprisingly strong performance on multiclass datasets uniformly across all values of k. Therefore, we recommend these two methods as the primary candidates in practice if there is only limited ambiguity in the labels and no budget to tune the k in top-k SVM. However, our experiments on multilabel datasets, in particular on MS COCO, show that the proposed top-k methods outperform their multiclass counterparts in the presence of significant class ambiguity.

Top-k Calibration is our extension of the concept of classification calibration for the top-k error as the target performance measure. Notably, we have shown that the softmax loss is not only classification calibrated for the standard (top-1) error, but is uniformly top-k calibrated for all $k \geq 1$. This is a strong property, although of an asymptotic nature, which may offer an explanation for the strong top-k performance of the softmax loss.

OVA vs Multiclass is an old dispute among researchers and we do not aim to settle it here. In our experience, OVA scheme offers hardly any advantage in terms of training time over an efficiently implemented multiclass method, unless the training is massively parallelized. On the other hand, we observed that multiclass methods tend to perform on par or better compared to the OVA scheme, particularly in top-k error for $k > 1$.

Efficient Projection algorithms enable scalable optimization with vector valued loss functions, such as the ones used for multiclass, top-k, and multilabel learning. SDCA updates based on projections are optimal in the sense that they yield maximal increases of the dual objective over small batches of dual variables given other variables fixed. Moreover, the same projection algorithms can be used to compute the gradients of the corresponding loss functions, and may be integrated into a deep learning architecture. The downside of the SDCA framework, however, is that one has to maintain the dual variables which requires memory that scales linearly both with the number of classes and training examples.

Novel Losses can be defined via the conjugate of an existing loss function, as we have demonstrated with the introduction of the top-k entropy loss. In particular, the effective domain of the conjugate can be modified to introduce the desired properties of the loss, while the primal loss function is then obtained by computing the conjugate of the modified conjugate loss. The procedure is reminiscent of the Moreau-Yosida regularization discussed next.

Smoothed Loss can be obtained from a nonsmooth loss using Moreau-Yosida regularization which adds an ℓ_2 regularizer to the conjugate loss. We used that technique to obtain smooth multiclass, top-k, and multilabel SVMs, which demonstrated significantly faster training times, and often improved the generalization performance.

7.2 Future Perspectives

This final section gives a brief outlook on further research directions.

Learning Using Privileged Information

We start with the framework of learning using privileged information (LUPI) and list some ideas that are motivated by our results.

SVM– We have already considered the question whether it is possible to formulate an SVM– algorithm in § 3.4.5. There, we considered a rather intuitive, yet naive, modification which was quickly dismissed as it leads to a nonconvex problem. Instead, we can develop the insight discussed above and realize that the asymmetric constraint between the losses in the decision and the correcting spaces can be reversed. That is, one could consider an alternative formulation where the loss in the correcting space gives a *lower* bound, similar to the loss inequality regularization method of Wang and Ji, (2015).

developed a fast optimization scheme for SVM+ based on dual coordinate ascend, and You et al., (2017)

Multiclass, Multilabel SVM+ Research so far has mainly focused on the binary SVM+ algorithm, and used the classical OVA scheme for multiclass problems (Sharmanska et al., 2014). There are only limited attempts to develop multiclass algorithms that would utilize the privileged information without resorting to binary classifiers: the M^{2}PiSVM of Ji et al., (2012), the ν-K-SVCR+ model of Liu et al., (2013) and the SVM+MTL method of Ren et al., (2015). A multilabel formulation was proposed by You et al., (2017). It would be interesting to further investigate the use of privileged information in multiclass and multilabel algorithms, such as the ones developed by Crammer and Singer, 2001, 2003; Elisseeff and Weston, 2001.

LR+, Softmax+ The adoption of the logistic loss in the LUPI paradigm is somewhat limited (Wang and Ji, 2016; Wang et al., 2014b), while the softmax loss, to the best of our knowledge, has not been considered yet. In particular, Wang et al., (2014b) proposed an LR+ method, where the agreement between predictions in the decision and the correcting spaces is enforced with the ℓ_2 distance. While computationally tractable, the method is not quite in line with the original Vapnik's idea, since the dependency between the spaces is now symmetric. Instead, it might be interesting to consider enforcing an asymmetric constraint between the losses, and it might be more elegant if done in the dual space.

Efficient Optimization Finally, there is only limited work on developing efficient optimization algorithms for SVM+ and the related methods (Li et al., 2016; Pechyony and Vapnik, 2011; You et al., 2017), which limits their applicability in modern large scale datasets. One could extend the work of Li et al., (2016) adapting the SDCA framework to the proposed SVM+ variations, as well as consider the Moreau-Yosida smoothing technique.

Multitask Representation Learning

We have already listed a number of possible MTL-SDCA extensions in § 4.2.2 (page 67), which mainly consider the adoption of different loss functions and regularizers. In hindsight, our formulation can be interpreted from the perspective of deep learning, where there is already a substantial amount of ongoing work. Although MTL-SDCA in its current form is closer related to supervised dictionary learning and matrix factorization methods, the introduction of a nonlinear transform on top of the lower dimensional representation $U^\top x$ turns our method into a classical two layer neural network.

One idea, which is inspired by the iterative SDCA optimization scheme, might be worth exploring. Recall that we use a form of block coordinate descent to find a local minimum of a biconvex objective. In doing so, we perform decoupled iterative updates of the two matrices, U and W, and maintain monotonic descent of the global objective. The latter may be desirable for the diagnostic purposes. In contrast, stochastic backpropagation performs joint updates of all the layers at every iteration and does not guarantee monotonicity. Moreover, it requires full forward and backward passes, which is hard to parallelize and can be computationally expensive in very deep networks. An interesting ongoing research direction is based on the idea that the optimization of individual layers can be decoupled

(Carreira-Perpiñán and Wang, 2014; Jaderberg et al., 2016; Lee et al., 2015b). We believe this is a promising avenue for further research.

Top-k Optimization and Label Ranking

Finally, we provide a brief outlook from the perspective of our most recent work on top-k error optimization.

Top-k Calibration The calibration of smooth and nonsmooth top-k hinge loss is, unfortunately, an open question, as Table 6.1 in Chapter 6 shows. We know that top-1 multiclass SVM is not calibrated, while smooth OVA SVM is. It would be nice to provide a definitive answer regarding top-k calibration of smooth and nonsmooth top-k SVMs for $k > 1$.

Generalization Bounds Theoretical analysis of the proposed top-k SVM would be incomplete without a measure of its generalization performance. Lei et al., (2015) have recently obtained a data-dependent generalization error bound with a logarithmic dependency on the number of classes. In particular, their analysis applies to the multiclass SVM of Crammer and Singer, (2001), and it should be possible to extend it to the top-k SVM as well.

Multilabel Softmax One limitation of the multilabel softmax loss (LR^{ML}) considered in Chapter 6 is that the summation inside the logarithm goes over all the classes, including both positive and negative labels. While that formulation followed from a natural model of the conditional probability, an alternative loss function might be interesting to consider:

$$L(u) = \tfrac{1}{|Y|} \textstyle\sum_{y \in Y} \log \Big(\textstyle\sum_{j \in \bar{Y}} \exp(u_j - u_y) \Big),$$

where the inner sum goes over the negative labels only.

Ranking Losses When we introduced the top-k hinge loss (α), we argued that it offers a tighter convex upper bound on the discrete top-k error compared to the top-k hinge loss (β). Following that line of thought, one could study if our construction can be generalized to the family of ranking losses considered by Usunier et al., (2009), which would lead to tighter convex upper bounds on the corresponding discrete losses.

Smooth Ranking Shalev-Shwartz and Singer, (2006) considered an interesting generalization of label ranking where the ground truth is given in the form of a graph of preferences, with nodes being the labels and edges expressing the ranking of labels. In their approach, the graph is decomposed into bipartite subgraphs which enforce pairwise label ranking and are used to define the overall loss. Being quite general, their framework includes as special cases the multiclass SVMs of Weston, Watkins, et al., (1999) and Crammer and Singer, (2001), as well as the multilabel SVM of Elisseeff and Weston, (2001). However, their framework could be extended in a number of ways. First, their loss functions are nonsmooth, and it should be possible to obtain a smoothed version using Moreau-Yosida regularization. Second, it might be possible to extend their framework to the softmax loss, and develop the corresponding optimization scheme. Finally, as Shalev-Shwartz and Singer, (2006) note themselves, one could consider k-partite decompositions, which would model higher order label dependencies.

A Convex Analysis

Here, we recall some of the well-known results from convex analysis that are used throughout the dissertation.

A.1 Lagrangian Duality

Duality equips us with a rigorous framework to handle constrained optimization problems that often arise in machine learning. Here, we cover the Lagrangian duality that is traditionally popular in research on support vector machines.

The material presented in this section is based on the *Convex Optimization* book by Boyd and Vandenberghe, (2004).

A.1.1 The Lagrange Dual Problem

Consider an optimization problem below, which we will call the *primal problem*,

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) \leq 0, \quad i = 1, \ldots, m, \\ & h_i(x) = 0, \quad i = 1, \ldots, p, \end{array} \tag{A.1}$$

where $x \in \mathbb{R}^n$ and the domain $\mathcal{D} = \bigcap_{i=0}^{m} \mathbf{dom}\, f_i \cap \bigcap_{i=1}^{p} \mathbf{dom}\, h_i$ is nonempty.

The *Lagrangian* $\mathcal{L} : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^p \to \mathbb{R}$ associated with the problem (A.1) is defined as

$$\mathcal{L}(x, \lambda, \nu) = f_0(x) + \sum_{i=1}^{m} \lambda_i f_i(x) + \sum_{i=1}^{p} \nu_i h_i(x),$$

with $\mathbf{dom}\, \mathcal{L} = \mathcal{D} \times \mathbb{R}^m \times \mathbb{R}^p$. The vectors λ and ν are called the *dual variables*, while the vector x is called the *primal variable*. Throughout the thesis, we often omit explicit enumeration of the domain and all the variables of the Lagrangian.

The *Lagrange dual function* $g : \mathbb{R}^m \times \mathbb{R}^p \to \mathbb{R}$ is defined as the minimum value of the Lagrangian over x:

$$g(\lambda, \nu) = \inf_{x \in \mathcal{D}} \mathcal{L}(x, \lambda, \nu).$$

The *Lagrange dual problem* associated with the problem (A.1) is defined as

$$\begin{array}{ll} \text{maximize} & g(\lambda, \nu) \\ \text{subject to} & \lambda_i \geq 0, \quad i = 1, \ldots, m. \end{array} \tag{A.2}$$

The Lagrange dual problem (A.2) is a convex optimization problem even when the primal problem is not convex.

Let $p^\star$ and $d^\star$ be the optimal values of the primal and the dual problems respectively. The following inequality, known as *weak duality*, always holds:

$$d^\star \leq p^\star.$$

Often, in particular for convex optimization problems, *strong duality* holds:

$$d^\star = p^\star,$$

and we say that the optimal duality gap is zero. Strong duality can be written as

$$\sup_{\lambda \geq 0,\, \nu} \inf_{x} \mathcal{L}(x, \lambda, \nu) = \inf_{x} \sup_{\lambda \geq 0,\, \nu} \mathcal{L}(x, \lambda, \nu),$$

which is known as the *strong max-min property* or the *saddle-point property.*

Strong duality holds for all convex optimization problems in this thesis where we consider dual optimization.

A.1.2 KKT Optimality Conditions

The *Karush-Kuhn-Tucker* (KKT) conditions, which we introduce in this section, are known to be necessary and sufficient for the points $x^\star$ and $(\lambda^\star, \nu^\star)$ to be primal and dual optimal if the optimization problem is convex and strong duality holds. The KKT conditions associated with the problem (A.1) are defined as:

$$\begin{aligned} \nabla f_0(x^\star) + \textstyle\sum_{i=1}^m \lambda_i^\star \nabla f_i(x^\star) + \sum_{i=1}^p \nu_i^\star \nabla h_i(x^\star) &= 0, \\ f_i(x^\star) &\leq 0, \quad i = 1, \ldots, m, \\ h_i(x^\star) &= 0, \quad i = 1, \ldots, p, \\ \lambda_i^\star &\geq 0, \quad i = 1, \ldots, m, \\ \lambda_i^\star f_i(x^\star) &= 0, \quad i = 1, \ldots, m. \end{aligned} \tag{A.3}$$

The first condition states that the gradient of the Lagrangian vanishes at $x^\star$, which implies that $x^\star$ minimizes $\mathcal{L}(x, \lambda^\star, \nu^\star)$ over x. The next two constraints state that $x^\star$ is primal feasible, while $\lambda_i^\star \geq 0$ states that $\lambda^\star$ is dual feasible. Finally, the condition $\lambda_i^\star f_i(x^\star) = 0$ is known as *complementary slackness.* It means that the ith optimal dual variable is zero unless the ith constraint is active at the optimum.

The KKT conditions are important for a number of reasons: (i) optimization algorithms often solve (A.3) to find the optimal points; (ii) it may be possible to solve the KKT conditions analytically or obtain useful formulas, such as expressing the primal variables in terms of the dual variables; (iii) analysis of machine learning algorithms from the optimization perspective, e.g. our analysis of the SVM+ method in Chapter 3, largely relies on the optimality conditions.

A.1.3 Examples

We consider two examples that demonstrate application of Lagrangian duality to the following learning methods: (i) weighted support vector machine (WSVM) and (ii) support vector machine with privileged information (SVM+). In particular, we state the primal problem, the KKT conditions, and the associated dual problem.

Weighted Support Vector Machine

Binary weighted support vector machine (WSVM) with the linear kernel solves the following convex optimization problem:

$$\begin{array}{lll} \text{minimize} & (1/2)\,\|w\|^2 + \sum_{i=1}^n c_i \xi_i & \\ \text{subject to} & y_i(\langle w, x_i \rangle + b) \geq 1 - \xi_i, & i = 1, \dots, n, \\ & \xi_i \geq 0, & i = 1, \dots, n, \end{array} \tag{A.4}$$

with variables $w \in \mathbb{R}^d$, $b \in \mathbb{R}$, $\xi \in \mathbb{R}^n$, *feature vectors* $x_i \in \mathbb{R}^d$, ground truth *labels* $y_i \in \{\pm 1\}$, and nonnegative *instance weights* (or *costs*) $c \in \mathbb{R}^n_+$.

Let $(w^\star, b^\star, \xi^\star)$ and $(\alpha^\star, \beta^\star)$ be the optimal primal and dual points respectively. The KKT conditions associated with the problem (A.4) are given below.

$$\begin{aligned} \textstyle\sum_{i=1}^n \alpha_i^\star y_i x_i &= w^\star, \\ \textstyle\sum_{i=1}^n \alpha_i^\star y_i &= 0, \\ \alpha_i^\star + \beta_i^\star &= c_i, \\ \alpha_i^\star[\xi_i^\star - 1 + y_i(\langle w^\star, x_i \rangle + b^\star)] &= 0, \\ \beta_i^\star[\xi_i^\star] &= 0, \\ \xi_i^\star - 1 + y_i(\langle w^\star, x_i \rangle + b^\star) &\geq 0, \\ \alpha_i^\star \geq 0, \quad \beta_i^\star \geq 0, \quad \xi_i^\star &\geq 0. \end{aligned} \tag{A.5}$$

The Lagrange dual problem associated with the WSVM problem (A.4) is

$$\begin{array}{ll} \text{maximize} & \sum_{i=1}^n \alpha_i - (1/2) \sum_{i,j=1}^n y_i \alpha_i y_j \alpha_j \langle x_i, x_j \rangle \\ \text{subject to} & \sum_{i=1}^n y_i \alpha_i = 0, \\ & 0 \leq \alpha_i \leq c_i, \quad i = 1, \dots, n. \end{array}$$

The problem above can be rewritten equivalently as

$$\begin{array}{ll} \text{minimize} & (1/2)\,\alpha^\top Y K Y \alpha - \mathbf{1}^\top \alpha \\ \text{subject to} & y^\top \alpha = 0, \\ & 0 \leq \alpha_i \leq c_i, \quad i = 1, \dots, n, \end{array} \tag{A.6}$$

where we let $y = (y_1, \dots, y_n)^\top$, $Y = \mathbf{diag}(y)$, $K_{ij} = \langle x_i, x_j \rangle$ for $i, j = 1, \dots, n$.

Support Vector Machine with Privileged Information

Binary support vector machine with privileged information (SVM+) solves the following convex optimization problem:

$$\begin{array}{lll} \text{minimize} & (1/2)(\|w\|^2 + \gamma\,\|\tilde{w}\|^2) + C\sum_{i=1}^{n}(\langle \tilde{w}, \tilde{x}_i\rangle + \tilde{b}) & \\ \text{subject to} & y_i(\langle w, x_i\rangle + b) \geq 1 - (\langle \tilde{w}, \tilde{x}_i\rangle + \tilde{b}), & i = 1, \ldots, n, \\ & \langle \tilde{w}, \tilde{x}_i\rangle + \tilde{b} \geq 0, & i = 1, \ldots, n, \end{array} \tag{A.7}$$

with variables $w \in \mathbb{R}^d$, $b \in \mathbb{R}$, $\tilde{w} \in \mathbb{R}^p$, $\tilde{b} \in \mathbb{R}$, *feature vectors* $x_i \in \mathbb{R}^d$, *privileged features* $\tilde{x}_i \in \mathbb{R}^p$, ground truth *labels* $y_i \in \{\pm 1\}$, and nonnegative parameters $\gamma \in \mathbb{R}_+$ and $C \in \mathbb{R}_+$ that control the trade-offs in the objective.

The KKT conditions associated with the problem (A.7) are given below.

$$\begin{aligned} \textstyle\sum_{i=1}^{n} \alpha_i^\star y_i x_i &= w^\star, \\ \textstyle\sum_{i=1}^{n} \alpha_i^\star y_i &= 0, \\ \textstyle\sum_{i=1}^{n}(\alpha_i^\star + \beta_i^\star - C)\tilde{x}_i &= \gamma \tilde{w}^\star, \\ \textstyle\sum_{i=1}^{n}(\alpha_i^\star + \beta_i^\star - C) &= 0, \\ \alpha_i^\star[\langle \tilde{w}^\star, \tilde{x}_i\rangle + \tilde{b}^\star - 1 + y_i(\langle w^\star, x_i\rangle + b^\star)] &= 0, \\ \beta_i^\star[\langle \tilde{w}^\star, \tilde{x}_i\rangle + \tilde{b}^\star] &= 0, \\ \langle \tilde{w}^\star, \tilde{x}_i\rangle + \tilde{b}^\star - 1 + y_i(\langle w^\star, x_i\rangle + b^\star) &\geq 0, \\ \alpha_i^\star \geq 0, \;\; \beta_i^\star \geq 0, \;\; \langle \tilde{w}^\star, \tilde{x}_i\rangle + \tilde{b}^\star &\geq 0, \end{aligned} \tag{A.8}$$

where $(w^\star, b^\star, \tilde{w}^\star, \tilde{b}^\star)$ and $(\alpha^\star, \beta^\star)$ are the optimal primal and dual points.

The Lagrange dual problem associated with the SVM+ problem (A.7) is equivalent to the following simplified problem:

$$\begin{array}{ll} \text{minimize} & (1/2)\Big(\alpha^\top Y K Y \alpha + (1/\gamma)\,\tilde{\alpha}^\top \tilde{K} \tilde{\alpha}\Big) - \mathbf{1}^\top \alpha \\ \text{subject to} & y^\top \alpha = 0, \\ & \mathbf{1}^\top \tilde{\alpha} = 0, \\ & 0 \leq \alpha_i \leq C + \tilde{\alpha}_i, \;\; i = 1, \ldots, n, \end{array} \tag{A.9}$$

where we let $\tilde{\alpha}_i = \alpha_i + \beta_i - C$ for all $i = 1, \ldots, n$.

A.2 Fenchel Duality

We have seen in § A.1 that the KKT optimality conditions are invaluable in convex optimization as they characterize primal and dual optimal points. Furthermore, the KKT conditions can be used to derive the Lagrange dual problem, which may be computationally more attractive in certain applications.

In this section, we consider an alternative way to arrive at a dual problem. The Fenchel duality theory presented here is attractive for the two reasons: (i) we extend the theory to also include nonsmooth (non-differentiable) functions which may take the value $+\infty$; and (ii) we focus on the objectives that are given by a sum of two functions. The latter is particularly well suited to the learning problems in this thesis where the objectives are given by a sum of the loss and the regularization terms. That last property also allows one to work with the loss and the regularizer independently, which makes it easy to derive (Fenchel) dual problems for an arbitrary combination of different losses and regularizers.

The material in this section is based on the book by Borwein and Lewis, (2000). A close connection (essentially, equivalence) between the Lagrangian and Fenchel duality is discussed in (Hiriart-Urruty and Lemaréchal, 1993; Magnanti, 1974).

A.2.1 Subgradients and the Fenchel Conjugate

In this section, we recall some of the basic definitions and results from convex analysis that become relevant in the following.

The *(effective) domain* of a function $f : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ is the set

$$\mathbf{dom}\, f = \{x \in \mathbb{R}^n \,|\, f(x) < +\infty\}.$$

The function f is *proper* if its domain is nonempty. The *core* of a set $C \subset \mathbb{R}^n$ is

$$\mathbf{core}(C) = \{x \in C \,|\, \forall\, d \in \mathbb{R}^n \; \exists\, t > 0 \;:\; x + td \in C\}.$$

In particular, the core of C contains the interior of C, and if C is convex, then the core and the interior are identical (Borwein and Lewis, 2000, Theorem 4.1.4).

The idea of the derivative is that it allows to approximate a given function using a linear function. The same concept can be extended to nonsmooth functions. As we consider minimization problems, one-sided approximation is sufficient and leads to the following definition. A *subgradient* of f at x_0 is a vector $g \in \mathbb{R}^n$ such that

$$\langle g, x - x_0 \rangle \leq f(x) - f(x_0) \quad \forall\, x \in \mathbb{R}^n.$$

The *subdifferential* of f at x_0 is the set $\partial f(x_0)$ of all subgradients of f at x_0.

Proposition A.1 (Subgradient at optimality)**.** Let f be a proper function. The point x_0 is a (global) minimizer of f if and only if the condition $0 \in \partial f(x_0)$ holds.

Proposition A.2 (Differentiability of convex functions)**.** Let f be a convex function and $x_0 \in \mathbf{core}(\mathbf{dom}\, f)$. If f is differentiable at x_0, then $\partial f(x_0) = \{\nabla f(x_0)\}$.

The *Fenchel conjugate* (the *convex conjugate*) of a function $f : \mathbb{R}^n \to \mathbb{R} \cup \{\pm\infty\}$ is the function $f^* : \mathbb{R}^n \to \mathbb{R} \cup \{\pm\infty\}$ defined by

$$f^*(y) = \sup_{x \in \mathbb{R}^n} \{\langle y, x\rangle - f(x)\}.$$

Note that f^* is a convex function as it is the pointwise supremum of affine functions of y. The subgradient and conjugation are related via the following inequality.

Proposition A.3 (Fenchel-Young inequality). Let $f : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ be a proper function. Any points $x \in \mathbf{dom}\, f$ and $y \in \mathbb{R}^n$ satisfy the inequality

$$f(x) + f^*(y) \geq \langle y, x\rangle \,.$$

Equality holds if and only if $y \in \partial f(x)$.

A.2.2 Fenchel Duality

In this section, we state the main theorem that defines the Fenchel primal and dual problems as well as provides a useful optimality condition. Let $\mathbf{cont}\, f$ be the set of points where the function f is finite and continuous.

Theorem A.1 (Fenchel duality). For given functions $f : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ and $g : \mathbb{R}^m \to \mathbb{R} \cup \{+\infty\}$, and a linear map $A \in \mathbb{R}^{m \times n}$, let p and d be primal and dual values defined, respectively, by the **Fenchel problems**

$$\begin{aligned} p &= \inf_{x \in \mathbb{R}^n} \{f(x) + g(Ax)\}, \\ d &= \sup_{y \in \mathbb{R}^m} \{-f^*(A^\top y) - g^*(-y)\}. \end{aligned} \tag{A.10}$$

These values satisfy the *weak duality* inequality $d \leq p$. If f and g are convex and satisfy $A\, \mathbf{dom}\, f \cap \mathbf{cont}\, g \neq \emptyset$, then $d = p$ and

$$\partial(f + g \circ A)(x) = \partial f(x) + A^\top \partial g(Ax).$$

The points $x^\star \in \mathbb{R}^n$ and $y^\star \in \mathbb{R}^m$ are respectively primal and dual optimal for (A.10) if and only if they satisfy

$$A^\top y^\star \in \partial f(x^\star), \qquad\qquad -y^\star \in \partial g(Ax^\star). \tag{A.11}$$

Let us briefly discuss the implications of Theorem A.1. First, note that the functions f and g need not be smooth. That allows us to have a unified optimization framework that works equally well for smooth and nonsmooth loss functions and regularizers.

Second, note that both the primal and the dual objectives are given by the sum of two functions which can be interpreted as the loss and the regularization terms, see examples in § A.2.3 below. More importantly, the functions f, g are decoupled in the sense that their conjugates can be computed and plugged into the dual *independently*, making it easy to combine different losses and regularizers.

Finally, note that whenever we use Fenchel duality in the thesis, the functions f and g are convex, and the technical condition $A\,\mathbf{dom}\,f \cap \mathbf{cont}\,g \neq \emptyset$ is satisfied, therefore, the optimality condition (A.11) holds.

A.2.3 Basic Properties

The results in this section can be found in many textbooks on convex analysis, see e.g. in (Borwein and Lewis, 2000; Boyd and Vandenberghe, 2004).

Conjugate of the Conjugate

The *epigraph* of a function $f : \mathbb{R}^n \to \mathbb{R}$ is defined as

$$\mathbf{epi}\,f = \{(x,t) \,|\, x \in \mathbf{dom}\,f,\ f(x) \leq t\}.$$

A function $f : \mathbb{R}^n \to \mathbb{R}$ is *closed* if $\mathbf{epi}\,f$ is a closed set.

Proposition A.4. If f is a closed convex function, then $f^{**} = f$.

For example, if $\mathbf{dom}\,f = \mathbb{R}^n$ and f is convex, then $f^{**} = f$, i.e. the biconjugate of a closed convex function coincides with the function itself.

Scaling

For $a > 0$ and $b \in \mathbb{R}$, the conjugate of $g(x) = af(x) + b$ is $g^*(y) = af^*(y/a) - b$.

Sum of Independent Functions

If $f(u,v) = f_1(u) + f_2(v)$ is a sum of two convex functions, then

$$f^*(w,z) = f_1^*(w) + f_2^*(z),$$

where f_1^* and f_2^* are the conjugates of f_1 and f_2 respectively.

A.3 Examples of Convex Functions

Here, we highlight a few examples of convex functions that appear in the thesis.

Sum of k Largest Components

For $x \in \mathbb{R}^n$ we denote by x_{π_i} the ith largest component of x, i.e.

$$x_{\pi_1} \geq x_{\pi_2} \geq \ldots \geq x_{\pi_n}$$

are the components of x sorted in nonincreasing order. Then the function

$$f_k(x) = \sum_{i=1}^{k} x_{\pi_i},$$

i.e., the *sum of the k largest elements of x*, is a convex function. This can be seen by writing it as the maximum of all possible sums of k different components of x,

$$f_k(x) = \max\{x_{i_1} + \ldots + x_{i_k} \,|\, 1 \leq i_1 \leq \ldots \leq i_k \leq n\}.$$

Moreover, it can be shown (Boyd and Vandenberghe, 2004) that

$$g_k(x) = \sum_{i=1}^{k} w_i x_{\pi_i}$$

is convex for $w_1 \geq w_2 \geq \ldots \geq w_k \geq 0$.

Distance Function and the Euclidean Projection

Let $C \subset \mathbb{R}^n$ be a nonempty closed convex set. The *distance function* is defined as

$$d_C(x) \triangleq \min_{y \in C} \|x - y\|,$$

and the *Euclidean projection* is defined as

$$\mathbf{proj}_C(x) \triangleq \arg\min_{y \in C} \|x - y\|,$$

where $\|\cdot\|$ is the ℓ_2 (Euclidean) norm.

Proposition A.5. If $C \subset \mathbb{R}^n$ is a nonempty closed convex set, then

$$\begin{aligned} \nabla d_C(x) &= d_C(x)^{-1}(x - \mathbf{proj}_C(x)), && \forall\, x \notin C, \\ \nabla \tfrac{1}{2} d_C^2(x) &= x - \mathbf{proj}_C(x), && \forall\, x \in \mathbb{R}^n. \end{aligned}$$

The last equation in Proposition A.5 is important as it allows us to compute the gradient of the function

$$f(x) \triangleq \tfrac{1}{2} d_C^2(x) = \tfrac{1}{2} \|x - \mathbf{proj}_C(x)\|^2,$$

if we know how to compute the projection $\mathbf{proj}_C(x)$. This is useful in optimization of the loss functions that are defined in terms of the distance functions.

B Further Details and Results

The material in this chapter provides additional technical details and experimental results that extend the main content of the thesis.

B.1 Multitask Representation Learning

In this section, we give further details on the implementation of our solvers from Chapter 4 (page 61), detailed runtime analysis, and visualization of selected results.

B.1.1 Implementation Details

Here, we discuss implementation details of our STL-SDCA and MTL-SDCA solvers. In particular, we derive closed form SDCA updates and discuss dual optimization based on the Gram matrices. In contrast to § 4.2.1, where for convenience our algorithm is presented using the primal variables, we now discuss our implementation that operates entirely on the dual variables and is, therefore, computationally efficient with high dimensional image descriptors.

First, let us recall some notation. The single task learning (STL) approach learns T linear predictors w_t for each task t which can be thought of as a matrix of primal variables $W \in \mathbb{R}^{d \times T}$. The proposed multitask learning (MTL) method, on the other hand, learns a matrix $U \in \mathbb{R}^{d \times k}$ for the shared feature space, and another matrix $W \in \mathbb{R}^{k \times T}$ for the linear predictors in a low dimensional subspace. We let $X \in \mathbb{R}^{d \times n}$ be the matrix of the original (high dimensional) features, $Y \in \{\pm 1\}^{n \times T}$ the matrix of labels, and $Z = U^\top X$ the matrix of new features in the shared representation. We also define the following Gram matrices: $K = K_X = X^\top X$, $K_Z = Z^\top Z$, and $M = K_W = W^\top W$, where the W in the last formula is for the MTL method. Note that the main expensive operation here is the computation of K_X on the original features, which is done only once. Finally, let $A \in \mathbb{R}^{n \times T}$ be the matrix of stacked dual variables α. When necessary, we use the subscript A_{STL} to distinguish the dual variables of STL and MTL methods.

Next, we derive efficient SDCA updates for the STL and MTL approaches.

STL-SDCA. The STL optimization problem for a task t is defined as follows:

$$\min_{w_t \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} \max\left(0, 1 - y_{it} \langle w_t, x_i \rangle\right) + \frac{\lambda}{2} \|w_t\|_2^2 ,$$

where $\lambda > 0$ is the regularization parameter. This yields the following dual problem, see (Shalev-Shwartz and Zhang, 2013b).

$$\max_{\alpha_t \in \mathbb{R}^n} \frac{1}{n} \sum_{i=1}^{n} y_{it}\alpha_{it} - \frac{\lambda}{2} \Big\| \frac{1}{\lambda n} \sum_{i=1}^{n} \alpha_{it} x_i \Big\|_2^2 \quad \text{s.t.} \;\; 0 \leq y_{it}\alpha_{it} \leq 1 \;\; \text{for all} \;\; i = 1, \ldots, n.$$

At step s, a dual variable is updated via $\alpha_{it}^{(s)} = \alpha_{it}^{(s-1)} + \Delta\alpha_{it}$, where the update $\Delta\alpha_{it}$ can be computed as:

$$\Delta\alpha_{it} = y_{it} \max\left(-y_{it}\alpha_{it}^{(s-1)}, \min\left(1 - y_{it}\alpha_{it}^{(s-1)}, \frac{1 - y_{it}\langle w_t, x_i\rangle}{\frac{1}{\lambda n}\|x_i\|_2^2}\right)\right). \tag{B.1}$$

Note that $\|x_i\|_2^2 = K_{ii} = K_X[i,i]$ and, since

$$w_t = \frac{1}{\lambda n} \sum_{i=1}^{n} \alpha_{it} x_i = \frac{1}{\lambda n} X\alpha_t, \quad W = \frac{1}{\lambda n} XA, \tag{B.2}$$

one obtains

$$\langle w_t, x_i \rangle = \frac{1}{\lambda n} \sum_{j=1}^{n} \alpha_{jt} \langle x_i, x_j \rangle = \frac{1}{\lambda n} K_i^\top \alpha_t = K_i^\top \tilde{\alpha}_t,$$

where $K_i = K_X[:,i]$ is the ith column of K_X and we apply a change of variables $\tilde{\alpha}_{it} = \frac{1}{\lambda n}\alpha_{it}$. From now on, we always use the transformed $\tilde{\alpha}_{it}$ variables and drop the $\tilde{}$ notation for convenience.

The vector $K\alpha_t \in \mathbb{R}^n$ can be precomputed using the initial $\alpha_t^{(0)}$ and then updated whenever $\Delta\alpha_{it} \neq 0$ as follows:

$$K\alpha_t^{(s)} = K\alpha_t^{(s-1)} + \Delta\alpha_{it} K_i. \tag{B.3}$$

Note that this update, as well as the $K\alpha_t$ itself, can be computed efficiently using BLAS routines `xAXPY` and `xGEMV`. Let

$$h = \frac{1 - y_{it} K_i^\top \alpha_t}{K_{ii}} \quad \text{and} \quad C = \frac{1}{\lambda n}, \tag{B.4}$$

where $K_i^\top \alpha_t$ is the ith element of the precomputed vector $K\alpha_t$. Then $\alpha_{it}^{(s)}$ can be computed directly as follows:

$$\alpha_{it}^{(s)} = \begin{cases} \max\left(0, \min\left(C, \alpha_{it}^{(s-1)} + h\right)\right) & \text{if } y_{it} = +1, \\ \max\left(-C, \min\left(0, \alpha_{it}^{(s-1)} - h\right)\right) & \text{if } y_{it} = -1. \end{cases} \tag{B.5}$$

Based on (B.5), the update (B.1) can be shown to be 0, i.e. $\Delta\alpha_{it} = 0$, in the following two cases which typically hold for most of the data points after the first few epochs. Note that if the update is zero, then computation of (B.3) and (B.4) is avoided:

$$\Delta\alpha_{it} = 0 \quad \text{if } \left(\alpha_{it}^{(s-1)} = 0 \wedge h \leq 0\right) \vee \left(\alpha_{it}^{(s-1)} = y_{it}C \wedge h \geq 0\right).$$

The intuition here is that if a point is not a support vector ($\alpha_{it}^{(s-1)} = 0$) and there is no loss on this example ($h \leq 0$), then there is no incentive for the data point to become a support vector. Similarly, if there is some non-negative loss ($h \geq 0$), but the point already exerts the maximum force ($\alpha_{it}^{(s-1)} = y_{it}C$), then it will not be updated.

Since $K_{ii} > 0$ (we skip examples with $K_{ii} = 0$), the conditions $h \gtreqless 0$ can be simplified and one obtains

$$\Delta\alpha_{it} = 0 \text{ if } \left(\alpha_{it}^{(s-1)} = 0 \wedge y_{it}K_i^\top \alpha_t \geq 1\right) \vee \left(\alpha_{it}^{(s-1)} = y_{it}C \wedge y_{it}K_i^\top \alpha_t \leq 1\right). \tag{B.6}$$

U-SDCA. Our MTL-SDCA algorithm alternates between learning the predictors w_t via STL-SDCA on Z and learning the matrix U via an algorithm that we call U-SDCA. Let W be fixed. The problem of learning U is formulated as

$$\min_{U \in \mathbb{R}^{d \times k}} \frac{1}{nT} \sum_{i=1}^{n} \sum_{t=1}^{T} \max\left(0, 1 - y_{it}\left\langle w_t, U^\top x_i\right\rangle\right) + \frac{\mu}{2}\left\|U\right\|_F^2,$$

where $\mu > 0$ is the regularization parameter and $\|\cdot\|_F$ is the Frobenius norm. The corresponding dual problem is given below.

$$\begin{aligned} \max_{A \in \mathbb{R}^{n \times T}} \quad & \frac{1}{nT} \sum_{i,t} y_{it}\alpha_{it} - \frac{\mu}{2}\left\| \frac{1}{\mu nT} \sum_{i,t} \alpha_{it} x_i w_t^\top \right\|_F^2 \\ \text{subject to} \quad & 0 \leq y_{it}\alpha_{it} \leq 1 \quad \text{for all} \quad i = 1, \ldots, n, \ t = 1, \ldots, T. \end{aligned}$$

Similar to STL-SDCA, an update $\Delta\alpha_{it}$ can be computed as follows:

$$\Delta\alpha_{it} = y_{it} \max\left(- y_{it}\alpha_{it}^{(s-1)}, \min\left(1 - y_{it}\alpha_{it}^{(s-1)}, \frac{1 - y_{it}\left\langle w_t, U^\top x_i\right\rangle}{\frac{1}{\mu nT}\left\|x_i\right\|_2^2 \left\|w_t\right\|_2^2}\right)\right),$$

Note that $\|x_i\|_2^2 = K_{ii} = K_X[i,i]$, $\|w_t\|_2^2 = M_{tt} = K_W[t,t]$ and, since

$$U = \frac{1}{\mu nT} \sum_{i,t} \alpha_{it} x_i w_t^\top = \frac{1}{\mu nT} XAW^\top, \tag{B.7}$$

one obtains

$$\left\langle w_t, U^\top x_i\right\rangle = \frac{1}{\mu nT} \sum_{j,s} \alpha_{js} \left\langle x_i, x_j\right\rangle \left\langle w_s, w_t\right\rangle = \frac{1}{\mu nT} K_i^\top A M_t = K_i^\top \tilde{A} M_t,$$

Algorithm B.1 MTL-SDCA (implementation details)

```
Input: labels $Y$, matrices $K_X$ and $K_Z$, parameters $\lambda$, $\mu$, and $\epsilon$
Let: $A, A_{\text{old}}, B, B_{\text{old}} = \mathbf{0}$  // $A$, $B$ are the dual variables for $U$, $W$ respectively
loop
  Update $B$ via STL-SDCA($Y$, $K_Z$, $\lambda$)
  Let $K_W = B^\top K_Z B$  // since $W = ZB$, see (B.2)
  Update $A$ via U-SDCA($Y$, $K_X$, $K_W$, $\mu$)
  if RMSE$\left((A, B) - (A_{\text{old}}, B_{\text{old}})\right) < \epsilon$ then
    break
  end if
  Let $K_Z = K_X A K_W A^\top K_X$   // since $Z = U^\top X$ and $U = XAW^\top$, see (B.7)
  Let $A_{\text{old}} = A$, $B_{\text{old}} = B$
end loop
return  $A$, $K_W$
```

where $K_i = K_X[:, i]$ is the ith column of K_X, $M_t = K_W[:, t]$ is the tth column of K_W and we apply a change of variables $\tilde{\alpha}_{it} = \frac{1}{\mu n T} \alpha_{it}$. As before, we always use the transformed $\tilde{\alpha}_{it}$ variables and drop the ˜ notation.

Note that the matrix A now introduces coupling between all tasks and all examples, hence every non-zero update $\Delta\alpha_{it}$ affects all scores $\left\langle w_t, U^\top x_i \right\rangle$. We experimented with one approach where the whole matrix KAM is precomputed and then updated via rank-1 updates $\Delta\alpha_{it} x_i w_t^\top$ (using BLAS routine `xGER`). However, this strategy seemed inferior in terms of runtime when compared to the approach we present next (most likely due to less efficient memory access pattern).

Instead of sampling both i and t at every iteration, we proceed as follows. At each epoch, we iterate over all tasks in random order (task IDs are permuted at the beginning of the epoch) and precompute KAM_t for a given task t. Then we iterate over all examples in random order and update the *vector* KAM_t in $\mathbb{R}^n$ similar to (B.3):

$$KA^{(s)} M_t = KA^{(s-1)} M_t + \Delta\alpha_{it} K_i M_{tt}.$$

The formula (B.5) for $\alpha_{it}^{(s)}$ remains unchanged, while the h and C are now computed differently:

$$h = \frac{1 - y_{it} K_i^\top A M_t}{K_{ii} M_{tt}} \quad \text{and} \quad C = \frac{1}{\mu n T},$$

where $K_i^\top A M_t$ is the ith element of the precomputed vector KAM_t. Similarly, the condition (B.6) becomes

$$\Delta\alpha_{it} = 0 \;\text{ if }\; \left(\alpha_{it}^{(s-1)} = 0 \,\wedge\, y_{it} K_i^\top A M_t \geq 1\right) \vee \left(\alpha_{it}^{(s-1)} = y_{it} C \,\wedge\, y_{it} K_i^\top A M_t \leq 1\right).$$

MTL-SDCA. We now describe the master problem of the MTL-SDCA algorithm. Recall that the joint problem is nonconvex and we use an STL solution as the initial

Routine	STL	MTL	MTL/STL
Prepare image encoder (fit GMM for FV)	1.4 hours		–
Compute FV descriptors (train and testsubsets)	9.3 hours		–
Compute train and test kernels	11.1 mins		–
Training time (SDCA optimization problem)	2.2 mins	24.9 mins	11.23
+ compute kernels + compute the initial point	8.0 mins	32.9 mins	4.13
+ compute descriptors for training images	6.2 hours	6.7 hours	1.07

Table B.1.: Runtime comparison for the STL and MTL methods (**wall-clock** time).

Routine	STL	MTL	MTL/STL
Prepare image encoder (fit GMM for FV)	7.3 hours		–
Compute FV descriptors (train and testsubsets)	3.3 days		–
Compute train and test kernels	2.8 hours		–
Training time (SDCA optimization problem)	15.8 mins	5.8 hours	22.00
+ compute kernels + compute the initial point	1.7 hours	7.5 hours	4.43
+ compute descriptors for training images	2.0 days	2.3 days	1.12

Table B.2.: Runtime comparison for the STL and MTL methods (**CPU** time).

point, i.e. $U^{(0)} = W_{\text{STL}} = XA_{\text{STL}}$, where A_{STL} are the dual variables computed by STL-SDCA on X (using the kernel K_X). It follows that

$$Z = {U^{(0)}}^\top X = A_{\text{STL}}^\top K_X \quad \text{and} \quad K_Z = Z^\top Z.$$

MTL-SDCA takes Y, K_X, K_Z, λ, and μ as input and outputs A and K_W. Since $U = XAW^\top$, the test scores are given as:

$$W^\top Z_{\text{tst}} = W^\top U^\top X_{\text{tst}} = K_W A^\top K_{\text{tst}},$$

where $K_{\text{tst}} = X^\top X_{\text{tst}}$. The procedure is summarized in Algorithm B.1.

Note that both STL-SDCA and U-SDCA support warm restart, hence the dual variables A and B are actually updated rather than recomputed from scratch.

B.1.2 Runtime Analysis

To estimate the overhead of the proposed MTL method relative to the standard STL approach, we perform a single run of the full pipeline for the best performing setting: SIFT+LCS+PN+L2 features, $n_{\text{class}} = 50$ examples per class. Both solvers are compiled using GCC version 4.4 with the `-O3` option and linked to the Intel MKL library version 11.1 for the BLAS subroutines. The experiment is run on a 32 core 64 bit 2.7 GHz Intel CPU machine with 256 GB of RAM.

Tables B.1 and B.2 report respectively the elapsed wall-clock and CPU time for the different steps of the pipeline excluding the model selection step. Note that most of the time (over 9 hours) is spent in the computation of image descriptors, where the Fisher Vector encoding is the most expensive operation compared to SIFT and LCS feature extraction. MTL-SDCA training takes more time than STL-SDCA, but is faster than training an encoder, where most of the time is spent in SIFT and LCS extraction and in learning a visual words vocabulary. Moreover, the MTL time also includes the STL-SDCA training on the original features to obtain the initial point for the MTL-SDCA algorithm.

We conclude from the presented runtime comparison that the relative overhead of the proposed multitask learning method is rather small when other steps of the pipeline are taken into account (roughly a factor of 4 when all features are precomputed and about 12% otherwise).

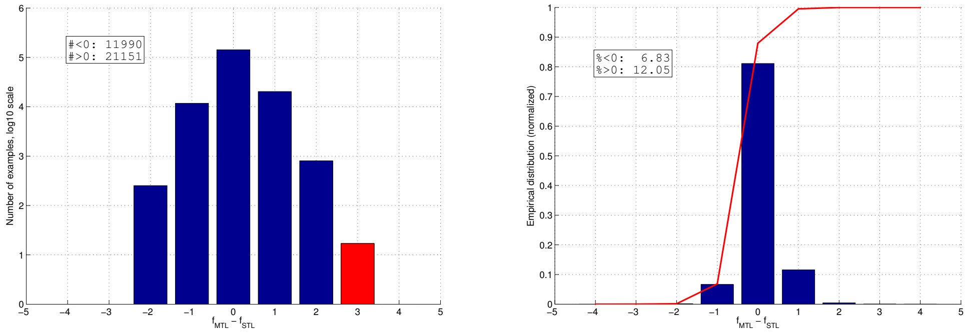


Figure B.1.: Top-5 prediction results modulo human confusions. For each method, the top-5 predictions are intersected with human predictions to count "reasonable" confusions, denoted $f_{\mathrm{STL/MTL}}$. The plots show the distribution of $f_{\mathrm{MTL}} - f_{\mathrm{STL}}$ over all 10 splits of SUN 397. **Left:** counts on the $\log_{10}$ scale. **Right:** normalized distribution. The highlighted (red) bar shows that MTL results agree with the human predictions more often than STL.

B.1.3 Top-5 Predictions Modulo Human Confusions

Next, we show that the proposed MTL method not only improves the top-k accuracy for different k, but also tends to produce more reasonable confusions in the following sense. Let $P^5_{\mathrm{STL}}(x_i)$ be a set of top-5 prediction results for the baseline STL method on a given image x_i, similarly, let $P^5_{\mathrm{MTL}}(x_i)$ be a set of top-5 predictions of the proposed MTL approach, finally, let P_{human} be a set of all classes that AMT workers confused with the given class based on the confusion matrix of "good workers". Let

$$f_{\mathrm{STL}}(x_i) = \left|P^5_{\mathrm{STL}}(x_i) \cap P_{\mathrm{human}}\right|, \qquad f_{\mathrm{MTL}}(x_i) = \left|P^5_{\mathrm{MTL}}(x_i) \cap P_{\mathrm{human}}\right|,$$

where $|A|$ is the cardinality of a set A. Figure B.1 shows the distribution of $\Delta_f = f_{\mathrm{MTL}} - f_{\mathrm{STL}}$ on the test sets across all 10 splits of the SUN 397 dataset. Note that $\Delta_f > 0$ for more test examples than $\Delta_f < 0$, which means there is a tendency for the MTL method to produce more human-like confusions.

B.1.4 Selected Prediction Results on SUN 397

Finally, we visualize top-5 predictions of STL and MTL methods on a few selected examples where MTL produces more human-like confusions. Human performance is estimated based on the confusion matrix of "good workers" provided by Xiao et al., (2010). Classifiers are trained using the SIFT descriptor with the Hellinger kernel and $n_{\text{class}} = 20$ images per class.

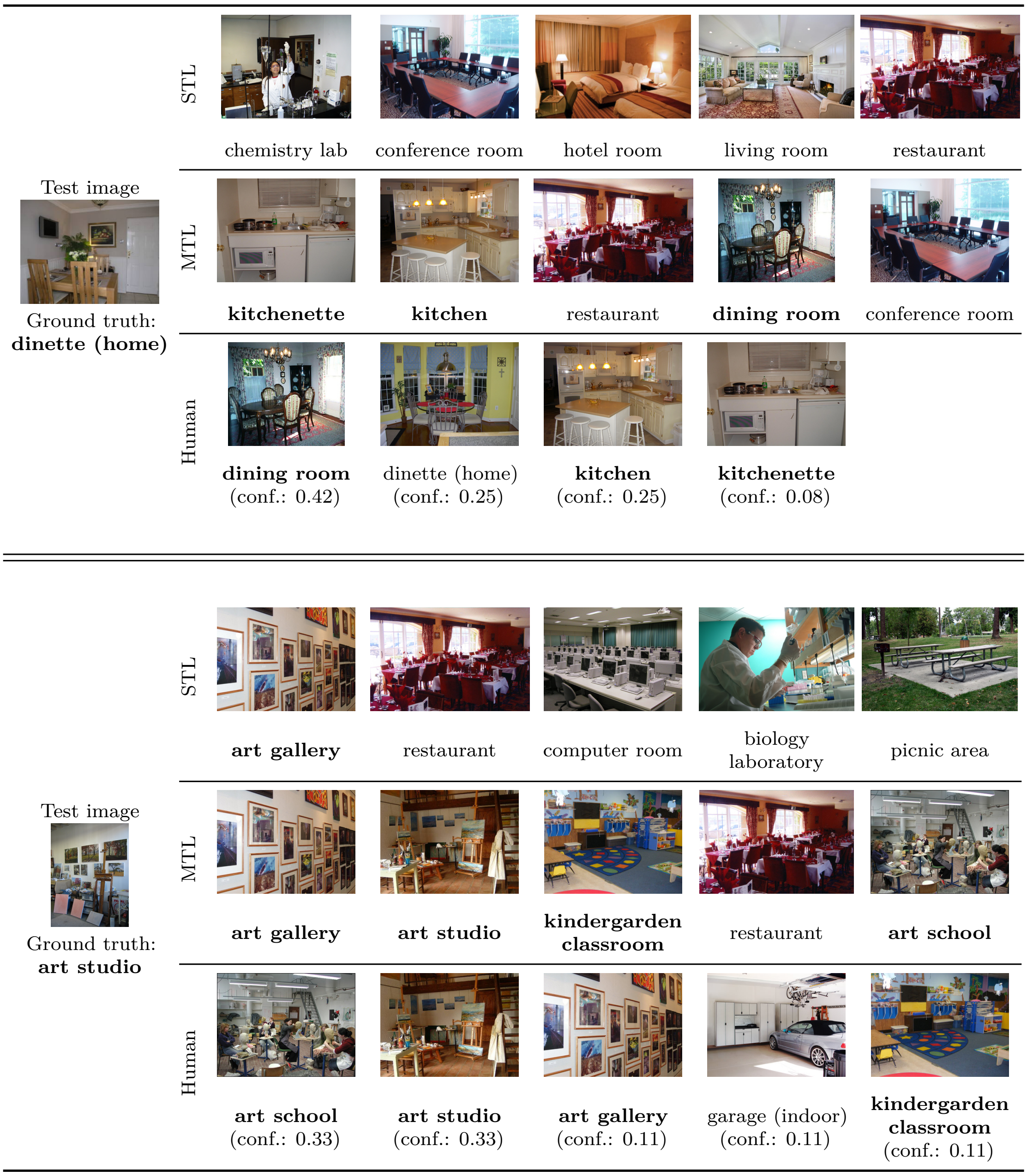


Figure B.2.: Multitask learning improves upon single task learning in top-k accuracy and in agreement with human predictions. **STL:** top-5 OVA SVM predictions. **MTL:** top-5 predictions of our method. **Human:** estimated human predictions and confidence scores based on the provided confusion matrix. Labels marked **bold** indicate agreement with human predictions.

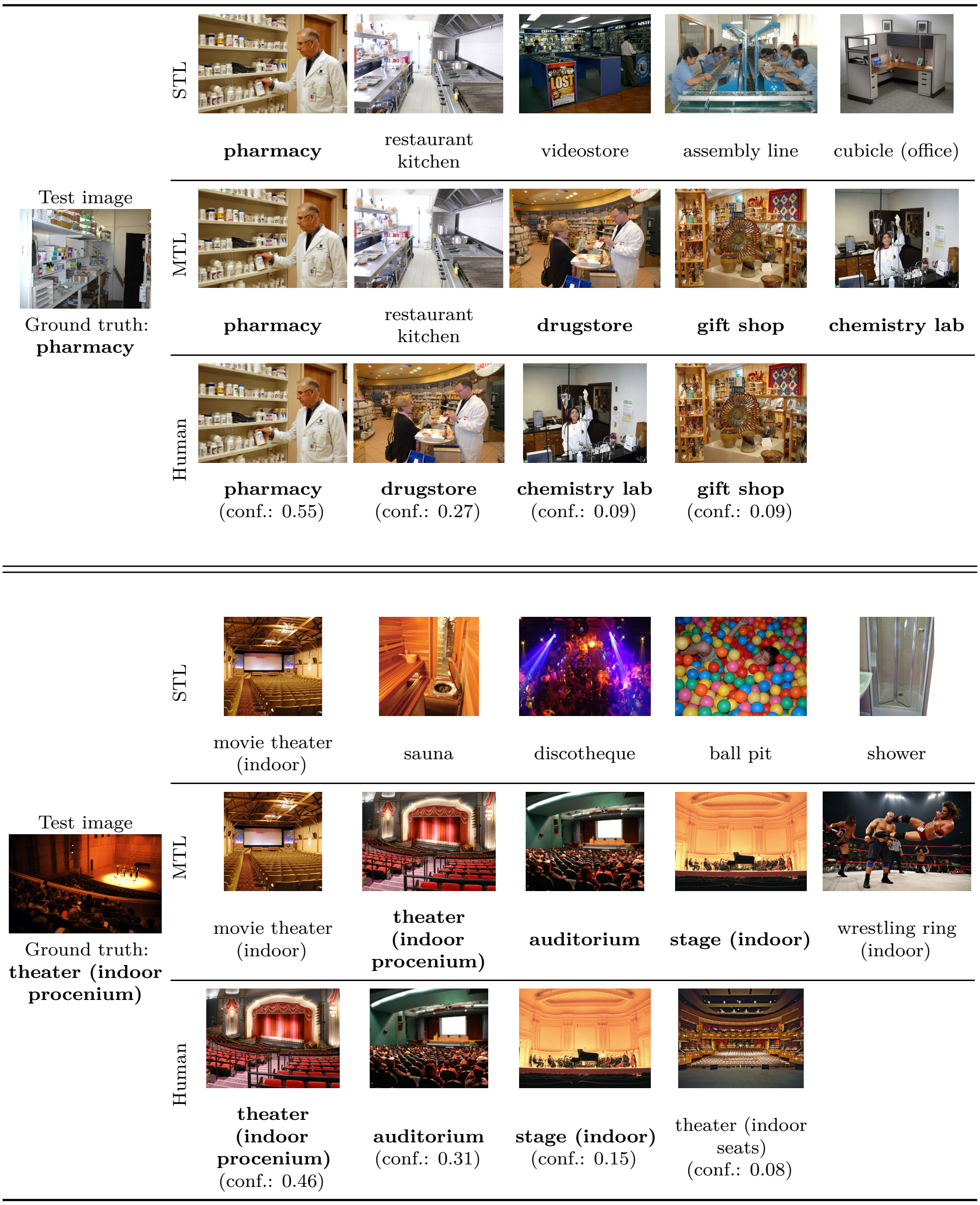


Figure B.3.: Continuation of Figure B.2.

Bibliography

Agarwal, S. (2011). “The Infinite Push: A New Support Vector Ranking Algorithm that Directly Optimizes Accuracy at the Absolute Top of the List.” In: *SDM*, pp. 839–850 (cit. on p. 26).

Agarwal, S. (2013). “Surrogate Regret Bounds for the Area Under the ROC Curve via Strongly Proper Losses.” In: *COLT*, pp. 338–353 (cit. on p. 28).

Aggarwal, A., S. Ghoshal, M. Ankith, S. Sinha, G. Ramakrishnan, P. Kar, and P. Jain (2017). “Scalable Optimization of Multivariate Performance Measures in Multi-instance Multi-label Learning.” In: *AAAI* (cit. on p. 27).

Akata, Z., F. Perronnin, Z. Harchaoui, and C. Schmid (2014). “Good practice in large-scale learning for image classification.” In: *PAMI* 36.3, pp. 507–520 (cit. on p. 159).

Amit, Y., M. Fink, N. Srebro, and S. Ullman (2007). “Uncovering shared structures in multiclass classification.” In: *ICML*, pp. 17–24 (cit. on p. 24).

Ando, R. K. and T. Zhang (2005). “A framework for learning predictive structures from multiple tasks and unlabeled data.” In: *JMLR* 6, pp. 1817–1853 (cit. on pp. 24, 25).

Argyriou, A., T. Evgeniou, and M. Pontil (2007). “Multi-task feature learning.” In: *NIPS* 19, p. 41 (cit. on p. 25).

Argyriou, A., T. Evgeniou, and M. Pontil (2008). “Convex multi-task feature learning.” In: *Machine Learning* 73.3, pp. 243–272 (cit. on pp. 24, 62, 64).

Ballard, D. H. and C. M. Brown (1982). *Computer vision.* Prentice-Hall (cit. on p. 6).

Banko, M. and E. Brill (2001). “Scaling to very very large corpora for natural language disambiguation.” In: *ACL*, pp. 26–33 (cit. on p. 5).

Barla, A., F. Odone, and A. Verri (2003). “Histogram intersection kernel for image classification.” In: *ICIP.* Vol. 3, pp. III–513 (cit. on p. 18).

Barry, D. A., S. J. Barry, and P. J. Culligan-Hensley (1995). “Algorithm 743: WAPR: A Fortran routine for calculating real values of the W-function.” In: *ACM Transactions on Mathematical Software* 21.2, pp. 172–181 (cit. on p. 24).

Barry, D. A., J.-Y. Parlange, L. Li, H. Prommer, C. Cunningham, and F. Stagnitti (2000). “Analytical approximations for real values of the Lambert W-function.” In: *Mathematics and Computers in Simulation* 53.1, pp. 95–103 (cit. on p. 24).

Bartlett, P. L., M. I. Jordan, and J. D. McAuliffe (2006). “Convexity, classification and risk bounds.” In: *JASA* 101, pp. 138–156 (cit. on pp. 4, 28, 129).

Baxter, J. et al. (2000). “A model of inductive bias learning.” In: *Journal of Artificial Intelligence Research* 12.149-198, p. 3 (cit. on p. 24).

Beck, A. and M. Teboulle (2012). “Smoothing and first order methods, a unified framework.” In: *SIOPT* 22 (2), pp. 557–580 (cit. on p. 115).

Beck, A. and M. Teboulle (2009). "A fast iterative shrinkage-thresholding algorithm for linear inverse problems." In: *SIAM Journal on Imaging Sciences* 2.1, pp. 183–202 (cit. on pp. 23, 24, 138, 149).

Belkin, M. and P. Niyogi (2003). "Laplacian eigenmaps for dimensionality reduction and data representation." In: *Neural Computation* 15.6, pp. 1373–1396 (cit. on p. 7).

Ben-David, S., N. Eiron, and P. M. Long (2003). "On the difficulty of approximately maximizing agreements." In: *Journal of Computer and System Sciences* 66.3, pp. 496–514 (cit. on pp. 4, 28).

Ben-David, S. and R. Schuller (2003). "Exploiting task relatedness for multiple task learning." In: *Learning Theory and Kernel Machines*. Springer, pp. 567–580 (cit. on p. 24).

Bengio, Y. (2009). "Learning deep architectures for AI." In: *Foundations and trends® in Machine Learning* 2.1, pp. 1–127 (cit. on p. 20).

Bengio, Y., A. Courville, and P. Vincent (2013). "Representation learning: A review and new perspectives." In: *PAMI* 35.8, pp. 1798–1828 (cit. on p. 9).

Berger, A. L., V. J. D. Pietra, and S. A. D. Pietra (1996). "A maximum entropy approach to natural language processing." In: *Computational Linguistics* 22.1, pp. 39–71 (cit. on p. 10).

Bertsekas, D. P. (1982). "Projected Newton methods for optimization problems with simple constraints." In: *SIAM Journal on Control and Optimization* 20.2, pp. 221–246 (cit. on p. 23).

Besl, P. J. and R. C. Jain (1985). "Three-dimensional object recognition." In: *ACM Computing Surveys (CSUR)* 17.1, pp. 75–145 (cit. on p. 7).

Bickel, S., M. Brückner, and T. Scheffer (2007). "Discriminative learning for differing training and test distributions." In: *ICML*, pp. 81–88 (cit. on p. 22).

Bilen, H. and A. Vedaldi (2017). "Universal representations: The missing link between faces, text, planktons, and cat breeds." In: *arXiv preprint arXiv:1701.07275* (cit. on p. 25).

Bo, L., K. Lai, X. Ren, and D. Fox (2011). "Object recognition with hierarchical kernel descriptors." In: *CVPR*, pp. 1729–1736 (cit. on p. 19).

Bo, L., X. Ren, and D. Fox (2010). "Kernel descriptors for visual recognition." In: *NIPS*, pp. 244–252 (cit. on p. 18).

Bordes, A., L. Bottou, P. Gallinari, and J. Weston (2007). "Solving multiclass support vector machines with LaRank." In: *ICML*, pp. 89–96 (cit. on p. 87).

Borwein, J. M. and A. S. Lewis (2000). *Convex Analysis and Nonlinear Optimization: Theory and Examples*. Cms Books in Mathematics Series. Springer Verlag (cit. on pp. 47, 138, 183, 185).

Boureau, Y.-L., F. Bach, Y. LeCun, and J. Ponce (2010). "Learning mid-level features for recognition." In: *CVPR*, pp. 2559–2566 (cit. on p. 19).

Bousquet, O. and L. Bottou (2008). "The tradeoffs of large scale learning." In: *NIPS*, pp. 161–168 (cit. on p. 92).

Bousquet, O. and A. Elisseeff (2002). "Stability and generalization." In: *JMLR* 2, pp. 499–526 (cit. on p. 7).

Boutell, M. R., J. Luo, X. Shen, and C. M. Brown (2004). "Learning multi-label scene classification." In: *Pattern Recognition* 37.9, pp. 1757–1771 (cit. on p. 162).

Boyd, S., C. Cortes, M. Mohri, and A. Radovanovic (2012). “Accuracy at the top.” In: *NIPS*, pp. 953–961 (cit. on p. 26).

Boyd, S. and L. Vandenberghe (2004). *Convex Optimization*. Cambridge University Press (cit. on pp. 4, 121, 179, 185, 186).

Breiman, L., J. Friedman, C. J. Stone, and R. A. Olshen (1984). *Classification and regression trees*. CRC press (cit. on p. 22).

Brucker, P. (1984). “An $O(n)$ algorithm for quadratic knapsack problems.” In: *Operations Research Letters* 3.3, pp. 163–166 (cit. on p. 23).

Bu, S., Z. Liu, J. Han, and J. Wu (2013). “Superpixel segmentation based structural scene recognition.” In: *ACM-MM*, pp. 681–684 (cit. on p. 102).

Buffoni, D., P. Gallinari, N. Usunier, and C. Calauzénes (2011). “Learning scoring functions with order-preserving losses and standardized supervision.” In: *ICML*, pp. 825–832 (cit. on p. 28).

Bunescu, R. C. and R. J. Mooney (2007). “Multiple instance learning for sparse positive bags.” In: *ICML*, pp. 105–112 (cit. on p. 21).

Burges, C., T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender (2005). “Learning to rank using gradient descent.” In: *ICML*, pp. 89–96 (cit. on p. 26).

Burges, C. J. C. and D. J. Crisp (1999). “Uniqueness of the SVM solution.” In: *NIPS*, pp. 223–229 (cit. on p. 37).

Burl, M. C., M. Weber, and P. Perona (1998). “A probabilistic approach to object recognition using local photometry and global geometry.” In: *ECCV*, pp. 628–641 (cit. on p. 6).

Burns, J. B., R. S. Weiss, and E. M. Riseman (1992). “The non-existence of general-case view-invariants.” In: *Geometric Invariance in Computer Vision* 1, pp. 554–559 (cit. on p. 7).

Calauzenes, C., N. Usunier, P. Gallinari, et al. (2012). “On the (Non-)existence of Convex, Calibrated Surrogate Losses for Ranking.” In: *NIPS*, pp. 197–205 (cit. on p. 28).

Cao, Z., T. Qin, T.-Y. Liu, M.-F. Tsai, and H. Li (2007). “Learning to rank: from pairwise approach to listwise approach.” In: *ICML* (cit. on p. 26).

Carreira-Perpiñán, M. Á. and W. Wang (2014). “Distributed optimization of deeply nested systems.” In: *AISTATS*, pp. 10–19 (cit. on p. 177).

Caruana, R. (1997). “Multitask learning.” In: *Machine learning* 28.1, pp. 41–75 (cit. on pp. 24, 62).

Chang, C.-C. and C.-J. Lin (2011). “LIBSVM: A library for support vector machines.” In: *ACM TIST* 2 (3), pp. 1–27 (cit. on pp. 55, 159).

Chapeau-Blondeau, F. and A. Monir (2002). “Numerical evaluation of the Lambert W function and application to generation of generalized Gaussian noise with exponent 1/2.” In: *IEEE Transactions on Signal Processing* 50.9, pp. 2160–2165 (cit. on p. 24).

Chapelle, O. (2007). “Training a support vector machine in the primal.” In: *Neural Computation* 19.5, pp. 1155–1178 (cit. on p. 52).

Chapelle, O., P. Haffner, and V. N. Vapnik (1999). “Support vector machines for histogram-based image classification.” In: *IEEE Transactions on Neural Networks* 10.5, pp. 1055–1064 (cit. on p. 18).

Chapelle, O., B. Schölkopf, and A. Zien (2006). *Semi-Supervised Learning.* The MIT Press (cit. on p. 31).

Chapelle, O., V. Vapnik, O. Bousquet, and S. Mukherjee (2002). "Choosing multiple parameters for support vector machines." In: *Machine learning* 46.1-3, pp. 131–159 (cit. on p. 52).

Chatfield, K., V. Lempitsky, A. Vedaldi, and A. Zisserman (2011). "The devil is in the details: an evaluation of recent feature encoding methods." In: *BMVC* (cit. on pp. 19, 62).

Chawla, N. V., N. Japkowicz, and A. Kotcz (2004). "Editorial: Special issue on learning from imbalanced data sets." In: *ACM SIGKDD Explorations Newsletter* 6.1, pp. 1–6 (cit. on p. 22).

Chen, J., X. Liu, and S. Lyu (2012). "Boosting with side information." In: *ACCV*, pp. 5–9 (cit. on p. 21).

Chen, J., L. Tang, J. Liu, and J. Ye (2013). "A convex formulation for learning a shared predictive structure from multiple tasks." In: *PAMI* 35.5, pp. 1025–1038 (cit. on p. 24).

Chen, K. and J.-K. Kamarainen (2014). "Learning to Count with Back-propagated Information." In: *ICPR*, pp. 4672–4677 (cit. on p. 21).

Cho, Y. and L. K. Saul (2009). "Kernel methods for deep learning." In: *NIPS*, pp. 342–350 (cit. on p. 19).

Cimpoi, M., S. Maji, and A. Vedaldi (2015). "Deep Filter Banks for Texture Recognition and Segmentation." In: *CVPR* (cit. on p. 158).

Cireşan, D., U. Meier, and J. Schmidhuber (2012). "Multi-column deep neural networks for image classification." In: *CVPR*, pp. 3642–3649 (cit. on pp. 18, 20).

Clinchant, S., G. Csurka, F. Perronnin, and J.-M. Renders (2007). "XRCE's participation to ImagEval." In: *ImageEval workshop at CVIR* (cit. on p. 70).

Collobert, R. and J. Weston (2008). "A unified architecture for natural language processing: Deep neural networks with multitask learning." In: *ICML*, pp. 160–167 (cit. on p. 24).

Condat, L. (2014). "Fast projection onto the simplex and the ℓ_1 ball." In: *Math. Prog.* Pp. 1–11 (cit. on pp. 23, 99).

Corless, R. M., G. H. Gonnet, D. E. Hare, D. J. Jeffrey, and D. E. Knuth (1996). "On the Lambert W function." In: *Advances in Computational Mathematics* 5.1, pp. 329–359 (cit. on pp. 24, 138, 140, 145).

Cortes, C., Y. Mansour, and M. Mohri (2010). "Learning bounds for importance weighting." In: *NIPS*, pp. 442–450 (cit. on p. 22).

Cortes, C., M. Mohri, V. Kuznetsov, and S. Yang (2016). "Structured Prediction Theory and Voted Risk Minimization." In: *arXiv:1605.06443* (cit. on p. 28).

Cortes, C. and V. Vapnik (1995). "Support-vector networks." In: *Machine learning* 20.3, pp. 273–297 (cit. on pp. 10, 52).

Cossock, D. and T. Zhang (2006). "Subset ranking using regression." In: *COLT*, pp. 605–619 (cit. on p. 26).

Cossock, D. and T. Zhang (2008). "Statistical analysis of Bayes optimal subset ranking." In: *IEEE Trans. Inf. Theory* 54.11, pp. 5140–5154 (cit. on p. 28).

Crammer, K. and Y. Singer (2001). “On the algorithmic implementation of multiclass kernel-based vector machines.” In: *JMLR* 2, pp. 265–292 (cit. on pp. 10, 67, 80, 82, 86, 88, 100, 102, 106, 114, 176, 177).

Crammer, K. and Y. Singer (2003). “A family of additive online algorithms for category ranking.” In: *JMLR* 3, pp. 1025–1058 (cit. on pp. 13, 108, 122, 176).

Csurka, G., C. Dance, L. Fan, J. Willamowski, and C. Bray (2004). “Visual categorization with bags of keypoints.” In: *Workshop on statistical learning in computer vision, ECCV*. Vol. 1. 1-22, pp. 1–2 (cit. on pp. 8, 18).

Dai, J., K. He, and J. Sun (2016). “Instance-aware semantic segmentation via multi-task network cascades.” In: *CVPR*, pp. 3150–3158 (cit. on p. 25).

Dalal, N. and B. Triggs (2005). “Histograms of oriented gradients for human detection.” In: *CVPR*. Vol. 1, pp. 886–893 (cit. on p. 18).

Dattorro, J. (2010). *Convex optimization & Euclidean distance geometry*. Lulu.com (cit. on p. 97).

Decoste, D. and B. Schölkopf (2002). “Training invariant support vector machines.” In: *Machine Learning* 46.1-3, pp. 161–190 (cit. on p. 17).

Dembczynski, K. J., W. Waegeman, W. Cheng, and E. Hüllermeier (2011). “An exact algorithm for F-measure maximization.” In: *NIPS*, pp. 1404–1412 (cit. on p. 28).

Dembczynski, K., W. Cheng, and E. Hüllermeier (2010). “Bayes optimal multilabel classification via probabilistic classifier chains.” In: *ICML*, pp. 279–286 (cit. on p. 112).

Devroye, L., L. Györfi, and G. Lugosi (2013). *A probabilistic theory of pattern recognition*. Vol. 31. Springer Science & Business Media (cit. on p. 2).

Doersch, C., A. Gupta, and A. A. Efros (2013). “Mid-level visual element discovery as discriminative mode seeking.” In: *NIPS*, pp. 494–502 (cit. on p. 102).

Domingos, P. (1998). “How to get a free lunch: A simple cost model for machine learning applications.” In: *Proceedings of AAAI-98/ICML-98 workshop on the methodology of applying machine learning*, pp. 1–7 (cit. on p. 22).

Domingos, P. (1999). “MetaCost: A general method for making classifiers cost-sensitive.” In: *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 155–164 (cit. on p. 22).

Donahue, J., Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell (2013). “Decaf: A deep convolutional activation feature for generic visual recognition.” In: *arXiv:1310.1531* (cit. on p. 72).

Duchi, J. C., L. W. Mackey, and M. I. Jordan (2010). “On the consistency of ranking algorithms.” In: *ICML*, pp. 327–334 (cit. on p. 28).

Duda, R. O., P. E. Hart, and D. G. Stork (2012). *Pattern classification*. John Wiley & Sons (cit. on pp. 3, 111).

Dudík, M., R. E. Schapire, and S. J. Phillips (2005). “Correcting sample selection bias in maximum entropy density estimation.” In: *NIPS* 17, pp. 323–330 (cit. on p. 22).

Duygulu, P., K. Barnard, J. F. de Freitas, and D. A. Forsyth (2002). “Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary.” In: *ECCV*, pp. 97–112 (cit. on p. 162).

Elisseeff, A. and J. Weston (2001). “A kernel method for multi-labelled classification.” In: *NIPS*, pp. 681–687 (cit. on pp. 112, 122, 162, 176, 177).

Elkan, C. (2001). “The foundations of cost-sensitive learning.” In: *IJCAI*, pp. 973–978 (cit. on pp. 22, 32).

Everingham, M., L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman (2010). “The PASCAL visual object classes (VOC) challenge.” In: *IJCV* 88.2, pp. 303–338 (cit. on pp. 109, 112, 162).

Evgeniou, T., C. A. Micchelli, and M. Pontil (2005). “Learning multiple tasks with kernel methods.” In: *JMLR* 6.Apr, pp. 615–637 (cit. on p. 24).

Fan, R.-E., K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin (2008). “LIBLINEAR: A Library for Large Linear Classification.” In: *JMLR* 9, pp. 1871–1874 (cit. on pp. 101, 141, 158).

Felzenszwalb, P. F., R. B. Girshick, D. McAllester, and D. Ramanan (2010). “Object detection with discriminatively trained part-based models.” In: *PAMI* 32.9, pp. 1627–1645 (cit. on p. 6).

Felzenszwalb, P. F. and D. P. Huttenlocher (2005). “Pictorial structures for object recognition.” In: *IJCV* 61.1, pp. 55–79 (cit. on p. 6).

Felzenszwalb, P., D. McAllester, and D. Ramanan (2008). “A discriminatively trained, multiscale, deformable part model.” In: *CVPR* (cit. on p. 67).

Fercoq, O. and P. Richtárik (2015). “Accelerated, parallel, and proximal coordinate descent.” In: *SIOPT* 25.4, pp. 1997–2023 (cit. on p. 23).

Fergus, R., P. Perona, and A. Zisserman (2003). “Object class recognition by unsupervised scale-invariant learning.” In: *CVPR* (cit. on pp. 6, 18).

Feyereisl, J. and U. Aickelin (2012). “Privileged information for data clustering.” In: *Information Sciences* 194, pp. 4–23 (cit. on p. 21).

Fitsch, F. N., R. E. Shafer, and W. P. Crowley (1973). “Algorithm 443: Solution of the Transcendental Equation $we^w = x$.” In: *Communications of the AMC* 16.2, pp. 123–124 (cit. on p. 24).

Forsyth, D. A. and J. Ponce (2003). *Computer Vision: A Modern Approach.* Prentice Hall (cit. on p. 6).

Fouad, S., P. Tino, S. Raychaudhury, and P. Schneider (2012). “Learning using privileged information in prototype based models.” In: *ICANN*, pp. 322–329 (cit. on p. 21).

Frank, A. and A. Asuncion (2010). *UCI Machine Learning Repository.* URL: http://archive.ics.uci.edu/ml (cit. on p. 57).

Frénay, B. and M. Verleysen (2014). “Classification in the presence of label noise: a survey.” In: *NNLS* 25.5, pp. 845–869 (cit. on p. 27).

Freund, Y., R. Iyer, R. E. Schapire, and Y. Singer (2003). “An efficient boosting algorithm for combining preferences.” In: *JMLR* 4.Nov, pp. 933–969 (cit. on p. 26).

Friedman, J., T. Hastie, and R. Tibshirani (2001). *The elements of statistical learning.* Springer (cit. on pp. 3, 10, 111, 128).

Fukushima, K. and S. Miyake (1982). “Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition.” In: *Competition and Cooperation in Neural Nets*, pp. 267–285 (cit. on p. 9).

Fukushima, T. (2013). “Precise and fast computation of Lambert W-functions without transcendental function evaluations.” In: *Journal of Computational and Applied Mathematics* 244, pp. 77–89. ISSN: 0377-0427 (cit. on pp. 24, 140).

Fürnkranz, J., E. Hüllermeier, E. L. Mencía, and K. Brinker (2008). “Multilabel classification via calibrated label ranking.” In: *Machine Learning* 73.2, pp. 133–153 (cit. on pp. 112, 122).

Gao, W. and Z.-H. Zhou (2011). “On the Consistency of Multi-Label Learning.” In: *COLT*. Vol. 19, pp. 341–358 (cit. on pp. 28, 109).

Girshick, R. (2015). “Fast R-CNN.” In: *ICCV*, pp. 1440–1448 (cit. on p. 169).

Gong, Y., Y. Jia, T. Leung, A. Toshev, and S. Ioffe (2013). “Deep convolutional ranking for multilabel image annotation.” In: *arXiv:1312.4894* (cit. on pp. 26, 27).

Gong, Y., L. Wang, R. Guo, and S. Lazebnik (2014). “Multi-scale Orderless Pooling of Deep Convolutional Activation Features.” In: *ECCV* (cit. on p. 102).

Goodfellow, I., Y. Bengio, and A. Courville (2016). *Deep Learning*. http://www.deeplearningbook.org. MIT Press (cit. on p. 20).

Gorski, J., F. Pfeuffer, and K. Klamroth (2007). “Biconvex sets and optimization with biconvex functions: a survey and extensions.” In: *Math. Meth. Oper. Res.* 66.3, pp. 373–407 (cit. on p. 64).

Grauman, K. and T. Darrell (2005). “The pyramid match kernel: Discriminative classification with sets of image features.” In: *ICCV*. Vol. 2, pp. 1458–1465 (cit. on p. 18).

Gray, R. M. and D. L. Neuhoff (1998). “Quantization.” In: *IEEE Transactions on Information Theory* 44.6, pp. 2325–2383 (cit. on p. 20).

Guillaumin, M., T. Mensink, J. Verbeek, and C. Schmid (2009). “TagProp: Discriminative metric learning in nearest neighbor models for image auto-annotation.” In: *ICCV*, pp. 309–316 (cit. on p. 27).

Guo, Y. and D. Schuurmans (2011). “Adaptive large margin training for multilabel classification.” In: *AAAI*, pp. 374–379 (cit. on p. 122).

Gupta, M. R., S. Bengio, and J. Weston (2014). “Training Highly Multiclass Classifiers.” In: *JMLR* 15, pp. 1461–1492 (cit. on pp. 26, 79, 80, 87, 101).

Halevy, A., P. Norvig, and F. Pereira (2009). “The unreasonable effectiveness of data.” In: *IEEE Intelligent Systems* 24.2, pp. 8–12 (cit. on p. 5).

He, H. and E. A. Garcia (2009). “Learning from imbalanced data.” In: *IEEE Transactions on Knowledge and Data Engineering* 21.9, pp. 1263–1284 (cit. on p. 22).

He, K., X. Zhang, S. Ren, and J. Sun (2016). “Deep residual learning for image recognition.” In: *CVPR* (cit. on pp. 10, 20, 166).

Hein, M. (2016). *Machine Learning*. University Lecture. URL: http://www.ml.uni-saarland.de/Lectures/ML1516/Material/MachineLearningNotes.pdf (cit. on pp. 3, 4).

Hein, M. and M. Maier (2006). “Manifold denoising.” In: *NIPS*. Vol. 19, pp. 561–568 (cit. on p. 7).

Held, M., P. Wolfe, and H. P. Crowder (1974). “Validation of subgradient optimization.” In: *Mathematical programming* 6.1, pp. 62–88 (cit. on p. 23).

Hernández-Lobato, D., V. Sharmanska, K. Kersting, C. H. Lampert, and N. Quadrianto (2014). "Mind the nuisance: Gaussian process classification using privileged noise." In: *NIPS*, pp. 837–845 (cit. on p. 21).

Hinton, G., O. Vinyals, and J. Dean (2015). "Distilling the knowledge in a neural network." In: *arXiv preprint arXiv:1503.02531* (cit. on p. 21).

Hiriart-Urruty, J.-B. and C. Lemaréchal (2001). *Fundamentals of Convex Analysis.* Berlin: Springer (cit. on pp. 115, 116).

Hiriart-Urruty, J.-B. and C. Lemaréchal (1993). *Convex Analysis and Minimization Algorithms II: Advanced Theory and Bundle Methods.* Springer-Verlag (cit. on p. 183).

Householder, A. S. (1970). *The Numerical Treatment of a Single Nonlinear Equation.* McGraw-Hill (cit. on pp. 140, 145, 156).

Hsieh, C.-J., K.-W. Chang, C.-J. Lin, S. S. Keerthi, and S. Sundararajan (2008). "A dual coordinate descent method for large-scale linear SVM." In: *ICML*, pp. 408–415 (cit. on p. 141).

Hsu, C.-W. and C.-J. Lin (2002). "A comparison of methods for multiclass support vector machines." In: *Neural Networks* 13.2, pp. 415–425 (cit. on p. 158).

Hu, M.-K. (1962). "Visual pattern recognition by moment invariants." In: *IRE Transactions on Information Theory* 8.2, pp. 179–187 (cit. on p. 7).

Huang, J., A. J. Smola, A. Gretton, K. M. Borgwardt, B. Schölkopf, et al. (2007). "Correcting sample selection bias by unlabeled data." In: *NIPS* 19, p. 601 (cit. on p. 22).

Hubel, D. H. and T. N. Wiesel (1959). "Receptive fields of single neurones in the cat's striate cortex." In: *The Journal of Physiology* 148.3, pp. 574–591 (cit. on p. 9).

Hull, J. J. (1994). "A database for handwritten text recognition research." In: *PAMI* 16.5, pp. 550–554 (cit. on p. 18).

Jaakkola, T., D. Haussler, et al. (1999). "Exploiting generative models in discriminative classifiers." In: *NIPS* (cit. on pp. 19, 61).

Jaderberg, M., W. M. Czarnecki, S. Osindero, O. Vinyals, A. Graves, and K. Kavukcuoglu (2016). "Decoupled neural interfaces using synthetic gradients." In: *arXiv preprint arXiv:1608.05343* (cit. on p. 177).

Jarrett, K., K. Kavukcuoglu, Y. LeCun, et al. (2009). "What is the best multi-stage architecture for object recognition?" In: *ICCV*, pp. 2146–2153 (cit. on p. 20).

Jawanpuria, P., M. Lapin, M. Hein, and B. Schiele (2015). "Efficient Output Kernel Learning for Multiple Tasks." In: *Advances in Neural Information Processing Systems 29 (NIPS)* (cit. on pp. 3, 13, 76).

Jawanpuria, P. and J. S. Nath (2012). "A Convex Feature Learning Formulation for Latent Task Structure Discovery." In: *ICML*, pp. 137–144 (cit. on p. 24).

Jégou, H., M. Douze, C. Schmid, and P. Pérez (2010). "Aggregating local descriptors into a compact image representation." In: *CVPR*, pp. 3304–3311 (cit. on p. 20).

Jegou, H., F. Perronnin, M. Douze, J. Sánchez, P. Perez, and C. Schmid (2012). "Aggregating local image descriptors into compact codes." In: *PAMI* 34.9 (cit. on p. 71).

Jenatton, R., J.-Y. Audibert, and F. Bach (2011a). “Structured variable selection with sparsity-inducing norms.” In: *JMLR* 12.Oct, pp. 2777–2824 (cit. on p. 76).

Jenatton, R., J. Mairal, G. Obozinski, and F. Bach (2011b). “Proximal Methods for Hierarchical Sparse Coding.” In: *JMLR* 12, pp. 2297–2334 (cit. on p. 25).

Ji, Y., S. Sun, and Y. Lu (2012). “Multitask multiclass privileged information support vector machines.” In: *ICPR*, pp. 2323–2326 (cit. on pp. 21, 176).

Jia, Y., E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell (2014). “Caffe: Convolutional Architecture for Fast Feature Embedding.” In: *arXiv:1408.5093* (cit. on pp. 102, 161).

Joachims, T. (1998). “Text categorization with support vector machines: Learning with many relevant features.” In: *ECML*, pp. 137–142 (cit. on pp. 8, 18).

Joachims, T. (2002). “Optimizing search engines using clickthrough data.” In: *KDD*, pp. 133–142 (cit. on p. 26).

Joachims, T. (2005). “A support vector method for multivariate performance measures.” In: *ICML*, pp. 377–384 (cit. on pp. 80, 81, 86, 88, 101, 112).

Juneja, M., A. Vedaldi, C. Jawahar, and A. Zisserman (2013). “Blocks that shout: distinctive parts for scene classification.” In: *CVPR* (cit. on pp. 20, 61, 62, 102).

Kanehira, A. and T. Harada (2016). “Multi-label Ranking from Positive and Unlabeled Data.” In: *CVPR* (cit. on p. 27).

Kang, Z., K. Grauman, and F. Sha (2011). “Learning with whom to share in multi-task feature learning.” In: *ICML* (cit. on pp. 18, 24, 62, 68, 69).

Katakis, I., G. Tsoumakas, and I. Vlahavas (2008). “Multilabel text classification for automated tag suggestion.” In: *ECML PKDD Discovery Challenge* 75 (cit. on p. 162).

Kiwiel, K. C. (2007). “On linear-time algorithms for the continuous quadratic knapsack problem.” In: *JOTA* 134.3, pp. 549–554 (cit. on p. 23).

Kiwiel, K. C. (2008a). “Breakpoint searching algorithms for the continuous quadratic knapsack problem.” In: *Math. Prog.* 112.2, pp. 473–491 (cit. on pp. 23, 99).

Kiwiel, K. C. (2008b). “Variable Fixing Algorithms for the Continuous Quadratic Knapsack Problem.” In: *JOTA* 136.3, pp. 445–458 (cit. on pp. 23, 95, 98–101, 125, 138, 141, 151, 152).

Klimt, B. and Y. Yang (2004). “The Enron corpus: A new dataset for email classification research.” In: *ECML*, pp. 217–226 (cit. on p. 162).

Kocev, D., C. Vens, J. Struyf, and S. Džeroski (2007). “Ensembles of multi-objective decision trees.” In: *ECML*, pp. 624–631 (cit. on pp. 162, 163).

Kokkinos, I. (2016). “UberNet: Training a ‘Universal’ Convolutional Neural Network for Low-, Mid-, and High-Level Vision using Diverse Datasets and Limited Memory.” In: *arXiv preprint arXiv:1609.02132* (cit. on p. 25).

Koskela, M. and J. Laaksonen (2014). “Convolutional Network Features for Scene Recognition.” In: *ACM-MM*, pp. 1169–1172 (cit. on p. 102).

Koyejo, O. O., N. Natarajan, P. K. Ravikumar, and I. S. Dhillon (2015). “Consistent multilabel classification.” In: *NIPS*, pp. 3321–3329 (cit. on pp. 28, 112, 113, 164).

Krishnapuram, B., L. Carin, M. A. Figueiredo, and A. J. Hartemink (2005). "Sparse multinomial logistic regression: Fast algorithms and generalization bounds." In: *PAMI* 27.6, pp. 957–968 (cit. on p. 125).

Krizhevsky, A. (2009). *Learning multiple layers of features from tiny images.* Tech. rep. University of Toronto (cit. on p. 9).

Krizhevsky, A., I. Sutskever, and G. Hinton (2012). "ImageNet classification with deep convolutional neural networks." In: *NIPS* (cit. on pp. 8, 20).

Kuznetsov, V., M. Mohri, and U. Syed (2014). "Multi-class deep boosting." In: *NIPS*, pp. 2501–2509 (cit. on p. 28).

Lampert, C. H. (2009). "Kernel methods in computer vision." In: *Foundations and Trends® in Computer Graphics and Vision* 4.3, pp. 193–285 (cit. on p. 8).

Lampert, C. H., H. Nickisch, and S. Harmeling (2009). "Learning to detect unseen object classes by between-class attribute transfer." In: *CVPR*, pp. 951–958 (cit. on p. 9).

Lang, K. (1995). "Newsweeder: Learning to filter netnews." In: *ICML*, pp. 331–339 (cit. on p. 158).

Lapin, M., M. Hein, and B. Schiele (2014a). "Learning Using Privileged Information: SVM+ and Weighted SVM." In: *Neural Networks* 53 (cit. on p. 31).

Lapin, M., M. Hein, and B. Schiele (2015). "Top-k Multiclass SVM." In: *Advances in Neural Information Processing Systems 29 (NIPS)* (cit. on p. 79).

Lapin, M., M. Hein, and B. Schiele (2016a). "Analysis and Optimization of Loss Functions for Multiclass, Top-k, and Multilabel Classification." In: *arXiv preprint arXiv:1612.03663* (cit. on p. 107).

Lapin, M., M. Hein, and B. Schiele (2016b). "Loss Functions for Top-k Error: Analysis and Insights." In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 108).

Lapin, M., B. Schiele, and M. Hein (2014b). "Scalable Multitask Representation Learning for Scene Classification." In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on pp. 61, 63, 102).

Lauer, F. and G. Bloch (2008). "Incorporating prior knowledge in support vector machines for classification: A review." In: *Neurocomputing* 71.7, pp. 1578–1594 (cit. on pp. 31, 59).

Lazebnik, S., C. Schmid, and J. Ponce (2004). "Semi-local affine parts for object recognition." In: *BMVC*, pp. 779–788 (cit. on p. 18).

Lazebnik, S., C. Schmid, and J. Ponce (2006). "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories." In: *CVPR*. Vol. 2, pp. 2169–2178 (cit. on p. 18).

LeCun, Y., B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel (1989). "Backpropagation applied to handwritten zip code recognition." In: *Neural Computation* 1.4, pp. 541–551 (cit. on pp. 9, 17, 20).

LeCun, Y., L. Bottou, Y. Bengio, and P. Haffner (1998). "Gradient-based learning applied to document recognition." In: *Proceedings of the IEEE* 86.11, pp. 2278–2324 (cit. on pp. 17, 18).

LeCun, Y., F. J. Huang, and L. Bottou (2004). "Learning methods for generic object recognition with invariance to pose and lighting." In: *CVPR* (cit. on p. 20).

LeCun, Y., L. D. Jackel, L. Bottou, A. Brunot, C. Cortes, J. S. Denker, H. Drucker, I. Guyon, U. A. Müller, E. Sackinger, P. Simard, and V. Vapnik (1995). "Comparison of learning algorithms for handwritten digit recognition." In: *International Conference on Artificial Neural Networks*. Vol. 60, pp. 53–60 (cit. on p. 17).

Lee, C.-Y., S. Xie, P. Gallagher, Z. Zhang, and Z. Tu (2015a). "Deeply-Supervised Nets." In: *AISTATS* (cit. on p. 169).

Lee, D.-H., S. Zhang, A. Fischer, and Y. Bengio (2015b). "Difference target propagation." In: *Machine Learning and Knowledge Discovery in Databases*, pp. 498–515 (cit. on p. 177).

Lei, Y., U. Dogan, A. Binder, and M. Kloft (2015). "Multi-class SVMs: From tighter data-dependent generalization bounds to novel algorithms." In: *NIPS*, pp. 2035–2043 (cit. on pp. 28, 177).

Leibe, B. and B. Schiele (2003). "Interleaved object categorization and segmentation." In: *BMVC* (cit. on p. 6).

Lenc, K. and A. Vedaldi (2015). "R-CNN minus R." In: *BMVC* (cit. on p. 169).

Levy, L. and N. Wolf (2013). "The SVM-minus similarity score for video face recognition." In: *CVPR* (cit. on p. 21).

Lew, M. S., N. Sebe, C. Djeraba, and R. Jain (2006). "Content-based multimedia information retrieval: State of the art and challenges." In: *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)* 2.1, pp. 1–19 (cit. on p. 9).

Li, L.-J., R. Socher, and L. Fei-Fei (2009). "Towards total scene understanding: Classification, annotation and segmentation in an automatic framework." In: *CVPR*, pp. 2036–2043 (cit. on p. 9).

Li, J. and J. Z. Wang (2003). "Automatic linguistic indexing of pictures by a statistical modeling approach." In: *PAMI* 25.9, pp. 1075–1088 (cit. on p. 9).

Li, N., R. Jin, and Z.-H. Zhou (2014a). "Top rank optimization in linear time." In: *NIPS*, pp. 1502–1510 (cit. on pp. 26, 80, 88, 101).

Li, P., Q. Wu, and C. J. Burges (2007). "McRank: Learning to rank using multiple classification and gradient boosting." In: *NIPS* (cit. on p. 26).

Li, W., D. Dai, M. Tan, D. Xu, and L. Van Gool (2016). "Fast Algorithms for Linear and Kernel SVM+." In: *CVPR*, pp. 2258–2266 (cit. on pp. 22, 176).

Li, W., L. Niu, and D. Xu (2014b). "Exploiting privileged information from web data for image categorization." In: *ECCV*, pp. 437–452 (cit. on p. 21).

Liang, L. and V. Cherkassky (2008). "Connection between SVM+ and multi-task learning." In: *IJCNN*, pp. 2048–2054 (cit. on p. 21).

Liang, L., F. Cai, V. Cherkassky, et al. (2009). "Predictive learning with structured (grouped) data." In: *Neural Networks* 22.5-6, pp. 766–773 (cit. on p. 21).

Lin, C.-F. and S.-D. Wang (2002). "Fuzzy support vector machines." In: *IEEE Transactions on Neural Networks* 13.2, pp. 464–471 (cit. on p. 22).

Lin, T.-Y., M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick (2014). "Microsoft COCO: Common objects in context." In: *ECCV*, pp. 740–755 (cit. on pp. 109, 162).

Lindeberg, T. (2013). *Scale-space theory in computer vision.* Springer Science & Business Media (cit. on p. 7).

Liu, C.-L., K. Nakashima, H. Sako, and H. Fujisawa (2003). "Handwritten digit recognition: benchmarking of state-of-the-art techniques." In: *Pattern Recognition* 36.10, pp. 2271–2285 (cit. on p. 17).

Liu, J., S. Ji, and J. Ye (2009). "Multi-task feature learning via efficient $\ell_{2,1}$-norm minimization." In: *UAI*, pp. 339–348 (cit. on p. 23).

Liu, J. and J. Ye (2009). "Efficient Euclidean projections in linear time." In: *ICML*, pp. 657–664 (cit. on pp. 100, 151, 154).

Liu, J., W. Zhu, and P. Zhong (2013). "A new multi-class support vector algorithm based on privileged information." In: *Journal of Information and Computational Science* 2 (cit. on pp. 21, 176).

Liu, T.-Y. (2009). "Learning to rank for information retrieval." In: *Foundations and Trends in Information Retrieval* 3.3, pp. 225–331 (cit. on pp. 26, 112).

Liu, T. and D. Tao (2016). "Classification with noisy labels by importance reweighting." In: *PAMI* 38.3, pp. 447–461 (cit. on p. 27).

López, V., A. Fernández, S. García, V. Palade, and F. Herrera (2013). "An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics." In: *Information Sciences* 250, pp. 113–141 (cit. on p. 22).

Lopez-Paz, D., L. Bottou, B. Schölkopf, and V. Vapnik (2015). "Unifying distillation and privileged information." In: *arXiv preprint arXiv:1511.03643* (cit. on p. 21).

Lowe, D. G. (1999). "Object recognition from local scale-invariant features." In: *ICCV*, pp. 1150–1157 (cit. on p. 8).

Lowe, D. G. (2004). "Distinctive image features from scale-invariant keypoints." In: *IJCV* 60.2, pp. 91–110 (cit. on pp. 8, 18, 62, 70).

Luo, Y., D. Tao, C. Xu, D. Li, and C. Xu (2013). "Vector-valued multi-view semi-supervised learning for multi-label image classification." In: *AAAI*, pp. 647–653 (cit. on p. 27).

Madjarov, G., D. Kocev, D. Gjorgjevikj, and S. Džeroski (2012). "An extensive experimental comparison of methods for multi-label learning." In: *Pattern Recognition* 45.9, pp. 3084–3104 (cit. on pp. 27, 109, 112, 113, 122, 156, 162–165).

Magnanti, T. L. (1974). "Fenchel and Lagrange duality are equivalent." In: *Mathematical Programming* 7.1, pp. 253–258 (cit. on p. 183).

Mairal, J., F. Bach, and J. Ponce (2012). "Task-driven dictionary learning." In: *PAMI* 34.4, pp. 791–804 (cit. on pp. 19, 25, 62).

Mairal, J., R. Jenatton, F. R. Bach, and G. R. Obozinski (2010). "Network flow algorithms for structured sparsity." In: *NIPS*, pp. 1558–1566 (cit. on pp. 23, 138, 149).

Maji, S. and A. C. Berg (2009). "Max-margin additive classifiers for detection." In: *ICCV*, pp. 40–47 (cit. on p. 19).

Manning, C. D., P. Raghavan, and H. Schütze (2008). *Introduction to Information Retrieval.* Cambridge University Press (cit. on p. 112).

Margineantu, D. D. (2002). “Class probability estimation and cost-sensitive classification decisions.” In: *ECML*, pp. 270–281 (cit. on p. 22).

Marr, D. and H. K. Nishihara (1978). “Representation and recognition of the spatial organization of three-dimensional shapes.” In: *Proceedings of the Royal Society of London B: Biological Sciences* 200.1140, pp. 269–294 (cit. on p. 6).

Maurer, A. (2006). “Bounds for linear multi-task learning.” In: *JMLR* 7.Jan, pp. 117–139 (cit. on p. 24).

Maurer, A., M. Pontil, and B. Romera-Paredes (2013). “Sparse coding for multitask and transfer learning.” In: *ICML* (cit. on pp. 25, 62, 63, 69).

McAuley, J. J., A. Ramisa, and T. S. Caetano (2013). “Optimization of robust loss functions for weakly-labeled image taxonomies.” In: *IJCV* 104.3, pp. 343–361 (cit. on p. 27).

McCallum, A., K. Nigam, et al. (1998). “A comparison of event models for naive bayes text classification.” In: *AAAI-98 Workshop on Learning for Text Categorization*. Vol. 752, pp. 41–48 (cit. on p. 18).

McCulloch, W. S. and W. Pitts (1943). “A logical calculus of the ideas immanent in nervous activity.” In: *The Bulletin of Mathematical Biophysics* 5.4, pp. 115–133 (cit. on p. 9).

McFee, B. and G. R. Lanckriet (2010). “Metric learning to rank.” In: *ICML*, pp. 775–782 (cit. on p. 112).

Meer, P., R. Lenz, and S. Ramakrishna (1998). “Efficient invariant representations.” In: *IJCV* 26.2, pp. 137–152 (cit. on p. 7).

Mensink, T., J. Verbeek, F. Perronnin, and G. Csurka (2013). “Distance-based image classification: Generalizing to new classes at near-zero cost.” In: *PAMI* 35.11, pp. 2624–2637 (cit. on p. 27).

Michelot, C. (1986). “A finite algorithm for finding the projection of a point onto the canonical simplex of $\mathbb{R}^n$.” In: *JOTA* 50.1, pp. 195–200 (cit. on p. 23).

Mikolajczyk, K., B. Leibe, and B. Schiele (2005). “Local features for object class recognition.” In: *ICCV*. Vol. 2, pp. 1792–1799 (cit. on p. 8).

Misra, I., A. Shrivastava, A. Gupta, and M. Hebert (2016). “Cross-stitch networks for multi-task learning.” In: *CVPR*, pp. 3994–4003 (cit. on p. 25).

Moravec, H. (1988). *Mind children: The future of robot and human intelligence.* Harvard University Press (cit. on p. 2).

Motiian, S., M. Piccirilli, D. A. Adjeroh, and G. Doretto (2016). “Information bottleneck learning using privileged information for visual recognition.” In: *CVPR*, pp. 1496–1505 (cit. on p. 22).

Mundy, J. L. (2006). “Object Recognition in the Geometric Era: A Retrospective.” In: *Toward Category-Level Object Recognition*. Ed. by J. Ponce, M. Hebert, C. Schmid, and A. Zisserman. Springer Berlin Heidelberg, pp. 3–28 (cit. on p. 7).

Murase, H. and S. K. Nayar (1995). “Visual learning and recognition of 3-D objects from appearance.” In: *IJCV* 14.1, pp. 5–24 (cit. on p. 7).

Narasimhan, H., H. G. Ramaswamy, A. Saha, and S. Agarwal (2015). “Consistent Multiclass Algorithms for Complex Performance Measures.” In: *ICML*, pp. 2398–2407 (cit. on p. 28).

Narasimhan, H., R. Vaish, and S. Agarwal (2014). “On the statistical consistency of plug-in classifiers for non-decomposable performance measures.” In: *NIPS*, pp. 1493–1501 (cit. on p. 28).

Nesterov, Y. (2005). “Smooth minimization of non-smooth functions.” In: *Math. Prog.* 103.1, pp. 127–152 (cit. on p. 115).

Nesterov, Y. (2014). “Subgradient methods for huge-scale optimization problems.” In: *Mathematical Programming* 146.1-2, pp. 275–297 (cit. on p. 23).

Niblack, C. W., R. Barber, W. Equitz, M. D. Flickner, E. H. Glasman, D. Petkovic, P. Yanker, C. Faloutsos, and G. Taubin (1993). “QBIC project: querying images by content, using color, texture, and shape.” In: *IS&T/SPIE's Symposium on Electronic Imaging: Science and Technology*. International Society for Optics and Photonics, pp. 173–187 (cit. on p. 18).

Nilsback, M.-E. and A. Zisserman (2008). “Automated flower classification over a large number of classes.” In: *ICVGIP*, pp. 722–729 (cit. on p. 158).

Niu, L., Y. Shi, and J. Wu (2012). “Learning using privileged information with L-1 support vector machine.” In: *WI-IAT*. Vol. 3, pp. 10–14 (cit. on p. 21).

Nocedal, J. and S. J. Wright (2006). *Numerical Optimization*. Springer Science+ Business Media (cit. on pp. 55, 145, 158).

Nowozin, S. and C. H. Lampert (2011). “Structured learning and prediction in computer vision.” In: *Foundations and Trends in Computer Graphics and Vision* 6.3-4 (cit. on p. 82).

Ogryczak, W. and A. Tamir (2003). “Minimizing the sum of the k largest functions in linear time.” In: *Information Processing Letters* 85.3, pp. 117–122 (cit. on p. 84).

Oquab, M., L. Bottou, I. Laptev, and J. Sivic (2014). “Learning and transferring mid-level image representations using convolutional neural networks.” In: *CVPR*, pp. 1717–1724 (cit. on p. 20).

Pan, S. J. and Q. Yang (2010). “A survey on transfer learning.” In: *IEEE Transactions on Knowledge and Data Engineering* 22.10, pp. 1345–1359 (cit. on p. 25).

Parikh, N. and S. P. Boyd (2014). “Proximal Algorithms.” In: *Foundations and Trends in Optimization* 1.3, pp. 127–239 (cit. on pp. 23, 115).

Patriksson, M. (2008). “A survey on the continuous nonlinear resource allocation problem.” In: *EJOR* 185.1, pp. 1–46 (cit. on pp. 23, 94).

Patriksson, M. and C. Strömberg (2015). “Algorithms for the continuous nonlinear resource allocation problem – New implementations and numerical studies.” In: *EJOR* 243.3, pp. 703–722 (cit. on pp. 23, 94, 138).

Pechyony, D. and V. Vapnik (2010). “On the theory of learnining with privileged information.” In: *NIPS*, pp. 1894–1902 (cit. on pp. 21, 40).

Pechyony, D. and V. Vapnik (2011). “Fast Optimization Algorithms for Solving SVM+.” In: *Statistical Learning and Data Science*. Chapman & Hall (cit. on pp. 21, 35, 55, 176).

Pentina, A. and C. H. Lampert (2014). “A PAC-Bayesian Bound for Lifelong Learning.” In: *ICML* (cit. on p. 24).

Pentina, A., V. Sharmanska, and C. H. Lampert (2015). “Curriculum learning of multiple tasks.” In: *CVPR*, pp. 5492–5500 (cit. on p. 25).

Perronnin, F. and C. Dance (2007). “Fisher kernels on visual vocabularies for image categorization.” In: *CVPR*, pp. 1–8 (cit. on p. 19).

Perronnin, F., C. Dance, G. Csurka, and M. Bressan (2006). “Adapted vocabularies for generic visual categorization.” In: *ECCV*, pp. 464–475 (cit. on p. 19).

Perronnin, F. and D. Larlus (2015). “Fisher vectors meet neural networks: A hybrid classification architecture.” In: *CVPR*, pp. 3743–3752 (cit. on p. 19).

Perronnin, F., J. Sánchez, and T. Mensink (2010). “Improving the Fisher kernel for large-scale image classification.” In: *ECCV*, pp. 143–156 (cit. on pp. 19, 61).

Petersen, K. B., M. S. Pedersen, et al. (2008). “The matrix cookbook.” In: *Technical University of Denmark* 450, pp. 7–15 (cit. on pp. 90, 118).

Pham, T., T. Tran, and S. Venkatesh (2017). “One Size Fits Many: Column Bundle for Multi-X Learning.” In: *arXiv preprint arXiv:1702.07021* (cit. on p. 27).

Pires, B. Á. and C. Szepesvári (2016). “Multiclass Classification Calibration Functions.” In: *arXiv:1609.06385* (cit. on p. 28).

Pitts, W. and W. S. McCulloch (1947). “How we know universals: The perception of auditory and visual forms.” In: *The Bulletin of Mathematical Biophysics* 9.3, pp. 127–147 (cit. on p. 7).

Platt, J. C. (1999). “Fast training of support vector machines using sequential minimal optimization.” In: *Advances in Kernel Methods*, pp. 185–208 (cit. on p. 21).

Plessis, M. C. du, G. Niu, and M. Sugiyama (2014). “Analysis of learning from positive and unlabeled data.” In: *NIPS*, pp. 703–711 (cit. on p. 27).

Qu, Z., P. Richtárik, and T. Zhang (2015). “Quartz: Randomized dual coordinate ascent with arbitrary sampling.” In: *NIPS*, pp. 865–873 (cit. on pp. 23, 139).

Quattoni, A., X. Carreras, M. Collins, and T. Darrell (2009). “An efficient projection for $l_{1,\infty}$ regularization.” In: *ICML*, pp. 857–864 (cit. on p. 23).

Quattoni, A. and A. Torralba (2009). “Recognizing indoor scenes.” In: *CVPR* (cit. on pp. 101, 158).

Rakotomamonjy, A. (2012). “Sparse Support Vector Infinite Push.” In: *ICML*, pp. 1335–1342 (cit. on p. 26).

Ramaswamy, H. G. and S. Agarwal (2012). “Classification calibration dimension for general multiclass losses.” In: *NIPS*, pp. 2078–2086 (cit. on p. 28).

Ramaswamy, H. G., S. Agarwal, and A. Tewari (2013). “Convex calibrated surrogates for low-rank loss matrices with applications to subset ranking losses.” In: *NIPS*, pp. 1475–1483 (cit. on p. 28).

Razavian, A. S., H. Azizpour, J. Sullivan, and S. Carlsson (2014). “CNN Features off-the-shelf: an Astounding Baseline for Recognition.” In: *CVPRW, DeepVision workshop* (cit. on pp. 20, 102, 158).

Read, J., B. Pfahringer, G. Holmes, and E. Frank (2009). “Classifier chains for multi-label classification.” In: *ECML*, pp. 254–269 (cit. on pp. 112, 162, 164).

Reid, M. D. and R. C. Williamson (2010a). “Composite Binary Losses.” In: *JMLR* 11, pp. 2387–2422 (cit. on p. 136).

Reid, M. D. and R. C. Williamson (2010b). “Composite binary losses.” In: *JMLR* 11.Sep, pp. 2387–2422 (cit. on p. 28).

Ren, G., T. Hong, and Y. Park (2015). “Multi-Class SVM+MTL for the Prediction of Corporate Credit Rating with Structured Data.” In: *Asia Pacific Journal of Information Systems* 25.3, pp. 579–596 (cit. on pp. 22, 176).

Rennie, J. D. (2001). *Improving multi-class text classification with naive Bayes.* Tech. rep. Massachusetts Institute of Technology (cit. on p. 158).

Rezatofighi, S. H., A. Milan, E. Abbasnejad, A. Dick, I. Reid, et al. (2016). “DeepSetNet: Predicting Sets with Deep Neural Networks.” In: *arXiv preprint arXiv:1611.08998* (cit. on p. 27).

Richtárik, P. and M. Takác (2016). “Distributed coordinate descent method for learning with big data.” In: *Journal of Machine Learning Research* 17.75, pp. 1–25 (cit. on p. 23).

Rifkin, R. and A. Klautau (2004). “In defense of one-vs-all classification.” In: *JMLR* 5, pp. 101–141 (cit. on p. 159).

Rocha, A. and S. K. Goldenstein (2014). “Multiclass from binary: Expanding one-versus-all, one-versus-one and ecoc-based approaches.” In: *NNLS* 25.2, pp. 289–302 (cit. on p. 158).

Romera-Paredes, B., A. Argyriou, N. Berthouze, and M. Pontil (2012). “Exploiting unrelated tasks in multi-task learning.” In: *AISTATS* (cit. on pp. 24, 62).

Rosenblatt, F. (1958). “The perceptron: A probabilistic model for information storage and organization in the brain.” In: *Psychological Review* 65.6, p. 386 (cit. on p. 9).

Ross, S., J. Zhou, Y. Yue, D. Dey, and D. Bagnell (2013). “Learning Policies for Contextual Submodular Prediction.” In: *ICML*, pp. 1364–1372 (cit. on p. 27).

Roux, N. L., M. Schmidt, and F. R. Bach (2012). “A stochastic gradient method with an exponential convergence rate for finite training sets.” In: *NIPS*, pp. 2672–2680 (cit. on p. 25).

Rubinstein, R., A. M. Bruckstein, and M. Elad (2010). “Dictionaries for sparse representation modeling.” In: *Proceedings of the IEEE* 98.6, pp. 1045–1057 (cit. on p. 25).

Rubinstein, R., M. Zibulevsky, and M. Elad (2008). *Efficient implementation of the K-SVD algorithm using batch orthogonal matching pursuit.* Tech. rep. CS Technion (cit. on p. 25).

Rudin, C. (2009). “The P-Norm Push: A simple convex ranking algorithm that concentrates at the top of the list.” In: *JMLR* 10, pp. 2233–2271 (cit. on p. 26).

Rumelhart, D. E., G. E. Hinton, and R. J. Williams (1986). “Learning representations by back-propagating errors.” In: *Nature* 323, pp. 533–536 (cit. on p. 9).

Russakovsky, O., J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei (2015). “ImageNet Large Scale Visual Recognition Challenge.” In: *IJCV* 115.3, pp. 211–252 (cit. on pp. 20, 80, 81, 101, 106, 109, 158).

Sánchez, J., F. Perronnin, T. Mensink, and J. Verbeek (2013). “Image classification with the Fisher vector: theory and practice.” In: *IJCV*, pp. 1–24 (cit. on pp. 14, 19, 20, 61, 62, 69–72, 102).

Schiele, B. and J. L. Crowley (1996). “Object recognition using multidimensional receptive field histograms.” In: *ECCV*, pp. 610–619 (cit. on pp. 7, 18).

Schmid, C. and R. Mohr (1997). “Local grayvalue invariants for image retrieval.” In: *PAMI* 19.5, pp. 530–535 (cit. on pp. 7, 18).

Schmidhuber, J. (2015). “Deep learning in neural networks: An overview.” In: *Neural networks* 61, pp. 85–117 (cit. on p. 20).

Schmidt, M., D. Kim, and S. Sra (2011). “Projected Newton-type methods in machine learning.” In: *Optimization for Machine Learning* (cit. on p. 23).

Schölkopf, B., C. Burges, and V. Vapnik (1996). “Incorporating invariances in support vector learning machines.” In: *International Conference on Artificial Neural Networks*, pp. 47–52 (cit. on p. 17).

Schölkopf, B., R. Herbrich, and A. J. Smola (2001). “A generalized representer theorem.” In: *COLT*, pp. 416–426 (cit. on pp. 34, 36).

Schölkopf, B., P. Simard, A. J. Smola, and V. Vapnik (1998). “Prior knowledge in support vector kernels.” In: *NIPS*, pp. 640–646 (cit. on p. 21).

Schölkopf, B. and A. J. Smola (2002). *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. The MIT Press (cit. on pp. 8, 31, 36, 110).

Sermanet, P., S. Chintala, and Y. LeCun (2012). “Convolutional neural networks applied to house numbers digit classification.” In: *ICPR*, pp. 3288–3291 (cit. on p. 20).

Sermanet, P., D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun (2014). “OverFeat: Integrated recognition, localization and detection using convolutional networks.” In: *ICLR* (cit. on p. 20).

Shalev-Shwartz, S. and T. Zhang (2013a). “Accelerated Proximal Stochastic Dual Coordinate Ascent for Regularized Loss Minimization.” In: *arXiv:1309.2375*. arXiv: 1309.2375 [stat.ML] (cit. on p. 66).

Shalev-Shwartz, S. and S. Ben-David (2014). *Understanding machine learning: From theory to algorithms*. Cambridge University Press (cit. on p. 5).

Shalev-Shwartz, S. and Y. Singer (2006). “Efficient Learning of Label Ranking by Soft Projections onto Polyhedra.” In: *JMLR* 7, pp. 1567–1599 (cit. on pp. 23, 151, 154, 177, 178).

Shalev-Shwartz, S. and T. Zhang (2013b). “Stochastic dual coordinate ascent methods for regularized loss minimization.” In: *JMLR* 14, pp. 567–599 (cit. on pp. 12, 22, 23, 62, 64, 65, 67, 79, 80, 91, 106, 138, 171, 188).

Shalev-Shwartz, S. and T. Zhang (2014). “Accelerated proximal stochastic dual coordinate ascent for regularized loss minimization.” In: *Math. Prog.* Pp. 1–41 (cit. on pp. 23, 67, 82, 91, 109, 115, 116, 118, 131, 141, 143).

Sharan, L., R. Rosenholtz, and E. Adelson (2009). “Material perception: What can you see in a brief glance?” In: *Journal of Vision* 9.8, pp. 784–784 (cit. on p. 158).

Sharmanska, V., N. Quadrianto, and C. H. Lampert (2013). “Learning to Rank Using Privileged Information.” In: *ICCV* (cit. on p. 21).

Sharmanska, V., N. Quadrianto, and C. H. Lampert (2014). “Learning to transfer privileged information.” In: *arXiv preprint arXiv:1410.0389* (cit. on pp. 21, 175).

Shimodaira, H. (2000). “Improving predictive inference under covariate shift by weighting the log-likelihood function.” In: *Journal of Statistical Planning and Inference* 90.2, pp. 227–244 (cit. on p. 22).

Simard, P. Y., D. Steinkraus, J. C. Platt, et al. (2003). “Best Practices for Convolutional Neural Networks Applied to Visual Document Analysis.” In: *International Conference on Document Analysis and Recognition*. Vol. 3, pp. 958–963 (cit. on p. 18).

Simonyan, K. and A. Zisserman (2015). “Very deep convolutional networks for large-scale image recognition.” In: *ICLR* (cit. on pp. 20, 102, 158, 159, 166).

Sivic, J., B. C. Russell, A. A. Efros, A. Zisserman, and W. T. Freeman (2005). “Discovering objects and their location in images.” In: *ICCV*. Vol. 1, pp. 370–377 (cit. on p. 18).

Sivic, J. and A. Zisserman (2003). “Video Google: A text retrieval approach to object matching in videos.” In: *ICCV*. Vol. 2. 1470, pp. 1470–1477 (cit. on pp. 8, 18).

Slonim, N., N. Friedman, and N. Tishby (2006). “Multivariate information bottleneck.” In: *Neural Computation* 18.8, pp. 1739–1789 (cit. on p. 22).

Smeulders, A. W., M. Worring, S. Santini, A. Gupta, and R. Jain (2000). “Content-based image retrieval at the end of the early years.” In: *PAMI* 22.12, pp. 1349–1380 (cit. on p. 9).

Snoek, C. G., M. Worring, J. C. Van Gemert, J.-M. Geusebroek, and A. W. Smeulders (2006). “The challenge problem for automated detection of 101 semantic concepts in multimedia.” In: *ACM-MM* (cit. on p. 162).

Sra, S. (2011). “Fast projections onto $\ell_{1,q}$-norm balls for grouped feature selection.” In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 305–317 (cit. on p. 23).

Steinwart, I. (2005). “Consistency of support vector machines and other regularized kernel classifiers.” In: *IEEE Transactions on Information Theory* 51.1, pp. 128–142 (cit. on p. 28).

Steinwart, I. (2007). “How to compare different loss functions and their risks.” In: *Constructive Approximation* 26.2, pp. 225–287 (cit. on p. 28).

Su, Y. and F. Jurie (2012). “Improving image classification using semantic attributes.” In: *IJCV* 100.1, pp. 59–77 (cit. on p. 72).

Suen, C. Y., C. Nadal, R. Legault, T. A. Mai, and L. Lam (1992). “Computer recognition of unconstrained handwritten numerals.” In: *Proceedings of the IEEE* 80.7, pp. 1162–1180 (cit. on p. 17).

Sugiyama, M., S. Nakajima, H. Kashima, P. V. Buenau, and M. Kawanabe (2008). “Direct importance estimation with model selection and its application to covariate shift adaptation.” In: *NIPS*, pp. 1433–1440 (cit. on p. 22).

Sun, J. and J. Ponce (2013). “Learning discriminative part detectors for image classification and cosegmentation.” In: *ICCV*, pp. 3400–3407 (cit. on p. 102).

Sun, Y., M. S. Kamel, A. K. Wong, and Y. Wang (2007). “Cost-sensitive boosting for classification of imbalanced data.” In: *Pattern Recognition* 40.12, pp. 3358–3378 (cit. on p. 22).

Sun, Y., A. K. Wong, and M. S. Kamel (2009). “Classification of imbalanced data: A review.” In: *International Journal of Pattern Recognition and Artificial Intelligence* 23.04, pp. 687–719 (cit. on p. 22).

Swain, M. J. and D. H. Ballard (1991). “Color indexing.” In: *IJCV* 7.1, pp. 11–32 (cit. on pp. 7, 18).

Swersky, K., B. J. Frey, D. Tarlow, R. S. Zemel, and R. P. Adams (2012). "Probabilistic n-Choose-k Models for Classification and Ranking." In: *NIPS*, pp. 3050–3058 (cit. on pp. 27, 101, 102, 158, 159).

Sydorov, V., M. Sakurada, and C. H. Lampert (2014). "Deep Fisher Kernels – End to End Learning of the Fisher Kernel GMM Parameters." In: *CVPR* (cit. on p. 19).

Szegedy, C., W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich (2015). "Going deeper with convolutions." In: *CVPR*, pp. 1–9 (cit. on p. 20).

Szeliski, R. (2010). *Computer Vision: Algorithms and Applications*. Springer Science & Business Media (cit. on p. 6).

Takác, M., A. S. Bijral, P. Richtárik, and N. Srebro (2013). "Mini-Batch Primal and Dual Methods for SVMs." In: *ICML*, pp. 1022–1030 (cit. on p. 23).

Taylor, M., J. Guiver, S. Robertson, and T. Minka (2008). "SoftRank: optimizing non-smooth rank metrics." In: *WSDM*, pp. 77–86 (cit. on p. 26).

Tewari, A. and P. L. Bartlett (2007). "On the consistency of multiclass classification methods." In: *JMLR* 8, pp. 1007–1025 (cit. on pp. 28, 129, 133, 135).

Thrun, S. and T. M. Mitchell (1995). "Lifelong robot learning." In: *Robotics and autonomous systems* 15.1-2, pp. 25–46 (cit. on p. 25).

Thrun, S. and L. Pratt (2012). *Learning to learn*. Springer Science & Business Media (cit. on p. 24).

Trier, Ø. D., A. K. Jain, and T. Taxt (1996). "Feature extraction methods for character recognition – A survey." In: *Pattern Recognition* 29.4, pp. 641–662 (cit. on p. 17).

Trohidis, K., G. Tsoumakas, G. Kalliris, and I. P. Vlahavas (2008). "Multi-Label Classification of Music into Emotions." In: *ISMIR*. Vol. 8, pp. 325–330 (cit. on p. 162).

Tsochantaridis, I., T. Joachims, T. Hofmann, and Y. Altun (2005). "Large margin methods for structured and interdependent output variables." In: *JMLR*, pp. 1453–1484 (cit. on pp. 5, 26, 82, 114, 122).

Tsoumakas, G. and I. Katakis (2007). "Multi Label Classification: An Overview." In: *IJDWM* 3.3, pp. 1–13 (cit. on p. 122).

Tsoumakas, G., I. Katakis, and I. Vlahavas (2008). "Effective and efficient multilabel classification in domains with large number of labels." In: *ECML/PKDD Workshop on Mining Multidimensional Data* (cit. on pp. 162, 163).

Turk, M. A. and A. P. Pentland (1991). "Face recognition using eigenfaces." In: *CVPR*, pp. 586–591 (cit. on p. 7).

Ullman, S. et al. (1996). *High-level vision: Object recognition and visual cognition.* Vol. 2. MIT Press (cit. on p. 7).

Usunier, N., D. Buffoni, and P. Gallinari (2009). "Ranking with ordered weighted pairwise classification." In: *ICML*, pp. 1057–1064 (cit. on pp. 11, 12, 26, 80, 81, 86, 87, 114, 121, 177).

Van Erven, T., P. D. Grünwald, N. A. Mehta, M. D. Reid, and R. C. Williamson (2015). "Fast rates in statistical and online learning." In: *JMLR* 16, pp. 1793–1861 (cit. on p. 28).

Van Gemert, J. C., C. J. Veenman, A. W. Smeulders, and J.-M. Geusebroek (2010). "Visual word ambiguity." In: *PAMI* 32.7, pp. 1271–1283 (cit. on p. 19).

Vapnik, V. (2006). "Empirical Inference Science Afterword of 2006." In: *Estimation of Dependences Based on Empirical Data*. Springer (cit. on p. 21).

Vapnik, V. and R. Izmailov (2015). "Learning using privileged information: similarity control and knowledge transfer." In: *JMLR* 16, pp. 2023–2049 (cit. on p. 21).

Vapnik, V. and A. Vashist (2009). "A new learning paradigm: learning using privileged information." In: *Neural Networks* 22.5, pp. 544–557 (cit. on pp. 15, 18, 21, 29, 31, 32, 40, 48, 55, 56, 58, 59).

Vapnik, V., A. Vashist, and N. Pavlovitch (2009). "Learning using hidden information (learning with teacher)." In: *IJCNN*, pp. 3188–3195 (cit. on pp. 21, 36).

Veberič, D. (2012). "Lambert W function for applications in physics." In: *Computer Physics Communications* 183.12, pp. 2622–2628 (cit. on pp. 24, 140).

Vedaldi, A. and B. Fulkerson (2008). *VLFeat: An Open and Portable Library of Computer Vision Algorithms*. http://www.vlfeat.org/ (cit. on p. 70).

Vedaldi, A. and K. Lenc (2015). "MatConvNet – Convolutional Neural Networks for MATLAB." In: *ACM-MM* (cit. on pp. 159, 166).

Vedaldi, A. and A. Zisserman (2012). "Efficient additive kernels via explicit feature maps." In: *PAMI* 34.3, pp. 480–492 (cit. on pp. 19, 71).

Vembu, S. and T. Gärtner (2010). "Label ranking algorithms: A survey." In: *Preference learning*. Springer, pp. 45–64 (cit. on p. 26).

Vernet, E., M. D. Reid, and R. C. Williamson (2011). "Composite multiclass losses." In: *NIPS*, pp. 1224–1232 (cit. on p. 28).

Vinyals, O., S. Bengio, and M. Kudlur (2015). "Order matters: Sequence to sequence for sets." In: *arXiv preprint arXiv:1511.06391* (cit. on p. 27).

Vogel, J. and B. Schiele (2007). "Semantic modeling of natural scenes for content-based image retrieval." In: *IJCV* 72.2, pp. 133–157 (cit. on p. 9).

Wah, C., S. Branson, P. Welinder, P. Perona, and S. Belongie (2011). *The Caltech-UCSD Birds-200-2011 Dataset*. Tech. rep. California Institute of Technology (cit. on p. 158).

Wallraven, C., B. Caputo, and A. Graf (2003). "Recognition with Local Features: The Kernel Recipe." In: *ICCV*. Vol. 3, pp. 257–264 (cit. on p. 18).

Wan, L., M. Zeiler, S. Zhang, Y. L. Cun, and R. Fergus (2013). "Regularization of neural networks using dropconnect." In: *ICML*, pp. 1058–1066 (cit. on p. 18).

Wang, J., Y. Song, T. Leung, C. Rosenberg, J. Wang, J. Philbin, B. Chen, and Y. Wu (2014a). "Learning fine-grained image similarity with deep ranking." In: *CVPR*, pp. 1386–1393 (cit. on p. 26).

Wang, J., Y. Yang, J. Mao, Z. Huang, C. Huang, and W. Xu (2016a). "CNN-RNN: A Unified Framework for Multi-label Image Classification." In: *arXiv:1604.04573* (cit. on p. 27).

Wang, J., J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong (2010). "Locality-constrained linear coding for image classification." In: *CVPR*, pp. 3360–3367 (cit. on p. 19).

Wang, L., S. Guo, W. Huang, and Y. Qiao (2015a). "Places205-VGGNet Models for Scene Recognition." In: *CoRR* abs/1508.01667 (cit. on pp. 20, 158, 159).

Wang, M., C. Luo, R. Hong, J. Tang, and J. Feng (2016b). "Beyond Object Proposals: Random Crop Pooling for Multi-label Image Recognition." In: *IEEE Trans. Image Process.* 25.12 (cit. on p. 169).

Wang, M., B. Ni, X.-S. Hua, and T.-S. Chua (2012). "Assistive tagging: A survey of multimedia tagging with human-computer joint exploration." In: *ACM Computing Surveys (CSUR)* 44.4, p. 25 (cit. on p. 27).

Wang, X., D. Fouhey, and A. Gupta (2015b). "Designing deep networks for surface normal estimation." In: *CVPR*, pp. 539–547 (cit. on p. 25).

Wang, X. and Q. Ji (2016). "Object Recognition with Hidden Attributes." In: *IJCAI*, pp. 3498–3504 (cit. on p. 176).

Wang, Z. and Q. Ji (2015). "Classifier learning with hidden information." In: *CVPR*, pp. 4969–4977 (cit. on pp. 22, 175).

Wang, Z., X. Wang, and Q. Ji (2014b). "Learning with Hidden Information." In: *ICPR*, pp. 238–243 (cit. on p. 176).

Weber, M., M. Welling, and P. Perona (2000). "Unsupervised learning of models for recognition." In: *ECCV*, pp. 18–32 (cit. on p. 6).

Wei, Z. and M. Hoai (2016). "Region Ranking SVM for Image Classification." In: *CVPR* (cit. on pp. 166, 169).

Weston, J., S. Bengio, and N. Usunier (2011). "WSABIE: scaling up to large vocabulary image annotation." In: *IJCAI*, pp. 2764–2770 (cit. on pp. 26, 87).

Weston, J., C. Watkins, et al. (1999). "Support vector machines for multi-class pattern recognition." In: *European Symposium on Artificial Neural Networks.* Vol. 99, pp. 219–224 (cit. on p. 177).

Williamson, R. C. (2014). "The Geometry of Losses." In: *COLT*, pp. 1078–1108 (cit. on p. 28).

Williamson, R. C., E. Vernet, and M. D. Reid (2016). "Composite Multiclass Losses." In: *JMLR* 17.223, pp. 1–52 (cit. on p. 28).

Winn, J., A. Criminisi, and T. Minka (2005). "Object categorization by learned universal visual dictionary." In: *ICCV.* Vol. 2, pp. 1800–1807 (cit. on p. 19).

Wu, X. and R. Srihari (2004). "Incorporating prior knowledge with weighted margin support vector machines." In: *SIGKDD*, pp. 326–333 (cit. on pp. 22, 59).

Xia, F., T.-Y. Liu, J. Wang, W. Zhang, and H. Li (2008). "Listwise approach to learning to rank: theory and algorithm." In: *ICML*, pp. 1192–1199 (cit. on p. 26).

Xiao, J., J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba (2010). "SUN database: Large-scale scene recognition from abbey to zoo." In: *CVPR* (cit. on pp. 62, 69, 70, 72, 73, 75, 81, 101, 102, 158, 193).

Xu, C., T. Liu, D. Tao, and C. Xu (2016a). "Local rademacher complexity for multi-label learning." In: *IEEE Transactions on Image Processing* 25.3, pp. 1495–1507 (cit. on pp. 27, 28).

Xu, C., D. Tao, and C. Xu (2016b). "Robust Extreme Multi-label Learning." In: *SIGKDD*, pp. 1275–1284 (cit. on p. 27).

Xu, J. and H. Li (2007). “AdaRank: a boosting algorithm for information retrieval.” In: *SIGIR*, pp. 391–398 (cit. on p. 26).

Xu, X., A. Shimada, H. Nagahara, R.-i. Taniguchi, and L. He (2016c). “Image annotation with incomplete labelling by modelling image specific structured loss.” In: *IEEJ T ELECTR ELECTR* 11.1, pp. 73–82 (cit. on p. 27).

Xu, X., W. Li, and D. Xu (2015). “Distance metric learning using privileged information for face verification and person re-identification.” In: *IEEE Transactions on Neural Networks and Learning Systems* 26.12, pp. 3150–3162 (cit. on p. 21).

Yang, H. and I. Patras (2013). “Privileged information-based conditional regression forest for facial feature detection.” In: *Automatic Face and Gesture Recognition (FG), 10th IEEE International Conference and Workshops on*, pp. 1–6 (cit. on p. 21).

Yang, J., K. Yu, Y. Gong, and T. Huang (2009). “Linear spatial pyramid matching using sparse coding for image classification.” In: *CVPR*, pp. 1794–1801 (cit. on p. 19).

Yang, X., Q. Song, and A. Cao (2005). “Weighted support vector machine for data classification.” In: *IJCNN*, pp. 859–864 (cit. on p. 22).

Yang, Y. (1999). “An evaluation of statistical approaches to text categorization.” In: *Information retrieval* 1.1-2, pp. 69–90 (cit. on p. 112).

Ye, N., K. M. Chai, W. S. Lee, and H. L. Chieu (2012). “Optimizing F-measure: A Tale of Two Approaches.” In: *ICML*, pp. 289–296 (cit. on p. 28).

You, S., C. Xu, Y. Wang, C. Xu, and D. Tao (2017). “Privileged Multi-label Learning.” In: *arXiv preprint arXiv:1701.07194* (cit. on pp. 22, 175, 176).

Yu, A. W., H. Su, and L. Fei-Fei (2012). “Efficient euclidean projections onto the intersection of norm balls.” In: *ICML* (cit. on p. 23).

Yu, H.-F., F.-L. Huang, and C.-J. Lin (2011). “Dual coordinate descent methods for logistic regression and maximum entropy models.” In: *Machine Learning* 85.1-2, pp. 41–75 (cit. on pp. 24, 125).

Yue, Y., T. Finley, F. Radlinski, and T. Joachims (2007). “A support vector method for optimizing average precision.” In: *SIGIR* (cit. on p. 26).

Zadrozny, B., J. Langford, and N. Abe (2003). “Cost-sensitive learning by cost-proportionate example weighting.” In: *ICDM*, pp. 435–442 (cit. on p. 22).

Zeiler, M. D. and R. Fergus (2014). “Visualizing and understanding convolutional networks.” In: *ECCV*, pp. 818–833 (cit. on p. 20).

Zhang, M.-L. and L. Wu (2015). “Lift: Multi-label learning with label-specific features.” In: *PAMI* 37.1, pp. 107–120 (cit. on p. 27).

Zhang, M.-L. and Z.-H. Zhou (2014). “A review on multi-label learning algorithms.” In: *KDE* 26.8, pp. 1819–1837 (cit. on pp. 27, 112, 122).

Zhang, T. (2004). “Statistical behavior and consistency of classification methods based on convex risk minimization.” In: *Annals of Statistics* (cit. on p. 28).

Zhang, Z., P. Luo, C. C. Loy, and X. Tang (2014). “Facial landmark detection by deep multi-task learning.” In: *ECCV*, pp. 94–108 (cit. on p. 25).

Zhao, F., Y. Huang, L. Wang, and T. Tan (2015). “Deep semantic ranking based hashing for multi-label image retrieval.” In: *CVPR* (cit. on p. 26).

Zhao, P. and T. Zhang (2015). “Stochastic Optimization with Importance Sampling for Regularized Loss Minimization.” In: *ICML*, pp. 1–9 (cit. on p. 23).

Zhao, R.-W., J. Li, Y. Chen, J.-M. Liu, Y.-G. Jiang, and X. Xue (2016). “Regional Gating Neural Networks for Multi-label Image Classification.” In: *BMVC* (cit. on pp. 27, 166, 169, 170).

Zhong, W. and J. T. Kwok (2012). “Convex Multitask Learning with Flexible Task Clusters.” In: *ICML*, pp. 49–56 (cit. on p. 24).

Zhou, B., A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva (2014). “Learning Deep Features for Scene Recognition using Places Database.” In: *NIPS*, pp. 487–495 (cit. on pp. 11, 20, 80, 101, 102, 106, 109, 158).

Zhu, X., X. Li, and S. Zhang (2016). “Block-row sparse multiview multilabel learning for image classification.” In: *IEEE Transactions on Cybernetics* 46.2, pp. 450–461 (cit. on p. 27).

Zou, H. and T. Hastie (2005). “Regularization and variable selection via the elastic net.” In: *J. R. Stat. Soc.* 67.2, pp. 301–320 (cit. on p. 67).

Publications

The following publications are included in this thesis:

[1] Lapin, M., M. Hein, and B. Schiele (2014a). “Learning Using Privileged Information: SVM+ and Weighted SVM.” In: *Neural Networks* 53.

[2] Lapin, M., B. Schiele, and M. Hein (2014b). “Scalable Multitask Representation Learning for Scene Classification.” In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*

[3] Lapin, M., B. Schiele, and M. Hein (2014c). “Scalable Multitask Representation Learning for Scene Classification.” In: *Scene Understanding Workshop (SUNw).*

[4] Lapin, M., M. Hein, and B. Schiele (2015). “Top-k Multiclass SVM.” In: *Advances in Neural Information Processing Systems 29 (NIPS).*

[5] Lapin, M., M. Hein, and B. Schiele (2016a). “Analysis and Optimization of Loss Functions for Multiclass, Top-k, and Multilabel Classification.” In: *arXiv preprint arXiv:1612.03663.*

[6] Lapin, M., M. Hein, and B. Schiele (2016b). “Loss Functions for Top-k Error: Analysis and Insights.” In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*

Furthermore, the following publication was part of my PhD research, however, it is not covered in this thesis:

[7] Jawanpuria, P., M. Lapin, M. Hein, and B. Schiele (2015). “Efficient Output Kernel Learning for Multiple Tasks.” In: *Advances in Neural Information Processing Systems 29 (NIPS).*

Saarbrücken, June 27, 2017 Maksim Lapin