Advances in Empirical Risk Minimization for Image Analysis and Pattern Recognition

Matthew Blaschko

École Normale Supérieure de Cachan

Advances in Empirical Risk Minimization for Image Analysis and Pattern Recognition

Matthew B. Blaschko
Inria Saclay – Île-de-France
École Centrale Paris
Date de soutenance: 7 novembre 2014

Mémoire d'habilitation à diriger des recherches

Centre de Mathématiques et de Leurs Applications

Membres du jury:

Barbara	**CAPUTO**	Sapienza – Università di Roma	Examinateur
Frederic	**JURIE**	Université de Caen Basse-Normandie	Rapporteur
Nikos	**KOMODAKIS**	École Nationale des Ponts et Chaussées	Rapporteur
Neil	**LAWRENCE**	University of Sheffield	Examinateur
Nikos	**PARAGIOS**	École Centrale Paris	Examinateur
Bernt	**SCHIELE**	Max Planck Institut für Informatik	Rapporteur
Nicolas	**VAYATIS**	École Normale Supérieure de Cachan	Garant

Contents

For Elena & Thomas.

Scientific Publications Contributing to this Thesis

[BGB14] Wacha Bounliphone, Arthur Gretton, and Matthew B. Blaschko. A low variance consistent test of relative dependency. Technical report, 2014. arXiv:1406.3852.

[BKR13] Matthew B. Blaschko, Juho Kannala, and Esa Rahtu. Non maximal suppression in cascaded ranking models. In Joni-Kristian Kämäräinen and Markus Koskela, editors, *Image Analysis*, volume 7944 of *Lecture Notes in Computer Science*, pages 408–419. Springer, 2013.

[BL09] Matthew B. Blaschko and Christoph H. Lampert. Object localization with global and local context kernels. In *Proceedings of the British Machine Vision Conference*, pages 63.1–63.11. BMVA Press, 2009.

[BL12] Matthew B. Blaschko and Christoph H. Lampert. Guest editorial: Special issue on structured prediction and inference. *International Journal of Computer Vision*, 99(3):257–258, 2012.

[Bla11] Matthew B. Blaschko. Branch and bound strategies for non-maximal suppression in object detection. In Yuri Boykov, Fredrik Kahl, Victor Lempitsky, and Frank R. Schmidt, editors, *Energy Minimization Methods in Computer Vision and Pattern Recognition*, volume 6819 of *Lecture Notes in Computer Science*, pages 385–398. Springer, 2011.

[Bla13] Matthew B. Blaschko. A note on k-support norm regularized risk minimization. Technical report, 2013. arXiv:1303.6390.

[BMR14] Matthew B. Blaschko, Arpit Mittal, and Esa Rahtu. An $\mathcal{O}(n \log n)$ cutting plane algorithm for structured output ranking. In *Pattern Recognition*, Lecture Notes in Computer Science. Springer, 2014.

[BSB09] Matthew B. Blaschko, Jacquelyn Shelton, and Andreas Bartels. Augmenting feature-driven fMRI analyses: Semi-supervised learning and resting state activity. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 126–134. 2009.

[BSB+11] Matthew B. Blaschko, Jacquelyn A. Shelton, Andreas Bartels, Christoph H. Lampert, and Arthur Gretton. Semi-supervised kernel canonical correlation analysis with application to human fMRI. *Pattern Recognition Letters*, 32(11):1572–1583, 2011.

[BVZ10] Matthew B. Blaschko, Andrea Vedaldi, and Andrew Zisserman. Simultaneous object detection and ranking with weak supervision. In J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R.S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 235–243. 2010.

[BZG13] Matthew B. Blaschko, Wojciech Zaremba, and Arthur Gretton. Taxonomic prediction with tree-structured covariances. In Hendrik Blockeel, Kristian Kersting, Siegfried Nijssen, and Filip Železný, editors, *Machine Learning and Knowledge Discovery in Databases*, volume 8189 of *Lecture Notes in Computer Science*, pages 304–319. Springer, 2013.

[FB12] Alex Flint and Matthew B. Blaschko. Perceptron learning of SAT. In P. Bartlett, F.C.N. Pereira, C.J.C. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 2780–2788. 2012.

[GBS14] Mahsa Ghafarianzadeh, Matthew B. Blaschko, and Gabe Sibley. Unsupervised spatio-temporal segmentation with sparse spectral-clustering. In *Proceedings of the British Machine Vision Conference*. BMVA Press, 2014.

[GDB+13] Katerina Gkirtzou, Jean-François Deux, Guillaume Bassez, Aristeidis Sotiras, Alain Rahmouni, Thibault Varacca, Nikos

Paragios, and Matthew B. Blaschko. Sparse classification with MRI based markers for neuromuscular disease categorization. In Guorong Wu, Daoqiang Zhang, Dinggang Shen, Pingkun Yan, Kenji Suzuki, and Fei Wang, editors, *Machine Learning in Medical Imaging*, volume 8184 of *Lecture Notes in Computer Science*, pages 33–40. Springer, 2013.

[GHS+13a] Katerina Gkirtzou, Jean Honorio, Dimitris Samaras, Rita Goldstein, and Matthew B. Blaschko. fMRI analysis of cocaine addiction using k-support sparsity. In *Biomedical Imaging (ISBI), 2013 IEEE 10th International Symposium on*, pages 1078–1081, 2013.

[GHS+13b] Katerina Gkirtzou, Jean Honorio, Dimitris Samaras, Rita Goldstein, and Matthew B. Blaschko. fMRI analysis with sparse Weisfeiler-Lehman graph statistics. In Guorong Wu, Daoqiang Zhang, Dinggang Shen, Pingkun Yan, Kenji Suzuki, and Fei Wang, editors, *Machine Learning in Medical Imaging*, volume 8184 of *Lecture Notes in Computer Science*, pages 90–97. Springer, 2013.

[LB09] Christoph H. Lampert and Matthew B. Blaschko. Structured prediction by joint kernel support estimation. *Machine Learning*, 77(2-3):249–269, 2009.

[MBZT12] Arpit Mittal, Matthew B. Blaschko, Andrew Zisserman, and Philip H.S. Torr. Taxonomic multi-class prediction and person layout using efficient structured ranking. In Andrew Fitzgibbon, Svetlana Lazebnik, Pietro Perona, Yoichi Sato, and Cordelia Schmid, editors, *Computer Vision – ECCV 2012*, Lecture Notes in Computer Science, pages 245–258. Springer, 2012.

[MKB+14] Michail Misyrlis, Anna B. Konova, Matthew B. Blaschko, Jean Honorio, Nelly Alia-Klein, Rita Z. Goldstein, and Dimitris Samaras. Predicting cross-task behavioral variables from fMRI data using the k-support norm. In *Sparsity Techniques in Medical Imaging*. 2014.

[MKR+13] Subhransu Maji, Juho Kannala, Esa Rahtu, Matthew B. Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. Technical report, 2013. arXiv:1306.5151.

[OB14] José Ignacio Orlando and Matthew B. Blaschko. Learning fully-connected CRFs for blood vessel segmentation in retinal images. In *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2014*, Lecture Notes in Computer Science. Springer, 2014.

[RKB11] Esa Rahtu, Juho Kannala, and Matthew B. Blaschko. Learning a category independent object detection cascade. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 1052–1059, 2011.

[VBZ11] Andrea Vedaldi, Matthew B. Blaschko, and Andrew Zisserman. Learning equivariant structured output SVM regressors. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 959–966, 2011.

[VMT+14] Andrea Vedaldi, Siddhartha Mahendran, Stavros Tsogkas, Subhransu Maji, Ross Girshick, Juho Kannala, Esa Rahtu, Iasonas Kokkinos, Matthew B. Blaschko, David Weiss, Ben Taskar, Karen Simonyan, Naomi Saphra, and Sammy Mohamed. Understanding objects in detail with fine-grained attributes. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE International Conference on*. 2014.

[ZGB13] Wojciech Zaremba, Arthur Gretton, and Matthew B. Blaschko. B-tests: Low variance kernel two-sample tests. In *Advances in Neural Information Processing Systems 26*. 2013.

[ZKGB13] Wojciech Zaremba, M. Pawan Kumar, Alexandre Gramfort, and Matthew B. Blaschko. Learning from M/EEG data with variable brain activation delays. In James C. Gee, Sarang Joshi, Kilian M. Pohl, William M. Wells, and Lilla Zöllei, editors, *Information Processing in Medical Imaging*, volume 7917 of *Lecture Notes in Computer Science*, pages 414–425. Springer, 2013.

Chapter 1

Introduction

State of the art approaches in computer vision and medical image analysis are intricately tied to recent advances in machine learning. In order to overcome the inherent variability of visual data, statistical learning techniques are widely applied to extract semantic information from images, and are central to increases in accuracy of systems for image categorization, image retrieval, object detection, and scene analysis. Similarly, analogous learning techniques are beginning to drive advances in areas such as medical image processing, diagnosis, and functional brain image analysis.

A core property of learning algorithms is expressed through the "no free lunch" theorem of machine learning [172]: no given algorithm will have the best possible performance across all problem domains. It is therefore necessary to empirically determine the algorithms that give the best possible performance in specific applications. The problems considered in this manuscript have some commonalities, but also some differences. In computer vision, a common subdomain of problems comes from the desire to perform semantic image analysis, e.g. to specify the key objects in a scene and to possibly infer something about their spatial relationship or functional interactions [123, 27, 112, 113, 28, 30, 29, 25, 24, LB09, BVZ10, VBZ11, RKB11, MBZT12, BKR13, BL09, Bla11, BZG13, MKR$^+$13]. In functional brain imaging, we may be interested in identifying regions of the brain implicated in addiction [GHS$^+$13a, GHS$^+$13b, 92, 164, 165] or visual processing and memory [ZKGB13, BSB09, BSB$^+$11, 11, 14, 37]. These problems have inherently different goals, but are nevertheless unified by the common goal of analysis of images where we expect spatial regularity. It is in this context that the research described in this manuscript applies, specializes, and extends the state-of-the-art in machine learning for visual data.

The unifying framework for analysis used here is that of *empirical risk minimization* (ERM). ERM informs the organization of the work and its grouping into chapters. After a section overviewing the scientific foundations on which my contributions are based (Chapter 2), Chapter 3 details my work on the development of novel risk objectives for learning with visual data, largely in the structured output prediction framework [9]. Subsequently, Chapter 4 presents my advances in the subject of regularization methods for statistical learning. As my work frequently makes use of the structured output prediction setting, Representation and Inference is a central theme of tractable instantiations of learning systems, which is covered in Chapter 5.

1.1 Composition of the Thesis

This thesis is composed of a number of scientific publications preceded by an extended introduction. These articles are approximately partitioned by their primary methodological contribution, each of which forms a chapter in this thesis. Chapter 3, *Empirical Risk*, describes advancements in methodology for a component of empirical risk minimization primarily addressed in publications [LB09, BVZ10, RKB11, VBZ11, MBZT12, ZKGB13, BKR13]. Chapter 4, *Function Classes and Regularization*, is based primarily on [BSB09, BSB$^+$11, GHS$^+$13a, GHS$^+$13b, GDB$^+$13, Bla13]. Chapter 5, *Representation and Inference*, addresses a component of empirical risk minimization for which contributions are published in [BL09, Bla11, FB12, BZG13, MKR$^+$13].

Papers published after my doctorate are distinguished from papers published during or before my doctorate by citation style. Papers published after my doctorate make use of alphabetical citations and are listed at the beginning of the manuscript, while those published during or before my doctorate are numerical and are listed in the bibliography at the end of the manuscript.

Chapter 2

Scientific Foundations

In this chapter, I present the scientific foundations preceeding the contributions made in publications after my doctorate.

The unifying theme of this thesis is the application of empirical risk minimization (ERM) to problems in computer vision, medical imaging, and machine learning. A central concept in ERM is *risk* [157]. We use the following definition

$$\mathcal{R}(f) := \int \ell(f(x), y) dP(x, y) \tag{2.1}$$

where $\ell : \mathcal{Y} \times \mathcal{Y} \mapsto \mathbb{R}_+$ is a task specific loss function, $f : \mathcal{X} \mapsto \mathcal{Y}$ is the prediction function we are evaluating, $\mathcal{X}$ is the space of observed variables, and $\mathcal{Y}$ is the output space to be predicted. $P(x, y)$ is a distribution that governs the probability that x and y are jointly observed in a correctly labeled data sample. An optimal prediction function within a function class $\mathcal{F}$ is one that achieves

$$f^* := \arg\min_{f \in \mathcal{F}} \mathcal{R}(f). \tag{2.2}$$

Direct minimization of the risk is impossible in practice, as the true distribution P is generally unknown.

Empirical risk minimization substitutes an approximation to $\mathcal{R}$ based on a finite sample $\mathcal{S} := \{x_i, y_i\}_{1 \leq i \leq n}$ drawn from P. The most basic assumption is that the sample is independent and identically distributed (i.i.d.), though it is possible to account for deviations from this assumption [135]. Under the i.i.d. assumption, the *empirical risk* takes the form

$$\mathcal{R}(f) \approx \hat{\mathcal{R}}(f) := \frac{1}{n} \sum_{i=1}^{n} \ell(f(x_i), y_i). \tag{2.3}$$

Direct mimization of the empirical risk, i.e. computation of

$$\arg\min_{f\in\mathcal{F}} \hat{\mathcal{R}}(f), \tag{2.4}$$

can lead to overfitting to the training sample $\mathcal{S}$, necessitating the use of regularization to penalize complex functions [85]. Typical choices of regularizer are L_1 or L_2 norms of linear function classes [151, 153], or reproducting kernel Hilbert space (RKHS) regularization [140].

The function class $\mathcal{F}$ to be employed plays a key role in the expressiveness and accuracy of the optimal solution and tractability of optimization. An overly expressive function class carries inherent risk [156]. Certain function classes based on kernels are beneficial due to the existance of a *representer theorem* [101, 137, 50] leading to efficient optimization strategies based on convex optimization in the case of a finite data sample [43, 157, 140, 39, 32, 33]. Other function classes, such as multi-layer neural networks, do not lead to convex objective functions, but nevertheless have shown promising results in recent benchmark competitions [19, 41, 90, 136, 54, 48, 109]. In this thesis, we will exploit favorable properties of specific function classes where appropriate or necessary, but we note that many central principles of the learning theory extend to a wide array of function classes and optimization strategies. It is generally the case that the advances reported in this manuscript are widely applicable to a range of function classes beyond those which are demonstrated in the experiments.

The output space $\mathcal{Y}$ itself can play an important role. The most studied output spaces are those corresponding to scalar regression [117, 71, 60, 85], in which $\mathcal{Y} = \mathbb{R}$, or binary classification [57, 58, 157, 140, 51], in which $\mathcal{Y} = \{-1, +1\}$. In this latter setting, simple necessary and sufficient conditions on convex loss functions are known for statistical consistency [15]. In the context of computer vision and medical imaging, key application areas explored in this thesis, a promising paradigm is to learn prediction functions for complex and interdependent output spaces, a theme that is discussed in more detail in Section 2.3 and Chapter 3.

In the remainder of this chapter, we discuss specific methods and frameworks employed in the subsequent chapters.

2.1 Semi-supervised Learning

In semi-supervised learning, in addition to a training sample $\mathcal{S}$, we have an additional sample $\{x_j\}_{n+1\leq j\leq p}$ drawn from the marginalized distribution

$P(x) = \int P(x, y)dy$. A number of approaches for addressing this setting are described in [40].

In this thesis, we focus on the approach of semi-supervised Laplacian regularization [17]. Semi-supervised Laplacian regularization is a data dependent regularization approach that (i) non-parametrically estimates the manifold structure of a data sample as a finite graph, (ii) constructs a discrete version of the Laplace operator on the graph, and (iii) penalizes functions that have large variation along the manifold. A semi-supervised Laplacian regularized learning objective has the form

$$J(f) = \lambda\Omega(f) + \lambda_L\langle f, \hat{L}f\rangle + \hat{\mathcal{R}}(f) \tag{2.5}$$

where $\hat{L}$ is a graph Laplacian estimated from the supervised and unsupervised samples from $\mathcal{X}$, $\lambda_L \in \mathbb{R}_+$ is a scalar controling the degree of Laplacian regularization, Ω is another regularizer (e.g. $\Omega(f) = \|f\|^2$), and $\lambda \in \mathbb{R}_+$ a scalar controling the degree of regularization by Ω.

2.2 Sparsity Regularization

We consider a regularized risk objective

$$J(f) = \lambda\Omega(f) + \hat{\mathcal{R}}(f). \tag{2.6}$$

One may interpret such an objective as the Lagrangian of a constrained optimization problem

$$\arg\min_{f\in\mathcal{F}} \quad \hat{\mathcal{R}}(f) \tag{2.7}$$

$$\text{s.t.} \quad \Omega(f) = C. \tag{2.8}$$

From this interpretation, one can see that for every constant $C \in \mathbb{R}_+$ there exists a $\lambda \in \mathbb{R}_+$ such that an f that minimizes Equation (2.6) also minimizes the optimization problem in Equation (2.7). It is therefore the case that the solution to Equation (2.6) is at a point where a level set of $\hat{\mathcal{R}}(f)$ is equal to a level set of $\Omega(f)$ [21].

We will assume that $\mathcal{X} = \mathbb{R}^d$ and f is linearly parametrized as

$$f(x) = \langle w, x\rangle \tag{2.9}$$

for some $w \in \mathbb{R}^d$. Under these assumptions, one measure of the complexity of f is the number of non-zero coefficients of w. We may be tempted to set

the regularizer to the L_0 pseudo-norm

$$\Omega(f) = \|w\|_0 = \sum_{i=1}^{d} [w_i \neq 0]. \tag{2.10}$$

We use the Iverson bracket notation here [104]. If $\ell(f(x), y)$ is convex in its first argument, we have that $J(f)$ is convex in w if $\Omega(f)$ is convex in w, a property not satisfied by the L_0 pseudo-norm. An additional problem with the L_0 regularizer is that it does not penalize large values of $w_i \neq 0$. A standard approach in sparsity regularization is to take the norm whose unit ball is the convex relaxation of the L_0 unit ball, the L_1 norm. The L_1 regularizer is convex, penalizes large values of w_i, and leads to sparse solutions [151].

More general conditions for a convex regularizer to result in sparse solutions have been explored, such as that the gradient be bounded away from zero [22, 4]. That the optimum is at a point where a level set of $\hat{\mathcal{R}}(f)$ is equal to a level set of $\Omega(f)$ indicates that this is more likely to occur at a point of the level set where there is a discontinuity in the gradient, and we may construct regularizers that have discontinuities in the gradient around solutions with a large number of zero coefficients. Such observations lead to the development of *structured sparsity* regularizers, which prefer certain configurations of the non-zero components [95, 62, 96, 6, 5]. We explore the application of such regularizers to a number of learning objectives and application areas in Section 4.2.

2.3 Structured Output Prediction

A central question of empirical risk minimization is the output space $\mathcal{Y}$ of the prediction function f. In application to visual data, common output spaces may be an image segmentation, a taxonomic classification of an image, or a bounding box surrounding an object or region of interest. It is clear that if we are faced with the task of predicting an element in such a space, optimizing a regularized risk objective of the form in Equation (2.6) with a binary output will not optimize the true risk of the form in Equation (2.1) for the final prediction task.

Setting $\mathcal{Y}$ to a more general output space, such as a bounding box or segmentation, brings to question the form of $f : \mathcal{X} \mapsto \mathcal{Y}$. We may address this by specifying a *compatibility function* $g : \mathcal{X} \times \mathcal{Y} \mapsto \mathbb{R}$ and defining

$$f(x) := \arg\max_{y \in \mathcal{Y}} g(x, y). \tag{2.11}$$

If $g(x, y)$ has some tractable form, e.g.

$$g(x, y) := \langle w, \phi(x, y) \rangle \tag{2.12}$$

for some parameter vector $w \in \mathbb{R}^d$ and feature function $\phi : \mathcal{X} \times \mathcal{Y} \mapsto \mathbb{R}^d$, we may then develop a regularized risk framework analogous to that developed for binary classification or scalar regression [9]. We will develop here one such framework, the structured output support vector machine [149, 154, 155, 97].

2.3.1 Structured Output Support Vector Machine

The structured output support vector machine (SOSVM) makes the assumptions in Equations (2.11) and (2.12). Furthermore, the framework assumes access to a loss function $\Delta : \mathcal{Y} \times \mathcal{Y} \mapsto \mathbb{R}_+$ that measures the loss between the ground truth labeling for a given training sample and an incorrect prediction. As above, we assume a training set $\mathcal{S}$, and for each training input x_i we will use the notation $\tilde{y}_i \in \mathcal{Y} \setminus \{y_i\}$ to denote some incorrect prediction. The learning objective has two variants, margin rescaling

$$\min_{w \in \mathbb{R}^d, \xi \in \mathbb{R}^n_+} \frac{\lambda}{2} \|w\|^2 + \frac{1}{n} \sum_i \xi_i \tag{2.13}$$

$$\text{s.t. } \langle w, \phi(x_i, y_i) - \phi(x_i, \tilde{y}_i) \geq \Delta(y_i, \tilde{y}_i) - \xi_i \quad \forall i, \tilde{y}_i \in \mathcal{Y} \setminus \{y_i\} \tag{2.14}$$

and slack rescaling

$$\min_{w \in \mathbb{R}^d, \xi \in \mathbb{R}^n_+} \frac{\lambda}{2} \|w\|^2 + \frac{1}{n} \sum_i \xi_i \tag{2.15}$$

$$\text{s.t. } \langle w, \phi(x_i, y_i) - \phi(x_i, \tilde{y}_i) \geq 1 - \frac{\xi_i}{\Delta(y_i, \tilde{y}_i)} \quad \forall i, \tilde{y}_i \in \mathcal{Y} \setminus \{y_i\}. \tag{2.16}$$

The two variants are different piecewise linear convex upper bounds to a step function that pays a penalty of $\Delta(y_i, \tilde{y}_i)$ if $\langle w, \phi(x_i, y_i) \rangle < \langle w, \phi(x_i, \tilde{y}_i) \rangle$ and 0 otherwise (Figure 2.1).

Optimization of the SOSVM objectives is not straightforward, as the number of constraints (Equations (2.14) and (2.16)) is proportional to the size of the output space $|\mathcal{Y}|$. A general framework for optimizing these objectives is therefore based on a cutting plane approach [155, 97]. These approaches rely on an iterative method in which an active set of constraints is maintained. At each iteration in the algorithm, the objective is optimized using only the active constraints, and given a fixed w, the most violated

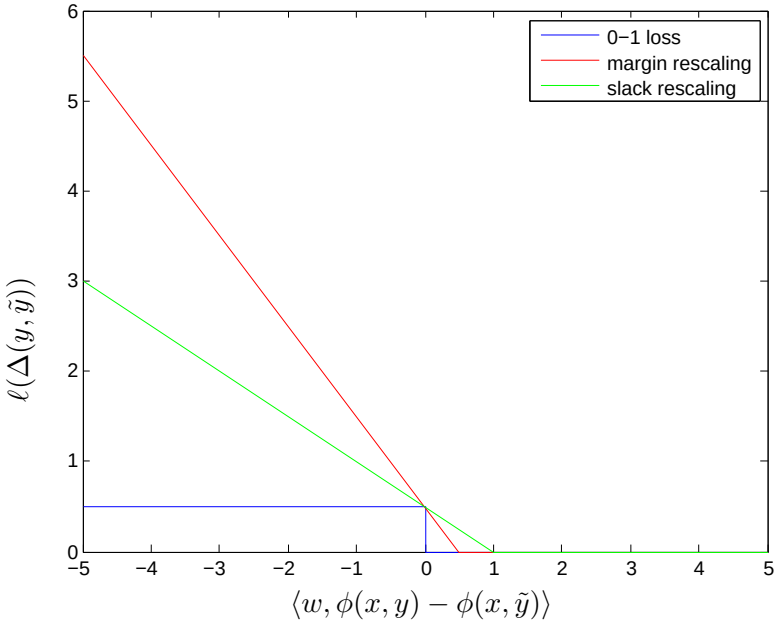


Figure 2.1: Margin rescaling and slack rescaling define different piecewise-linear convex upper bounds to a scaled step loss in two variants of a structured output support vector machine (SOSVM). In this example, we use a scaled 0-1 loss $\Delta(y, \tilde{y}) = \frac{1}{2}$.

constraint is determined and added to the set of active constraints. Finding the most violated constraint consists of the following two subproblems:

$$\arg \max_{\tilde{y}_i \in \mathcal{Y} \setminus \{y_i\}} \langle w, \phi(x_i, \tilde{y}_i) \rangle + \Delta(y_i, \tilde{y}_i) \tag{2.17}$$

for margin rescaling, and

$$\arg \max_{\tilde{y}_i \in \mathcal{Y} \setminus \{y_i\}} \left(\langle w, \phi(x_i, \tilde{y}_i) - \phi(x_i, y_i) \rangle + 1 \right) \Delta(y_i, \tilde{y}_i) \tag{2.18}$$

for slack rescaling. We refer to the optimization in Equation (2.11) as the inference problem, and the optimization in Equations (2.17) and (2.18) loss-augmented inference problems.

2.3.2 Inference and Loss-augmented Inference

There is typically a non-linear interaction between x and y in the joint feature function ϕ, meaning that the optimizations in Equations (2.11), (2.17) and (2.18) may be non-trivial or even NP-hard.

To illustrate the construction of a joint kernel map for a structured output problem, we may consider a family of log-linear Markov random-field models consisting of unary and pairwise energies [103]. We will denote unary energies as $f_u(x^i, y^i)$ for site i, and pairwise energies $f_p(y^i, y^j)$ for an edge in the set of edges defining the model $(i, j) \in \mathcal{E}$. We may define a vector representation of a site label in a discrete domain as an indicator vector $\phi_y(y^i) \in \{0, 1\}$, $\|\phi_y(y^i)\| = 1$. Similarly, we may define a vector representation of an observation at a specific site as $\phi_x(x^i) \in \mathcal{H}$ for some Hilbert space $\mathcal{H}$. Given these representations, we may define feature representations for log-linear unary and pairwise features as

$$\phi_u(x^i, y^i) = \phi_y(y^i) \otimes \phi_x(x^i), \tag{2.19}$$

$$\phi_p(y^i, y^j) = \phi_y(y^i) \otimes \phi_y(y^j), \tag{2.20}$$

respectively, where $\otimes$ represents the Kronecker product [125]. With these representations, our model assigns probabilities and energies

$$p(x, y) = \frac{1}{Z} \prod_i e^{f_u(x^i, y^i)} \prod_{(i,j) \in \mathcal{E}} e^{f_p(y^i, y^j)} \tag{2.21}$$

for some normalization constant, Z, and a corresponding compatability func-

tion (cf. Equation (2.12))

$$\begin{aligned} g(x,y) &= \log p(x,y) + \log Z & (2.22)\\ &= \sum_i f_u(x^i, y^i) + \sum_{(i,j)\in\mathcal{E}} f_p(y^i, y^j) & (2.23)\\ &= \langle w_u, \sum_i \phi_u(x^i, y^i)\rangle + \langle w_p, \sum_{(i,j)\in\mathcal{E}} \phi_p(y^i, y^j)\rangle. & (2.24) \end{aligned}$$

We may therefore represent the parameter vectors in the optimization problems described in Equations (2.13)-(2.16) as

$$w = \begin{pmatrix} w_u \\ w_p \end{pmatrix} \tag{2.25}$$

and the joint feature map

$$\phi(x,y) = \begin{pmatrix} \phi_u(x,y) \\ \phi_p(x,y) \end{pmatrix}. \tag{2.26}$$

It is clear from this construction that a seemingly simple assumption of parametrization by a joint feature map $\phi : \mathcal{X} \times \mathcal{Y} \mapsto \mathcal{H}$ leads to a very expressive class of functions, incorporating the rich literature of graphical models [105, 116, 99, 166, 61].

From this construction, we observe that the inference problems in Equations (2.11), (2.17) and (2.18) may have polynomial time algorithms, e.g. based on the forward-backward algorithm for tree-structured graphical models [8, 163]. The exact form of optimization, however, is dependent on the topology of the graph, and on the form of the pairwise constraints, e.g. whether they are submodular [124, 141, 65, 131, 106]. The assumption of submodular pairwise potentials in a discriminative learning setting such as a SOSVM means that the optimization problem must impose additional constraints on the model, such as was proposed by [3, 148]. In the absence of such constraints, the learned objective may not be in a tractable form, and may lead to NP-hard inference [46].

One must also consider modifications to the inference problem to incorporate the loss function as in Equations (2.17) and (2.18). For simplicity, we will presently assume that $\Delta(y, \tilde{y})$ is the Hamming loss [81, 149, 155]. In these cases, it is possible to modify the loss-augmented inference problems to incorporate the loss, resulting in a similar problem that may take advantage of existing inference algorithms, e.g. [155]. While the presentation here has used a general family of graphical models to describe the expressiveness

of the family of models, in Chapter 5, we extend this to cases where the form of a graphical model describing the problem is not immediately clear. We develop specialized inference algorithms for important computer vision problem settings that enable discriminative training of the form described in this section.

Chapter 3

Empirical Risk

This chapter addresses our contributions in the definition and training of models with novel risk formulations. Our contributions have included novel learning objectives and efficient optimization strategies [LB09, BVZ10, RKB11, VBZ11, MBZT12, BMR14]. We have applied novel ranking objectives in several works as strategies for learning cascaded object detectors [RKB11, BKR13]. Additionally, we have employed discriminative latent variable models, and have used such models for weakly supervised data and discriminative alignment of M/EEG recordings [BVZ10, ZKGB13]. This chapter focuses in detail on two contributions for novel structured prediction objectives: (i) joint kernel support estimation, and (ii) structured output ranking.

3.1 Joint Kernel Support Estimation

This section is based on [LB09].

Discriminative techniques, such as conditional random fields (CRFs) or structure aware maximum-margin techniques (maximum margin Markov networks (M^3N), structured output support vector machines (S-SVM)), are state-of-the-art in the prediction of structured data. However, to achieve good results these techniques require complete and reliable ground truth, which is not always available in realistic problems. Furthermore, training either CRFs or margin-based techniques is computationally costly, because the runtime of current training methods depends not only on the size of the training set but also on properties of the output space to which the training samples are assigned.

We propose an alternative model for structured output prediction, Joint Kernel Support Estimation (JKSE), which is rather generative in nature as

it relies on estimating the joint probability density of samples and labels in the training set. This makes it tolerant against incomplete or incorrect labels and also opens the possibility of learning in situations where more than one output label can be considered correct. At the same time, we avoid typical problems of generative models as we do not attempt to learn the full joint probability distribution, but we model only its support in a joint reproducing kernel Hilbert space. As a consequence, JKSE training is possible by an adaption of the classical one-class SVM procedure. The resulting optimization problem is convex and efficiently solvable even with tens of thousands of training examples. A particular advantage of JKSE is that the training speed depends only on the size of the training set, and not on the total size of the label space. No inference step during training is required nor do we have calculate a partition function. Experiments on realistic data show that, for suitable kernel functions, our method works efficiently and robustly in situations that discriminative techniques have problems with or that are computationally infeasible for them.

We follow the common language of the field [9] and treat structured prediction in probabilistic terms as a MAP-prediction problem. Let $\mathcal{X}$ be the space of observations and $\mathcal{Y}$ be the space of possible labels. Note that for our setup it is not required that $\mathcal{X}$ or $\mathcal{Y}$ decompose into smaller entities like nodes or edges in a graph. We assume that sample–label pairs (x, y) follow a joint-probability density $p(x, y)$, and that a set of i.i.d. samples (x_i, y_i) is for $i = 1, \ldots, n$ is available for training. The task is to learn a mapping $f : \mathcal{X} \to \mathcal{Y}$ that minimizes the expected loss in a classification sense, i.e. for a new sample $x \in \mathcal{X}$, we have to determine the label $y \in \mathcal{Y}$ that maximizes the posterior probability $p(y|x)$.

It is well known that, mathematically, the discriminative approach of directly modelling $p(y|x)$ and the generative approach of modelling $p(x, y)$ are equivalent, because Bayes rule allows one to use either quantity for optimal prediction:

$$\arg\max_{y \in \mathcal{Y}} p(y|x) = \arg\max_{y \in \mathcal{Y}} p(x, y). \tag{3.1}$$

In joint-kernel support estimation we follow the generative path and model an expression for $p(x, y)$ from the given training data. However, density estimation in high dimensional spaces is notoriously difficult. We therefore simplify the problem by assuming that the posterior probability is the feature space is *distinctive*, i.e. $p(y|x) \gg 0$ for correct predictions y and $p(y|x) \approx 0$ for incorrect y. Consequently, $p(x, y) \gg 0$ only if y is a correct label for x, and it suffices to estimate the *support* of $p(x, y)$ instead

of the full density. Afterwards, we can still use $f(x) := \arg\max p(x, y)$ for prediction. Note that we do not require $p(y|x)$ to be unimodal. Different $y \in \mathcal{Y}$ could be "correct" predictions for $x \in \mathcal{X}$, as occurs quite commonly in realistic structured prediction tasks.

The generative setup limits JKSE's ability to *extrapolate*, because $p(x, y) = p(y|x)p(x)$ and for a test sample $x \in \mathcal{X}$ this vanishes not only for wrong predictions y but also if x lies an area of $\mathcal{X}$ that has not been observed during training. On the other hand, the setup also has certain advantages, e.g., it opens the possibility to perform adaptive and online learning, building a smaller model of $p(x, y)$ first and later extending the label set without having to retrain for the previous labels.

3.1.1 Representation

Like most other structured prediction methods, we model probabilities by log-linear models [9]. Since our central quantity of interest is $p(x, y)$, we set

$$p(x, y) \equiv \frac{1}{Z} \exp(\, \langle w, \phi(x, y) \rangle \,) \tag{3.2}$$

where $Z \equiv \sum_{x,y} \exp(\, \langle w, \phi(x, y) \rangle \,)$ is a normalization constant (or *partition function*). Since our model is generative, *Z does not depend on y* and we can essentially ignore it during training as well as during inference. Consequently, the prediction step reduces to

$$f(x) \equiv \arg\max_{y \in \mathcal{Y}} \langle w, \phi(x, y) \rangle. \tag{3.3}$$

In the following, we will assume access to some form of inference algorithm for calculating the arg max in (3.3) or a suitable approximation to it. We do not require a method to calculate or approximate Z at any time.

The feature map $\phi(x, y)$ in Equation (3.2) is completely generic. While in many situations, such as sequence labeling or image segmentation, it is natural to form $\phi(x, y)$ by a concatenation or summation of per-site properties and neighborhood features, we do not require such a decomposition for our consideration. In fact, $\phi(x, y)$ does not have to be an explicit mapping at all, and in the following we will only study the case where it is induced by a suitable positive definite joint kernel function $k : (\mathcal{X} \times \mathcal{Y}) \times (\mathcal{X} \times \mathcal{Y}) \to \mathbb{R}$. The case of an explicitly known $\phi(x, y)$ can be included in this framework by setting $k(\, (x, y), (x', y') \,) \equiv \langle \phi(x, y), \phi(x', y') \rangle$.

3.1.2 Parameter Learning

Assuming a fixed kernel or feature map, learning consists only of finding a suitable weight vector w such that the right hand side of (3.2) reflects $p(x, y)$ over the training set. Since we are only interested in the support of $p(x, y)$, we can use a one-class support vector machine (OC-SVM) for this purpose, see Section 3.1.3 for a review of this technique. The result of OC-SVM training is a representation of $p(x, y)$ as a linear combination of kernel evaluations with the training samples. Thus, the JKSE prediction function can be written as

$$f(x) = \arg\max_{y \in \mathcal{Y}} \sum_{i=1}^{n} \alpha_i \, k(\,(x, y), (x_i, y_i)\,). \tag{3.4}$$

Note that this expansion is generally sparse, i.e. most α_i have the value 0.

In its training procedure, JKSE reduces the basically generative task of learning $p(x, y)$ to a maximum margin learning problem of estimating the support of $p(x, y)$.

The JKSE training procedure works for arbitrary Mercer kernels [157] and the resulting optimization problem is convex. Furthermore, only the matrix of joint-kernel values is required, and thus the training time depends only on the size of the training set, not on the structure of the output space. This is in contrast to many other techniques for structured prediction.

3.1.3 One-Class SVM Training

The one-class support vector machine (OC-SVM) was originally introduced to robustly estimate the support and quantiles of probability densities in high dimensional spaces [138]. In the case of JKSE, we replace the original samples by sample-label pairs and consider them as embedded into the latent Hilbert space $\mathcal{H}$ that is induced by the joint kernel function $k(\,(x, y), (x_i, y_i)\,)$.

The OC-SVM works by estimating a hyperplane in $\mathcal{H}$ that best separates the training samples from the origin, except for a set of *outliers* which are determined implicitly by the training procedure. A parameter $\nu \in (0, 1]$ acts as an upper bound to the percentage of outliers, i.e. the larger ν, the more freedom the method has to disregard any of the training samples. By this, ν simultaneously acts as a regularization parameter.[1] JKSE training

[1] For support estimation, it is generally more intuitive to study the problem of finding the ball of smallest radius in the feature space that encloses all training points except for the outliers. Both concepts are in fact equivalent for the common class of kernels

using OC-SVM can be written in primal form as the quadratic optimization problem

$$\min_{w\in\mathcal{H},\xi_i\in\mathbb{R}^+,\rho\in\mathbb{R}} \frac{1}{2}\|w\|^2 + \frac{1}{\nu n}\sum_i \xi_i - \rho \tag{3.5}$$

subject to

$$\langle\, w, \phi(x_i, y_i)\,\rangle_{\mathcal{H}} \geq \rho - \xi_i \quad \text{for } i = 1, \ldots, n, \tag{3.6}$$

where $\langle .\,, .\rangle_{\mathcal{H}}$ denotes the scalar product in the Hilbert space $\mathcal{H}$. To actually solve the minimization (3.5), one applies the representer theorem [140]. Consequently, all references to $\langle .\,, .\rangle_{\mathcal{H}}$ and ϕ disappear and kernel evaluations are required. We solve the dual problem:

$$\min_{\alpha} \sum_{ij} \alpha_i \alpha_j k((x_i, y_i), (x_j, y_j)) \tag{3.7}$$

subject to

$$0 \leq \alpha_i \leq \frac{1}{\nu n}, \quad \sum_i \alpha_i = 1 \quad \text{for } i = 1, \ldots, n. \tag{3.8}$$

This allows us to reuse existing OC-SVM implementations for JKSE learning: we provide the algorithm with the joint kernel matrix between pairs (x_i, y_i) instead of the ordinary kernel matrix measuring similarity between samples x_i. OC-SVM training will then learn coefficients, α_i, that can directly be used in the context of Equation (3.4).

Note that in the training procedure, only comparisons between training sample pairs are required. At no time do we have to evaluate a function over the space of all possible target labels, which would make learning dependent not only on the size of the training set, but also on the label space.

The many theoretical studies of OC-SVMs in the machine learning literature immediately carry over to the training of JKSE. One interesting aspect of this is that —at least for suitable kernels functions— one can prove consistency results for the approximation of $p(x, y)$, see [160]. Note that because of the additional arg max operation, this, however, does not imply consistency of JKSE's prediction step.

---

function where every sample has the same length in feature space, e.g. Gaussian kernels and generalizations (see [150]).

3.1.4 Large Scale Training

Disregarding the time to calculate the kernel matrix, training JKSE is identical to training a *OC-SVM*. In principle this requires the same computational effort as training an ordinary two-class SVM and one should therefore expect that both methods can be applied to problem of similar size and complexity. However, because OC-SVMs are less popular for pattern recognition tasks, significantly less effort has been spend on developing fast training routines and on optimizing the implementations. Existing packages such as `libSVM` can handle thousands of examples, but not tens of thousands. In the following, we therefore discuss possibilities to implement JKSE independently of the existing OC-SVM packages: by reformulating it as a *binary classification problem* and employing fast *stochastic online training techniques*.

In their original analysis, Schölkopf et al. showed that, for datasets that can be linearly separated from the origin and *in the case without slack variables*, the weight vector found by optimizing the OC-SVM problem (3.5) is equivalent to solving the optimization problem of a regular support vector machine for binary classification with only positive training examples, but additionally imposing that the hyperplane found has to pass through the origin [138, 21]. Alternatively, one can allow arbitrary hyperplanes, but add a mirrored copy of the training set with a negative training label. This will learn the same weight vector as the previous construction and the hyperplane in a symmetric problem automatically passes through the origin.

Linear separability can always be enforced by the right choice of kernel, e.g. any kernel with non-negative values. When generalizing the result to the case of slack variables, one and two-class training are still equivalent in the sense that for each ν, a corresponding regularization parameter C exists that results in the same hyperplane. However, the relationship between ν and C becomes non-explicit [138, 140].

However, we are not interested in equivalence for a specific ν, but intend to perform model selection over this parameter anyway. We can therefore make use of the equivalence result and train a two-class SVM with model selection over C instead.

3.1.5 Stochastic Online Training

With the availability of larger and larger data collections, machine learning research has focussed increasingly on the creation of methods that not only achieve high prediction accuracy, but that can also be trained efficiently on large datasets, see e.g. [33]. As a result, several fast learning algorithms for

support vector machines have been developed, many of them limited to linear kernels, e.g. [94, 98, 122, 142], but some also applicable to arbitrary Mercer kernels ([31, 32]). Typically, these method rely on ideas from online learning, such as *stochastic gradient descent* (see e.g. [20]). Using such approximate larger scale SVM learners in combination with the reformulation of OC-SVM as a regular SVM, we can train JKSE with dataset of tens of thousand of examples or more. For linear kernels, even millions of examples are in reach.

3.1.6 Experimental Evaluation

We evaluate the performance of JKSE on a real-life task from *computer vision*. The setup allows us to demonstrate the two major claims that we made: robustness of JKSE against high amounts of label noise, and the computational efficiencies of training without iterated inference. We compare JKSE to a structured regression method based on S-SVM that has been shown to achieve state-of-the-art performance for similar object localizationon tasks [29] .

3.1.7 Object Localization in Images

We adopt the setup from [29] to perform object localization by structured prediction: the observations are natural images, and the labels are the coordinates of the bounding box of an object. If an image contains more than one object, any of their bounding boxes is considered a correct label. For the dataset we use the `UIUCcars` set,[2] choosing the *multiscale* part for training and the *singlescale* part for testing. This leave us with 108 images showing 139 cars for training, which is close to the upper limit that the S-SVM in this situation can handle in reasonable time. The test set consists of 170 images containing 200 cars. Example images of the dataset are shown in Figure 3.1.

An additional part of the dataset consists of 1050 smaller images which were pre-cropped to show either a car or background region. This makes them useless for the task of object localization, but as we will see later, we can make use of them for fast model selection. All images are represented by densely sampled image SURF image descriptors [16], which are quantized into 1000 visual word clusters, see [113] for details. As a joint-kernel function we choose the *localization kernel* from [29]: given two sample-label pairs (x, y) and (x', y'), it forms a 4-level spatial pyramid bag-of-words histograms of those feature points within x and x' that fall into the box regions y and y'

[2] `http://l2r.cs.uiuc.edu/~cogcomp/Data/Car/`

Figure 3.1: Examples images of the `UIUCcars` dataset for object localization. The task is to predict tight bounding boxes for the car objects. Images are of different sizes and can contain more than one car, i.e. more than one output label can be correct.

respectively. The resulting histograms are combined into a kernel values by either a linear scalar product or a χ^2-kernel. The former has the advantage that a very fast MAP-inference is possible using an integral-image trick. This makes exact S-SVM training feasible. The latter is generally accepted as a better kernel for computer vision tasks, but MAP-inference has to be done by exhaustively scanning over all image locations and is therefore computationally very costly.

We are interested in the performance of JKSE and S-SVM for training sets with different amounts of label noise. To simulate this, we artificially introduce label errors into the dataset by swapping bounding box coordinates between different training images. While this preserves the overall label statistics, the image contents at the positions given by the swapped labels will not necessarily show cars and therefore obstruct the learning process. The percentage of swapped labels is a free parameter, r, that we vary between 0% (perfect labels) and 100% (random labels).

3.1.8 Model Selection

Training JKSE in the situation described takes only a few seconds. Evaluation takes also in the order of seconds for the linear kernel function, whereas for the χ^2 kernel it takes a several minutes per image. This is because within each image there are tens of thousands of possible object locations, and for each, a high-dimensional non-sparse histogram has to be formed and the classifier evaluated.

The S-SVM has identical evaluation time, as it solves the same inference problem. However, training requires iterative solution of the MAP estimate, each corresponding to a full evaluation of the prediction function over many images. This procedure is only feasible for the linear kernel, and even with

the fast integral image trick, the total training duration was approximately 5 hours. Within this time, on average close to 3,500 calls to the MAP-estimate were performed accounting for 97% of total training time.

For method with very long training time, as the S-SVM in our case, *model selection* is always a difficult issue. Except in special cases, it is not practical to perform full cross-validation runs even for a single value like the regularization parameter C. We therefore rely on a simplified criterion: we train S-SVM using values $C \in \{10^{-3}, 10^{-2}, \ldots, 10^{2}\}$. For testing, we use the weight vector of the value that achieves the highest area under curve when used as a classifier on the set of small additional images that we left out during training because they were pre-cropped. In order to facilitate comparability, we follow the same procedure for JKSE to select $\nu \in \{0.05, 0.1, \ldots, 1.0\}$. Note, however, that JKSE is in fact fast enough to perform full cross-validation, and this could be expected to improve the localization performance to a certain extent.

3.1.9 Results

The localization performance of S-SVM and JKSE are measured by precision-recall curves which are depicted in Figures 3.2 and 3.3. The former shows the results of S-SVM and JKSE with linear kernels. As one can see, S-SVM achieves higher precision and recall than JKSE for noiseless data as well as when 10% and 30% of labels are scrambled. One can assume that it is the S-SVM's Tikhonov regularization that successfully compensates the disturbance introduced by the label errors. However, when the label error rate reaches 50% or more, S-SVM performance takes a huge dive, and at 90% label errors, performance is basically random. A notable anomaly is that the 50%-curve lies below the 70%-curve. As there is no fundamental reason for this, we believe it to be an artifact of the model selection procedure. In fact, S-SVM in this setup has proven rather sensitive to a good choice of C.

In contrast, JKSE with a linear kernel starts from a lower precision level, but its performance decreases more continuously when the amount of label errors increase. Even for 90% label noise, JKSE achieves non-trivial localization performance. We attribute this somewhat surprising behavior to a successful application of the ν-formalism, as $\nu = 0.95$ was chosen at this level, thereby correctly treating a large amount of the training data as outliers. Furthermore, our analysis showed that JKSE is rather insensitive to the choice of ν.

Figure 3.3 shows results for JKSE with the χ^2 kernel function. In comparison with Figure 3.2 one can clearly see that JKSE's localization accuracy

	$r=0.0$	$r=0.1$	$r=0.3$	$r=0.5$	$r=0.7$	$r=0.9$	$r=1.0$
S-SVM (linear)	18%	16%	27%	76%	55%	92%	91%
JKSE (linear)	36%	35%	43%	60%	68%	79%	91%
JKSE (χ^2)	8%	11%	12%	14%	37%	62%	91%

Table 3.1: Equal-Error Rates for S-SVM and JKSE at different noise levels r.

is improved, even increasing it over the results achieved by S-SVM. Adding up to 30% label noise hardly decreases the accuracy compared to perfect labels. For higher noise levels the performance decreases, however always staying clearly above the results for S-SVM. Even when 90% of training labels are modified compared to the original dataset, JKSE reaches a recall level of 50% and over most of the plot precision lies above 40%.

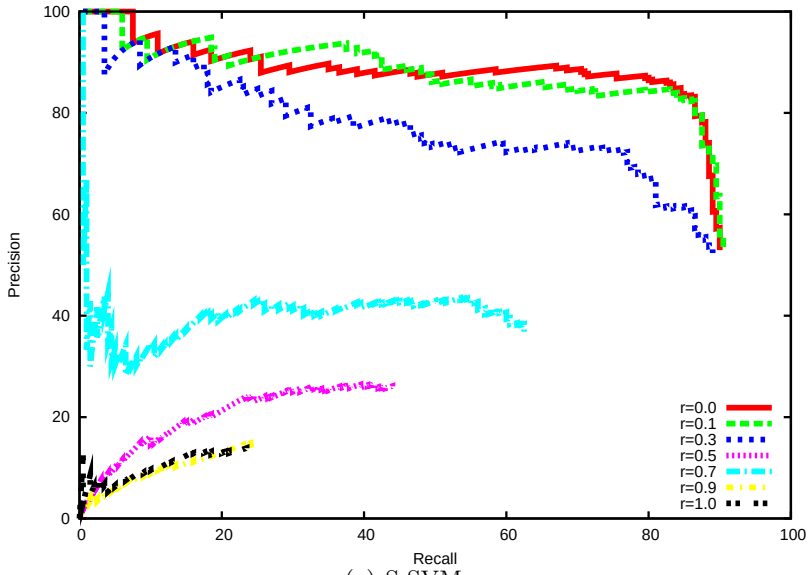
Clearly, the improved performance is a direct consequence of the use of a better kernel function. It is likely that S-SVM based localization would profit from a using a χ^2 localization kernel as well. However, as mentioned above, training S-SVM with such a kernel is not computationally feasible with current techniques.

3.2 Structured Output Ranking

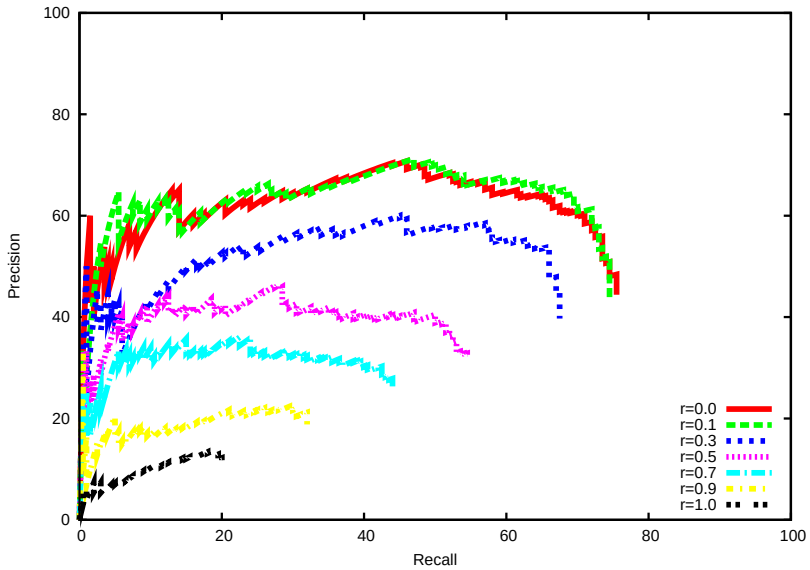
This section is based on [BVZ10, RKB11, MBZT12, BMR14] with a particular focus on [BMR14].

Learning to rank is a core task in machine learning and information retrieval [126]. We consider here a generalization to structured output prediction of the pairwise ranking SVM introduced in [89]. Similar extensions of ranking to the structured output setting [9] have recently been explored in [BVZ10, RKB11, 174]. In these works, pairwise constraints were introduced between elements in a structured output space, enforcing a margin between a lower ranked item and a higher ranked item proportional to the difference in their structured output losses. These works consider only bipartite preference graphs. Although efficient algorithms exist for cutting plane training in the bipartite special case, no feasible algorithm has previously been proposed for extending this approach to fully connected preference graphs for arbitrary loss functions.

Our work makes feasible structured output ranking with a complete preference graph for arbitrary loss functions. Joachims previously proposed an algorithm for ordinal regression with 0-1 loss and R possible ranks in $\mathcal{O}(nR)$



(a) S-SVM



(b) JKSE

Figure 3.2: Precision-recall plots of localization performance of S-SVM (3.2(a)) and JKSE (3.2(b)) for different levels of label noise (percentage indicated by r, i.e. $r = 1.0$ means completely randomized input data). At low noise levels, S-SVM clearly dominates JKSE in terms of accuracy. When the noise reaches 50% or more, S-SVM's performance drops sharply, whereas JKSE's only gradually decreases. At 90% randomized labels, JKSE is still able to achieve better than random performance.

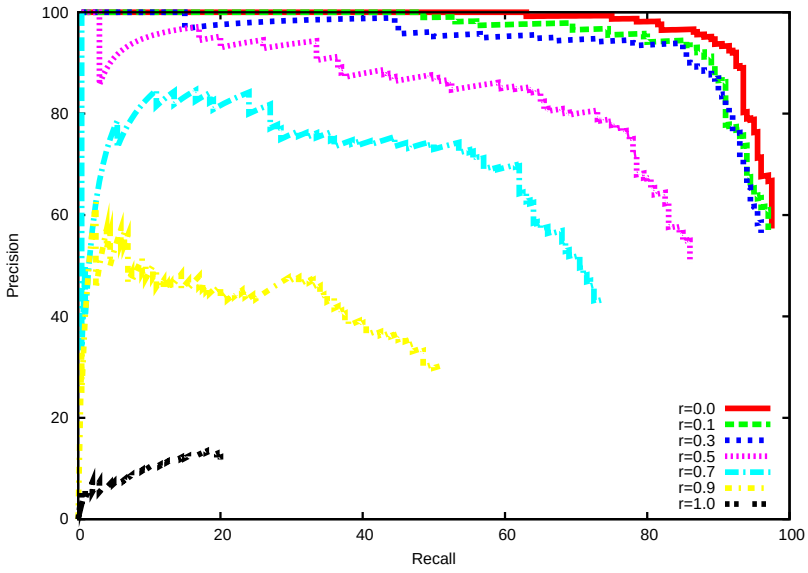


Figure 3.3: Precision-recall plots of localization performance of JKSE with χ^2-kernel. Precision and recall are clearly improved over the linear kernel. The resulting accuracy is also higher than with discriminatively trained S-SVM training (Fig. 3.2). Even if 90% of the training data is mislabeled ($r = 0.9$), localization performance is reasonable.

time for n samples [98]. This effectively enables a complete preference graph in this special setting. In practice, however, for structured output prediction with a sufficiently rich output space, the loss values may not be discrete, and may grow linearly with the number of samples. In this case, R is $\mathcal{O}(n)$. Mittal et al. have extended Joachims' $\mathcal{O}(nR)$ result to the structured output ranking setting in the case that there are a discrete set of loss values [MBZT12]. A direct extension of these approaches to the structured output setting with a fully connected preference graph and arbitrary loss functions results in a $\mathcal{O}(n^2)$ cutting plane iteration. One of the key contributions of our work is to show that this can be improved to $\mathcal{O}(n \log n)$ time. This enables us to train an objective with 5×10^7 samples on standard hardware (Section 3.2.4). Furthermore, straightforward parallelization schemes enable e.g. $\mathcal{O}(n)$ computation time on $\mathcal{O}(\log n)$ processors (Section 3.2.2). These results hold not only for the structured output prediction setting, but can be used to improve the computational efficiency of related ranking SVM approaches, e.g. [98].

Analogous to the structured output SVM [149, 154], we formulate structured output ranking in slack rescaling and margin rescaling variants. We show uniform convergence bounds for our ranking objective in a unified setting for both variants. Interestingly, the bounds for slack rescaling are dependent on the range of the loss values, while those for margin rescaling are not. Further details are given in Section 3.2.3. Structured output ranking is a natural strategy for cascade learning, in which an inexpensive feature function, ϕ, is used to filter a set of possible outputs y. We show empirical results in the cascade setting (Section 3.2.4) supporting the efficiency, accuracy, and generalization of the proposed solution to structured output prediction.

3.2.1 Structured Output Ranking

The setting considered here is to learn a compatibility function $g : \mathcal{X} \times \mathcal{Y} \mapsto \mathbb{R}$ that maps an input-output tuple to a real value indicating the prediction of how suitable the input is to a given output. We assume that there is an underlying ground truth prediction for a given input so that every $x_i \in \mathcal{X}$ in a training set is associated with a y_i^* corresponding to the optimal prediction for that input. Additionally, we assume that a loss function $\Delta : \mathcal{Y} \times \mathcal{Y} \mapsto \mathbb{R}$ is provided that measures the similarity of a hypothesized output to the optimal prediction $\Delta(y_i^*, y) \geq 0$. A training set will consist of input-ground truth-output tuples, where the input-ground truth pairs may be repeated, and the outputs are sampled over the output space: $\mathcal{S} = \{(x_i, y_i^*, y_i)\}_{1 \leq i \leq n}$

and (x_i, y_i^*) may equal (x_j, y_j^*) for $j \neq i$ (cf. Section 3.2.4). We will use the notation Δ_i to denote $\Delta(y_i^*, y_i)$.

In structured output ranking, we minimize with respect to a compatibility function, g, a risk of the form [1]

$$R(g) = \mathbb{E}_{((X_i,Y_i),(X_j,Y_j))}\big[|\Delta_{Y_j} - \Delta_{Y_i}| \cdot \big(\big[(\Delta_{Y_j} - \Delta_{Y_i}) \\ (g(X_i, Y_i) - g(X_j, Y_j)) < 0\big] \\ + \frac{1}{2}\big[g(X_i, Y_i) = g(X_j, Y_j)\big]\big], \quad (3.9)$$

where we again have used Iverson bracket notation [104], and the term penalizing equality is multiplied by $\frac{1}{2}$ in order to avoid double counting the penalty over the expectation. Here Δ_{Y_i} is the structured output loss associated with an output, Y_i. In contrast to other notions of risk, we take the expectation not with respect to a single sample, but with respect to pairs indexed by the structured output. Given two possible outputs sampled from some prior, the risk determines whether the samples are properly ordered according to the loss associated with predicting that output, and if not pays a penalty proportional to the difference in the losses. This risk penalizes pairs for which sample i has lower loss than sample j and also lower ranking score, i.e. we would like elements with low loss to be ranked higher than elements with high loss.

Two piecewise linear convex upper bounds are commonly used in structured output prediction: a margin rescaled hinge loss, and a slack rescaled hinge loss. The structured output ranking objectives corresponding to regularized risk minimization with these choices are

$$\min_{w \in \mathcal{H}, \xi \in \mathbb{R}} \lambda \Omega(w) + \xi \quad (3.10)$$

$$\text{s.t.} \sum_{(i,j) \in \mathcal{E}} \nu_{ij} \overbrace{\left(\langle w, \phi(x_i, y_i)\rangle - \langle w, \phi(x_j, y_j)\rangle + \Delta_i - \Delta_j\right)}^{\text{margin rescaling}} \geq -\xi \quad (3.11)$$

$$\text{or} \sum_{(i,j) \in \mathcal{E}} \nu_{ij} \underbrace{\left(\langle w, \phi(x_i, y_i) - \phi(x_j, y_j)\rangle - 1\right)(\Delta_j - \Delta_i)}_{\text{slack rescaling}} \geq -\xi \quad (3.12)$$

$$\xi \geq 0 \qquad \forall \nu \in \{0,1\}^{|\mathcal{E}|} \quad (3.13)$$

where $\mathcal{E}$ is the edge set associated with a preference graph $\mathcal{G}$,[3] and Ω is a regularizer monotonically increasing in some function norm applied to

[3] An edge from i to j in $\mathcal{G}$ indicates that output i should be ranked above output j. It will generally be the case that $\Delta_j \geq \Delta_i$ for all $(i, j) \in \mathcal{E}$.

w [110]. We have presented the one-slack variant here [97]. For a finite sample of (x_i, y_i^*, y_i), such objectives can be solved using a cutting plane approach [98, MBZT12, RKB11, VBZ11].

The form of $\mathcal{G}$ defines risk variants that encode different preferences in ranking. If an edge exists from node i to node j, this indicates that i should be ranked higher than j. Of particular interest in this work are bipartite graphs, which have efficiencies in computation, and fully connected graphs, which attempt to enforce a total ordering on the samples. Structured output ranking with bipartite preference graphs was previously explored in [RKB11], in which a linear time algorithm was presented for a cutting plane iteration. The algorithm presented in that work shares key similarities with previous strategies for cutting plane training of ranking support vector machines [98], but extends the setting to rescaled structured output losses. A linear time algorithm for fully connected preference graphs was presented in [MBZT12] in the special case that the loss values are in a small discrete set. Previous algorithms all degenerate to $\mathcal{O}(n^2)$ when applied to fully connected preference graphs with arbitrary loss values.

3.2.2 $\mathcal{O}(n \log n)$ Cutting Plane Algorithm

Cutting plane optimization of (3.10)-(3.13) consists of alternating between optimizing the objective with a finite set of active constraints, finding a maximally violated constraint of the current function estimate and adding it to the active constraint set [97]. Algorithm 1 gives a linear time procedure for finding the maximally violated constraint in the case of a complete bipartite preference graph [98, RKB11] and slack rescaling.[4] This algorithm follows closely the ordinal regression cutting plane algorithm of [98], and works by performing an initial sort on the current estimate of the sample scores. The algorithm subsequently makes use of the transitivity of violated pairwise constraints to sum all violated pairs in a single pass through the sorted list of samples.

In the case of fully connected preference graphs, Algorithm 2 is a recursive function that ensures that all pairs of samples are considered. Algorithm 2 uses a divide and conquer strategy and works by repeatedly calling Algorithm 1 for various bipartite subgraphs with disjoint edge sets, ensuring that the union of the edge sets of all bipartite subgraphs is the edge set of the preference graph. The set of bipartite subgraphs is constructed by partitioning the set of samples into two roughly equal parts by thresholding the

[4] An analogous algorithm for margin rescaling was given in [RKB11] and has the same computational complexity.

Algorithm 1 Finding maximally violated slack-rescaled constraint for structured output ranking with a complete bipartite preference graph.

Require: Δ, a list of loss values sorted from lowest to highest; s, a vector of the current estimate of compatibility scores ($s_u = \langle w, \phi(x_u, y_u)\rangle_{\mathcal{H}}$) in the same order as Δ; p, a vector of indices such that $s_{p_v} > s_{p_u}$ whenever $v > u$; t, a threshold such that $(u, v) \in \mathcal{E}$ whenever $u \leq t$ and $v > t$

Ensure: Maximally violated constraint is $\delta - \langle w, \sum_i \alpha_i \phi(x_i, y_i)\rangle \leq \xi$

$p^+ = p_{\{u|p_u \leq t\}}$, $p^- = p_{\{v|p_v > t\}}$
$i = 1$, $\delta = \Delta_+ = 0$, $\Delta^{\text{cum}} = \mathbf{0}$, $\alpha = \mathbf{0}$
$\Delta^{\text{cum}}_{n-t} = \Delta_{p^-_{n-t}}$
for $k = n - t - 1$ to 1 descending **do**
    $\Delta^{\text{cum}}_{p^-_k} = \Delta_{p^-_k} + \Delta^{\text{cum}}_{p^-_{k+1}}$
end for
for $j = 1$ to $n - t$ **do**
    while $s_{p^-_j} + 1 > s_{p^+_i} \wedge i \leq t + 1$ **do**
        $\alpha_{p^+_i} = \alpha_{p^+_i} + \Delta^{\text{cum}}_{p^-_j} - (n - t - j + 1)\Delta_{p^+_i}$
        $\Delta_+ = \Delta_+ + \Delta_{p^+_i}$, $i = i + 1$
    end while
    $\alpha_{p^-_j} = \alpha_{p^-_j} - ((j - 1)\Delta_{p^-_j} - \Delta_+)$
    $\delta = \delta + (j - 1)\Delta_{p^-_j} - \Delta_+$
end for
return (α, δ)

Algorithm 2 An $\mathcal{O}(n \log n)$ recursive algorithm for computing a cutting plane iteration for fully connected ranking preference graphs.

Require: Δ, a list of loss values sorted from lowest to highest; s, a vector of the current estimate of compatibility scores ($s_u = \langle w, \phi(x_u, y_u) \rangle_{\mathcal{H}}$) in the same order as Δ; p, an index such that $s_{p_v} > s_{p_u}$ whenever $v > u$

Ensure: Maximally violated constraint is $\delta - \langle w, \sum_i \alpha_i \phi(x_i, y_i) \rangle \leq \xi$

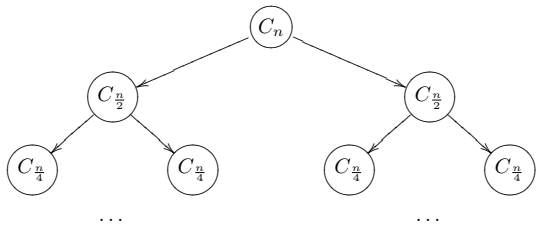
1: $n = \text{length}(\Delta)$
2: **if** $\Delta_1 = \Delta_n$ **then**
3: **return** $(\mathbf{0}, 0)$
4: **end if**
5: $t \approx \frac{n}{2}$
6: $p^a = p_{\{u | p_u \leq t\}}$
7: $(\alpha_1, \delta_1) =$ Algorithm 2$(\Delta_{1:t}, s_{1:t}, p^a)$
8: $p^b = p_{\{v | p_v > t\}}$
9: $p^b = p^b - t$ (subtract t from each element of p^b)
10: $(\alpha_2, \delta_2) =$ Algorithm 2$(\Delta_{t+1:n}, s_{t+1:n}, p^b)$
11: $(\alpha_0, \delta_0) =$ Algorithm 1(Δ, s, p, t)
12: $\alpha = \alpha_0 + \alpha_1 + \alpha_2, \delta = \delta_0 + \delta_1 + \delta_2$
13: **return** (α, δ)

loss function. As the samples are assumed to be sorted by their structured output loss, we simply divide the set by computing the index of the median element. In the event that there are multiple samples with the same loss, the partitioning (Algorithm 2, line 5) may do a linear time search from the median loss value to find a partitioning of the samples such that the first set has strictly lower loss than the second. The notation $p^a = p_{\{u | p_u \leq t\}}$ indicates that p^a contains the elements satisfying the condition in the subscript in the same order that they occured in p.

Complexity

Prior to calling either of the algorithms, the current data sample must be sorted by its structured output loss. Additionally an index vector, p, must be computed that encodes a permutation matrix that sorts the training sample by the current estimate of its compatibility scores, $\langle w, \phi_i \rangle$. Each of these operations has complexity $\mathcal{O}(n \log n)$. The serial complexity of computing the most violated 1-slack constraint is $\mathcal{O}(n \log_2 n)$, matching the complexity of the sorting operation. To show this, we consider the recursion in Algorithm 2. The computational costs of each call consist of (i) the processing needed to find the sorted list of scores for the higher ranked

Figure 3.4: The recursion tree for Algorithm 2. Each node in the tree corresponds to a set of constraints resulting from a bipartite preference graph. The cost of computing these constraints is labeled in each of the nodes.



and lower ranked subsets in the bipartite graph, (ii) the cost of calling Algorithm 1, and (iii) the cost of recursion. We will show that items (i) and (ii) can be computed in time linear in the number of samples.

That item (i) is linear in its complexity can be seen by noting that an index p already exists to sort the complete data sample. Rather than pay $\mathcal{O}(n \log n)$ to re-sort the subsets of samples, we may iterate through the elements of p once. As we do so, if $p_i \leq t$, we may add this element to the index that sorts the higher ranked subset. If $p_j > t$, we may add $p_j - t$ to the index that sorts the lower ranked subset. Item (ii) is also linear as the algorithm loops once over each data sample, executing a constant number of operations each time.

We calculate the complexity of Algorithm 2 by a recursive formula $R_n = C_n + 2R_{\frac{n}{2}}$ where C_n is the $\mathcal{O}(n)$ cost of processing items (i) and (ii). It follows that

$$R_n = \sum_{i=0}^{\log_2 n} C_{\frac{n}{2^i}} 2^i. \tag{3.14}$$

Examining the term $C_{\frac{n}{2^i}} 2^i$, we note that $C_{\frac{n}{2^i}}$ is $\mathcal{O}(\frac{n}{2^i})$ and must be paid 2^i times, resulting in a cost of $\mathcal{O}(n)$ per summand. As there are $\mathcal{O}(\log_2 n)$ summands, the total cost is $\mathcal{O}(n \log n)$. Graphically, the recursion tree is a binary tree in which the cost of each node is proportional to $\frac{1}{2^d}$, where d is the depth of the node (Figure 3.4). A C implementation of the algorithm takes a fraction second for 10^5 samples on a 2.13 GHz processor.

A straightforward parallelization scheme can be achieved by placing each recursive call in its own thread. Doing so results in $\mathcal{O}(n)$ computation on $\mathcal{O}(\log n)$ processors: each level of a tree at depth i can be computed independently in $C_{\frac{n}{2^i}} 2^i$ instructions, and there are $\mathcal{O}(\log n)$ levels of the tree. Each of $\log n$ processors can be assigned the nodes corresponding to a given level of the tree.

3.2.3 Generalization Bounds

In this section, we develop generalization bounds based on the uniform convergence bounds for ranking algorithms presented in [1]. For $\Delta \in [0, 1)$ we have tighter bounds for slack rescaling as compared to margin rescaling. For $\Delta \in [0, \sigma]$ where $\sigma > 1$ bounds are tighter for margin rescaling.

Definition 3.2.1 (Uniform loss stability (β))**.** A ranking algorithm which is trained on the sample $\mathcal{S}$ of size n has a uniform loss stability β with respect to the ranking loss function ℓ if,

$$|\ell(\mathcal{S}) - \ell(\mathcal{S}^k)| \leq \beta(n), \ \ \forall n \in \mathbb{N}, 1 \leq k \leq n \tag{3.15}$$

where $\mathcal{S}^k$ is a sample resulting from changing the kth element of $\mathcal{S}$, i.e., changing the input training sample by a single example leads to a difference of at most $\beta(n)$ in the loss incurred by the output ranking function on any pair of examples. Thus, a smaller value of $\beta(n)$ corresponds to a greater loss stability.

Definition 3.2.2 (Uniform score stability (ν))**.** A ranking algorithm with an output $g_{\mathcal{S}}$ on the training sample $\mathcal{S}$ of size n, has a uniform score stability ν if

$$|g_{\mathcal{S}}(x) - g_{\mathcal{S}^k}(x)| \leq \nu(n), \ \ \forall\, n \in N, 1 \leq k \leq n, \forall x \in \mathcal{X} \tag{3.16}$$

i.e., changing an input training sample by a single example leads to a difference of at most $\nu(n)$ in the score assigned by the ranking function to any instance x.

The hinge losses for margin and slack rescaling formulations are given by:

$$\ell_m = (|\Delta_j - \Delta_i| - \langle w, \phi(x_i, y_i) - \phi(x_j, y_j)\rangle \cdot \text{sign}(\Delta_j - \Delta_i))_+, \tag{3.17}$$

$$\ell_s = (|\Delta_j - \Delta_i| \cdot (1 - \langle w, \phi(x_i, y_i) - \phi(x_j, y_j)\rangle \cdot \text{sign}(\Delta_j - \Delta_i)))_+. \tag{3.18}$$

Theorem 3.2.3. *Let $\mathcal{A}$ be a ranking algorithm whose output on a training sample $\mathcal{S} \in (\mathcal{X}, \mathcal{Y})^n$ we denote by $f_{\mathcal{S}}$. Let $\nu : \mathbb{N} \to \mathbb{R}$ be such that $\mathcal{A}$ has uniform score stability ν. $\mathcal{A}$ has uniform loss stability β with respect to the slack rescaling loss ℓ_s, where for all $n \in \mathbb{N}$*

$$\beta(n) = 2\sigma\nu(n) \tag{3.19}$$

where $\sigma \geq \Delta$ is an upper bound on the structured output loss function.

Proof. Without loss of generality we assume that $\ell_s(\mathcal{S}) > \ell_s(\mathcal{S}^k)$. There are two non-trivial cases.

Case (i): Margin is violated by both $g_{\mathcal{S}}$ and $g_{\mathcal{S}^k}$.

$$|\ell_s(\mathcal{S}) - \ell_s(\mathcal{S}^k)| = |\Delta_j - \Delta_i| \cdot (1 - (g_{\mathcal{S}}(x_i) - g_{\mathcal{S}}(x_j)) \cdot \text{sign}(\Delta_j - \Delta_i)) - \tag{3.20}$$

$$|\Delta_j - \Delta_i| \cdot (1 - (g_{\mathcal{S}^k}(x_i) - g_{\mathcal{S}^k}(x_j)) \cdot \text{sign}(\Delta_j - \Delta_i))$$

$$\leq \sigma(|g_{\mathcal{S}}(x_i) - g_{\mathcal{S}^k}(x_i)| + |g_{\mathcal{S}}(x_j) - g_{\mathcal{S}^k}(x_j)|) \leq 2\sigma\nu(n) \tag{3.21}$$

Case (ii): Margin is violated by either of $g_{\mathcal{S}}$ or $g_{\mathcal{S}^k}$. This is a symmetric case, so we assume that the margin is violated by $g_{\mathcal{S}}$.

$$|\ell_s(\mathcal{S}) - \ell_s(\mathcal{S}^k)| = |\Delta_j - \Delta_i| \cdot (1 - (g_{\mathcal{S}}(x_i) - g_{\mathcal{S}}(x_j)) \cdot \text{sign}(\Delta_j - \Delta_i)) \tag{3.22}$$

$$\leq |\Delta_j - \Delta_i| \cdot (1 - (g_{\mathcal{S}}(x_i) - g_{\mathcal{S}}(x_j)) \cdot \text{sign}(\Delta_j - \Delta_i)) - \tag{3.23}$$

$$|\Delta_j - \Delta_i| \cdot (1 - (g_{\mathcal{S}^k}(x_i) - g_{\mathcal{S}^k}(x_j)) \cdot \text{sign}(\Delta_j - \Delta_i))$$

$$\leq \sigma(|g_{\mathcal{S}}(x_i) - g_{\mathcal{S}^k}(x_i)| + |g_{\mathcal{S}}(x_j) - g_{\mathcal{S}^k}(x_j)|) \leq 2\sigma\nu(n) \tag{3.24}$$

□

Theorem 3.2.4 (Slack Rescaling Generalization Bound)**.** *Let $\mathcal{H}$ be a RKHS with a joint-kernel*[5] *k such that $\forall (x, y) \in \mathcal{X} \times \mathcal{Y}$, $k((x, y), (x, y)) \leq \kappa^2 < \infty$. Let $\lambda > 0$ and ℓ_r be a rescaled ramp loss. The training algorithm trained on sample $\mathcal{S}$ of size n outputs a ranking function $g_{\mathcal{S}} \in \mathcal{H}$ that satisfies $g_{\mathcal{S}} = \arg\min_{g \in \mathcal{H}} \{\hat{R}_{\ell_s}(g; \mathcal{S}) + \lambda ||g||^2_{\mathcal{H}}\}$. Then for any $0 < \delta < 1$, with probability at least $1 - \delta$ over the draw of $\mathcal{S}$, the expected ranking error of the function is bounded by:*

$$R(g_{\mathcal{S}}) < \hat{R}_{\ell_r}(g_{\mathcal{S}}; \mathcal{S}) + \frac{32\sigma^2\kappa^2}{\lambda n} + \left(\frac{16\sigma^2\kappa^2}{\lambda} + \sigma\right)\sqrt{\frac{2\ln(1/\delta)}{n}} \tag{3.25}$$

Proof. From [1, Theorem 11], $\nu(n) = \frac{8\sigma\kappa^2}{\lambda n}$. Substituting this value of $\nu(n)$ in Equation (3.19) $\beta(n) = \frac{16\sigma^2\kappa^2}{\lambda n}$. Inequality (3.25) then follows by an application of [1, Theorem 6] which gives the generalization bound as a function of $\beta(n)$. □

[5] We assume a joint kernel map of the form given in [149, 154].

The proof of [1, Theorem 6] follows closely that of [34] for regression and classification, relying at its core on McDiarmid's inequality [130].

Theorem 3.2.5 (Margin Rescaling Generalization Bound)**.** *Under the conditions of Theorem 3.2.4, and a ranking function $f_{\mathcal{S}} \in \mathcal{H}$ that satisfies $f_{\mathcal{S}} = \arg\min_{f \in \mathcal{H}} \{\hat{R}_{\ell_m}(f; \mathcal{S}) + \lambda ||f||^2_{\mathcal{H}}\}$. Then for any $0 < \delta < 1$, with probability at least $1 - \delta$ over the draw of $\mathcal{S}$, the expected ranking error of the function is bounded by:*

$$R(f_{\mathcal{S}}) < \hat{R}_{\ell_r}(f_{\mathcal{S}}; \mathcal{S}) + \frac{32\kappa^2}{\lambda n} + \left(\frac{16\kappa^2}{\lambda} + \sigma\right)\sqrt{\frac{2\ln(1/\delta)}{n}} \tag{3.26}$$

The proof of Theorem 3.2.5 follows the outline given in [1, Section 5.2.1].

3.2.4 Experimental Results

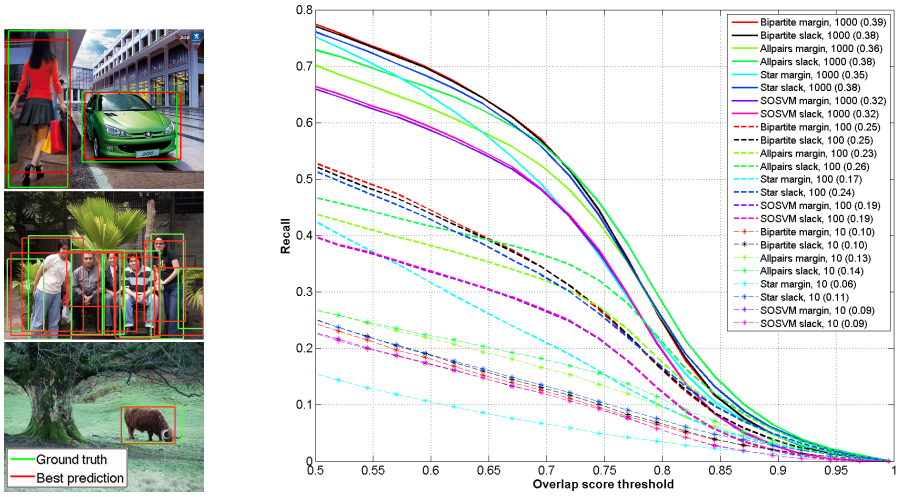
Results are presented as an evaluation of a cascade architecture [162], following the evaluation protocol of Rahtu et al. [RKB11]. The experiments are presented on the VOC 2007 dataset [52]. The images are annotated with ground-truth bounding boxes of objects from 20 classes. VOC 2007 train and validation sets are used only to construct the distribution for the initial window sampling, and the ranking function is learned using the dataset presented in [2]. This is done in order to obtain results comparable to those in [2, RKB11].

The performance is measured using a recall-overlap curve, which indicates the recall rate of ground truth boxes in the VOC 2007 test set for a given minimum value of the overlap score [158]

$$o(y, \tilde{y}) = \frac{\text{Area}(y \cap \tilde{y})}{\text{Area}(y \cup \tilde{y})}, \tag{3.27}$$

where y and $\tilde{y}$ denote the ground truth and predicted bounding box, respectively. We also report the area under the curve (AUC) between overlap scores 0.5 and 1, and normalize its value so that the maximum is 1 for perfect recall. The overlap limit 0.5 is chosen here since less accurately localized boxes have little practical importance.

Our framework for creating the set of predicted bounding boxes broadly follows that of [RKB11]. This setting has three main stages: (i) construction of the initial bounding boxes, (ii) feature extraction, and (iii) window selection. In the first stage we generate a pool of approximately 100,000 initial windows per image using random sampling and superpixel bounding boxes. The random samples are drawn from a distribution learned using the ground



(a) Example detections with a complete preference graph and slack rescaling. This setting corresponds to "Allpairs slack, 1000" in Fig. (b).

(b) Overlap/recall curves. Results are presented for varying preference graphs, margin and slack rescaling, and various numbers of returned windows. The AUC score is given in parentheses (a higher number at a given number of returned windows indicates better performance).

Figure 3.5: Example detections and overlap vs. recall for an object detection task. See Section 3.2.4 for a complete description of the experimental setting. This figure is best viewed in color.

truth object boxes in the training and validation sets. The superpixels are computed by a graph based method [55], which is selected for its computational efficiency. At overlap 0.5, the initial windows achieve approximately 98% recall.

In the second stage, the tentative bounding boxes are scored using several publicly available features. These features are window symmetry (WS), boundary edge distribution (BE), superpixel boundary integral (BI), color contrast (CC), superpixel straddling (SS), and multiscale saliency (MS). The WS, BE, and BI features are described in [RKB11] and SS, CC, and MS are from [2]. The joint feature map, $\phi(x_i, y_i)$, applied in learning is the feature vector corresponding to the bounding box y_i.

In the last stage, we select the final set of bounding boxes (10, 100 or 1000) based on the learned score. The feature weights for the linear combination are learned by using the structured output ranking framework presented in this section and the loss function proposed in [29]. This loss is based on the overlap ratio (3.27) and is defined as $\Delta_i \equiv 1 - o(y, y_i)$.

In order to run the proposed algorithm, we further need to define the structure of the preference graph $\mathcal{G}$. Three variants were considered: a bipartite graph in which 1000 best samples per image are ranked higher than all other initial windows (as in [RKB11]), a fully connected graph (denoted "Allpairs" in the legend of Figure 3.5(b)) where full ranking is pursued, or a bipartite graph in which only ground truth windows are to be ranked higher than all sampled windows (denoted "Star" in the legend, as the topology of a bipartite graph with one singleton set is a star graph). Finally, we have trained a standard structured output SVM (labeled "SOSVM") in the same manor as [29]. To ensure a diverse set of predictions, we have applied the non-maximal suppression approach described in [158].

The overlap-recall curves are shown in Figure 3.5. The legend in Figure 3.5 encodes the experimental setting for each curve. First, the structure of the preference graph, $\mathcal{G}$, is specified. The second component of the legend indicates whether slack rescaling or margin rescaling was employed. The third component states the number of top ranked windows used for evaluating the recall. Finally, the fourth component (in parentheses) gives the AUC value.

3.2.5 Discussion

The experiments described in Section 3.2.4 show that structured output ranking is a natural objective to apply to cascade detection models.

On average, a bipartite preference graph performs best if we require 1000 windows as output, which matches the training conditions. The bipartite graph was constructed such that constraints were included between the top 1000 sampled windows, and the remaining 99000 windows. However, when the number of returned windows deviates from 1000, the relative performance of the bipartite ranking decreases and other preference graphs give better performance. The objective is tuned to give the highest performance under a single evaluation setting, at the expense of other settings.

The complete preference graph ranking, labeled "Allpairs" in Figure 3.5, gives good performance and tends to have higher performance at high overlap levels. While the difference between slack rescaling and margin rescaling was minimal when using a bipartite preference graph, a much more noticeable difference is present in the case of a complete preference graph. While the bipartite preference graph performs better at certain overlap levels when 1000 windows are returned, the complete preference graph is much more stable across a wide number of windows, and gives the best performance at all overlap levels if 10 windows are returned per image. Finally, the standard

structured output SVM (labeled "SOSVM") performs substantially worse than all ranking variants.

In this section, we have explored the use of ranking for structured output prediction. We have analyzed both margin and slack rescaling variants of a ranking SVM style approach, showing better empirical results for slack rescaling, and proving generalization bounds for both variants in a unified framework. Furthermore, we have proposed an efficient and parallelizable algorithm for cutting plane training that scales to millions of data points on a single core. We have shown an example application of object detection in computer vision, demonstrating that ranking methods outperform a standard structured output SVM in this setting, and that fully connected preference graphs give excellent performance across a range of settings, particularly at high overlap with the ground truth.

The $\mathcal{O}(n \log n)$ algorithm presented here can be adapted to a wide variety of settings, improving the computational efficiency in a range of ranking approaches and applications. In the setting of [98, MBZT12], the $\mathcal{O}(nR)$ approach for ranking with a complete preference graph and a fixed number, R, of loss values can be improved in an analogous manner to $\mathcal{O}(n \log R)$.

3.3 Discussion

In this chapter, we have provided an overview of our contributions in novel risk formulations for structured prediction. We have given a specific focus to two contributions based on joint kernel support estimation and structured output ranking. In the next chapter, we present some of our contributions to the use of expressive function classes and regularization techniques.

Chapter 4

Function Classes and Regularization

This chapter outlines work that I have done contributing to the development of novel function classes and applications of semi-supervised and sparsity regularizers. These topics are naturally grouped in defining the notion of complexity of a prediction function. The chapter first discusses applications and contributions in semi-supervised Laplacian regularization (Section 4.1). Subsequently, in Section 4.2, structured sparsity regularization based on the k-support norm is presented. Finally, the development of a novel graph kernel for continuous and vector-valued node labels is presented in Section 4.3.

4.1 Semi-supervised Laplacian Regularization

This section is based in part on [BSB09, BSB+11] which in turn built upon [30]. In this section, we present the application of semi-supervised Laplacian regularization to canonical correlation analysis, and the application of the resulting statistical technique to fMRI analysis.

Canonical correlation analysis (CCA) is a fundamental technique in statistics and dimensionality reduction that relies on paired data to learn directions that maximize correlation between the projected representations in each space [93]. It is readily kernelized (KCCA), enabling a straightforward non-linear generalization [121, 111, 7, 82]. Dimensionality reduction techniques that rely on only one modality are incapable of distinguishing semantically meaningless noise directions, and are not discriminative in nature. In contrast, KCCA is able to learn relevant directions by requiring that embedded data be correlated with embeddings of data in other modalities,

and has been shown to increase class separability when compared to single modality dimensionality reduction [28].

While KCCA often gives superior results to dimensionality reduction techniques that work on a single modality, it does not directly estimate the data manifold in any given modality. Additionally, it is only able to utilize data for which correspondence is known to the other modalities. In order to more robustly learn the relevant directions in the feature space, we can modify our objective to favor directions that lie along the data manifold. In this section, we describe a method that incorporates these two goals by employing semi-supervised Laplacian regularization [17]. This method gives an embedding of the data that makes use of the information between modalities, as well as the information within each single modality. By using Laplacian regularization, we are able to learn directions that tend to lie along the data manifold estimated from a much larger set of data [17]. This gives us greater confidence that the learned directions represent the underlying statistical structure of the data and that we have not been misled by small sample effects. We show experimentally that learning along the manifold results in increased performance, even in the fully supervised setting, in that the learned embeddings give better hold out correlations for a human fMRI task. Additionally, we show that the learned projection vectors are interpretable as representing brain regions that are implicated in the corresponding visual processing task.

4.1.1 A Review of Kernel Canonical Correlation Analysis

Canonical Correlation Analysis

Canonical correlation analysis (CCA) utilizes datasets where samples are available in more than one modality. CCA projects the data samples from each modality into a subspace such that the empirical correlation of the projected data is maximized [93]. Given a sample from a paired dataset $\{(x_1, y_1), \ldots, (x_n, y_n)\} \in (\mathcal{X} \times \mathcal{Y})^n$, CCA simultaneously finds directions w_x and w_y that maximize the correlation of the projections of x onto w_x with the projections of y onto w_y. This is expressed as

$$\max_{w_x, w_y} \frac{\hat{E}[\langle x - \mu_x, w_x \rangle \langle y - \mu_y, w_{y\rangle}]}{\sqrt{\hat{E}[\langle x - \mu_x, w_x \rangle^2] \hat{E}[\langle y - \mu_y, w_y \rangle^2]}} \tag{4.1}$$

where $\hat{E}$ denotes the empirical expectation, and μ_x and μ_y the empirical means in each of the modalities. We may view the general assumptions

of CCA as being that samples from $\mathcal{X}$ and $\mathcal{Y}$ are generated from some underlying process which induces a dependence between our paired samples.

We introduce the notation C to represent the covariance matrix of samples in $\mathcal{X} \times \mathcal{Y}$, and note that C decomposes into auto-covariance matries, and cross-covariance matrices

$$C = \begin{pmatrix} C_{xx} & C_{xy} \\ C_{yx} & C_{yy} \end{pmatrix} \tag{4.2}$$

where C_{xx} and C_{yy} are auto-covariance matrices, and $C_{xy} = C_{yx}^T$ are cross covariance matrices. Using this notation, we may rewrite Equation (4.1) to obtain

$$\max_{w_x, w_y} \frac{w_x^T C_{xy} w_y}{\sqrt{w_x^T C_{xx} w_x \; w_y^T C_{yy} w_y}}. \tag{4.3}$$

This Rayleigh quotient can be optimized as a generalized eigenvalue problem, or by decomposing the problem as described in [82].

Kernel Canonical Correlation Analysis

We denote $\mathcal{H}_x$ the reproducing kernel Hilbert space (RKHS) associated with k_x, and denote the associated feature map $\phi_x : \mathcal{X} \to \mathcal{H}$, i.e. $k_x(x_i, x_j) = \langle \phi_x(x_i), \phi_x(x_j) \rangle$. We note that in general $\phi_x(x_i)$ may no longer have an interpretation in a finite dimensional vector space, but can be viewed as an element in a function space. We analogously define k_y, $\mathcal{H}_y$, and ϕ_y.

We may adapt the representer theorem [102, 140] to the case of multimodal data to state that minimizers of the risk functional

$$\begin{aligned} \min_{f_1, \ldots, f_k} \; & c\big((x_1^1, \ldots, x_1^k, f_1(x_1^1), \ldots, f_k(x_1^k)), \ldots, \\ & (x_n^1, \ldots, x_n^k, f_1(x_n^1), \ldots, f_k(x_n^k))\big) + \\ & \sum_{i=1}^{k} \Omega_i(\|f_i\|_{\mathcal{H}_i}^2), \end{aligned} \tag{4.4}$$

where c is an arbitrary loss function and Ω a strictly monotonic increasing function, admit representations of the form

$$f_i(x) = \sum_{j=1}^{n} \alpha_j^i k_i(x_j^i, x), \tag{4.5}$$

where x_j^i represents the jth sample in the ith modality and $f_i \in \mathcal{H}_i$ a function that maps a sample in the ith modality to the reals. This follows directly

from the representer theorem by considering each modality individually (f_i) while holding all other parameters fixed (f_l where $l \neq i$).

As a result, we may consider a kernelized version of CCA (KCCA). We replace vectors w_i in our previous linear formulation with functions f_i, and replace covariance matrices with the covariance operator

$$\hat{C} = \frac{1}{n}\sum_{i=1}^{n}(\phi(x_i) - \mu_\phi)(\phi(x_i) - \mu_\phi)^T, \tag{4.6}$$

a linear operator that maps $f \in \mathcal{H}$ to $\frac{1}{n}\sum_{i=1}^{n}\phi(x_i)\langle\phi(x_i), f\rangle$ [139]. As we are working with multimodal data, we may consider $\mathcal{H} = \bigoplus_{i=1}^{k}\mathcal{H}_i$ and f to be the concatenation of each f_i. We have used the notation μ_ϕ here to denote the empirical mean of our data sample in the Hilbert space. Analogously to Section 4.1.1, we may also define cross-covariance and auto-covariance operators $\hat{C}_{xy}$ and $\hat{C}_{xx}$.

Restricting ourselves for the present to the two modality case, we may write the KCCA objective as

$$\max_{f_x, f_y}\frac{f_x^T\hat{C}_{xy}f_y}{\sqrt{f_x^T\hat{C}_{xx}f_x\ f_y^T\hat{C}_{yy}f_y}} = \max_{\alpha,\beta}\frac{\alpha^T K_x K_y \beta}{\sqrt{\alpha^T K_x^2\alpha\ \beta^T K_y^2\beta}}, \tag{4.7}$$

where $f_x = \sum_i \alpha_i\phi_x(x_i)$, $f_y = \sum_i \beta_i\phi_y(y_i)$, K_x is the kernel matrix such that $[\tilde{K}_x]_{ij} = k_x(x_i, x_j)$ and $K_x = H\tilde{K}_x H$ where H is a centering matrix

$$H = I - \frac{1}{n}ee^T \tag{4.8}$$

$e \in \mathbb{R}^n$ being a vector of all ones. As discussed in [121, 7, 82] this optimization leads to degenerate solutions in the case that either K_x or K_y is invertible so we maximize the following regularized expression

$$\max_{\alpha,\beta}\frac{\alpha^T K_x K_y \beta}{\sqrt{\alpha^T\left(K_x^2 + \varepsilon_x K_x\right)\alpha\beta^T\left(K_y^2 + \varepsilon_y K_y\right)\beta}}, \tag{4.9}$$

which is equivalent to Tikhonov regularization of the norms of w_x and w_y in the denominator of Equation (4.3). In the limit case that $\varepsilon_x \to \infty$ and $\varepsilon_y \to \infty$, the algorithm maximizes covariance instead of correlation.

4.1.2 Semi-supervised Kernel Canonical Correlation Analysis

Semi-supervised learning is usually presented in the setting of regression or binary classification [40]. In this setting, the task is to learn a mapping f :

$\mathcal{X} \to \mathcal{Y}$, where training data are of the form $\{(x_1, y_1), \ldots, (x_n, y_n)\}$, with additional unlabeled training data available in the $\mathcal{X}$ domain, $\{x_{n+1}, \ldots, x_{n+p_x}\}$. We will use the variable $m_x = n + p_x$ for notational convenience.

Semi-supervised Laplacian Regularization

Laplacian regularization introduces an additional term into a regularized risk function. One may still regularize using a standard function norm on f, as in Tikhonov regularization, but an additional term penalizes deviations from the data manifold [17]. The representation of the data manifold is estimated empirically from training data, and the additional samples $\{x_{n+1}, \ldots, x_{m_x}\}$ allow us to obtain a much more robust estimate.

In the classic setting, we wish to solve

$$\begin{aligned} \min_{f_x \in \mathcal{H}_x} \; & c((x_1, y_1, f_x(x_1)), \ldots, (x_n, y_n, f_x(x_n))) \\ & + \varepsilon_x \|f_x\|^2_{\mathcal{H}_x} + \gamma_x \int_{x \in \mathcal{M}} \|\nabla_{\mathcal{M}} f_x\|^2 d\mathcal{P}_x(x) \end{aligned} \tag{4.10}$$

where γ_x is the regularization parameter controlling the degree of Laplacian regularization, $\mathcal{P}_x$ is the marginal distribution of x, and $\nabla_{\mathcal{M}}$ is the gradient of f_x along the manifold $\mathcal{M}$. We do not directly observe $\mathcal{M}$ or $\mathcal{P}_x$ so we must estimate these from the data. As the graph Laplacian converges to the Laplace-Beltrami operator under appropriate conditions [87], we can approximate the integral using the graph Laplacian [17]

$$\begin{aligned} \min_{f_x \in \mathcal{H}_x} \; & c((x_1, y_1, f_x(x_1)), \ldots, (x_n, y_n, f_x(x_n))) \\ & + \varepsilon_x \|f_x\|^2_{\mathcal{H}_x} + \frac{\gamma_x}{m_x^2} f_x^T \mathcal{L}_{\hat{x}} f_x, \end{aligned} \tag{4.11}$$

where $\hat{x}$ denotes that the empirical graph Laplacian $\mathcal{L}$ was estimated from both labeled and unlabeled data. One may also prove a representer theorem for this form of optimization, in which the minimizer lies in the span of the combined labeled and unlabeled training data [17].

The Two-modality Case

We now have all the necessary ingredients to apply semi-supervised Laplacian regularization to kernel canonical correlation analysis. KCCA deviates from the classic setting in that modalities $\mathcal{X}$ and $\mathcal{Y}$ are symmetric, and we wish to simultaneously optimize functions that act on each of them. Consequently, we develop notation for kernel matrices with and without semi-supervised data over both the $\mathcal{X}$ and $\mathcal{Y}$ domains. We denote the design

matrix $X = (x_1, \ldots, x_n)$ where each column represents a data sample that has a correspondence to an observation in $\mathcal{Y}$. We denote the extended design matrix $\hat{X} = (x_1, \ldots, x_{m_x})$, in which all data with and without correspondences are stored. We similarly define matrices Y and $\hat{Y}$. We now denote the kernel matrix computed only using the data in X as $K_{xx} \in \mathbb{R}^{n \times n}$, the matrix computed using $\hat{X}$ and X as $K_{\hat{x}x} \in \mathbb{R}^{m_x \times n}$, the matrix computed using $\hat{X}$ with itself as $K_{\hat{x}\hat{x}} \in \mathbb{R}^{m_x \times m_x}$, etc. Kernel matrices for $\mathcal{Y}$ are defined analogously. The following is a semi-supervised Laplacian regularized generalization of Equation (4.9)

$$\max_{\alpha,\beta} \frac{\alpha^T K_{\hat{x}x} K_{y\hat{y}} \beta}{\sqrt{\alpha^T \left(K_{\hat{x}x} K_{x\hat{x}} + R_{\hat{x}}\right) \alpha \beta^T \left(K_{\hat{y}y} K_{y\hat{y}} + R_{\hat{y}}\right) \beta}}, \tag{4.12}$$

where $R_{\hat{x}} = \varepsilon_x K_{\hat{x}\hat{x}} + \frac{\gamma_x}{m_x^2} K_{\hat{x}\hat{x}} \mathcal{L}_{\hat{x}} K_{\hat{x}\hat{x}}$ and $R_{\hat{y}} = \varepsilon_y K_{\hat{y}\hat{y}} + \frac{\gamma_y}{m_y^2} K_{\hat{y}\hat{y}} \mathcal{L}_{\hat{y}} K_{\hat{y}\hat{y}}$.

4.1.3 Experimental Results

Data

fMRI data of one human volunteer was acquired using a Siemens 3T TIM scanner, and consisted of 350 time slices of 3-dimensional fMRI brain volumes. Time-slices were separated by 3.2 s (TR), each with a spatial resolution of 46 slices (2.6 mm width, 0.4 mm gap) with 64x64 pixels of 3x3 mm, resulting in a spatial resolution of 3x3x3 mm. The subject watched 2 movies of 18.5 min length, one of which had labels indicating the continuous content of the movie (i.e. degree of visual contrast, or the degree to which a face was present, etc.). The imaging data were pre-processed using standard procedures using the Statistical Parametric Mapping (SPM5) toolbox before analysis [64]. This included a slice-time correction to compensate for acquisition delays between slices, a spatial realignment to correct for small head-movements, a spatial normalization to the SPM standard brain space (near MNI), and spatial smoothing using a Gaussian filter of 6 mm full width at half maximum (FWHM). Subsequently, images were skull-and-eye stripped and the mean of each time-slice was set to the same value (global scaling). A temporal high-pass filter with a cut-off of 512 s was applied, as well as a low-pass filter with the temporal properties of the hemodynamic response function (hrf), in order to reduce temporal acquisition noise.

The label time-series were obtained using two separate methods, using computer frame-by-frame analysis of the movie [14], and using subjective ratings averaged across an independent set of five human observers [11].

The computer-derived labels indicated luminance change over time (temporal contrast), visual motion energy (i.e. the fraction of temporal contrast that can be explained by motion in the movie). The human-derived labels indicated the intensity of subjectively experienced color, and the degree to which faces and human bodies were present in the movie. In prior studies, each of these labels had been shown to correlate with brain activity in particular and distinct sets of areas specialized to process the particular label in question [11, 14].

Evaluation Methodology

In order to evaluate the effect of semi-supervised Laplacian regularization on the performance of KCCA, we have evaluated three variants of the algorithm. In the first variant, we have run KCCA without any Laplacian regularization. This is achieved by setting $\gamma_x = \gamma_y = 0$. The second variant consists of Laplacian regularization where the empirical Laplacian matrix was computed using only data for which correspondences between $\mathcal{X}$ and $\mathcal{Y}$ were known. In the final variant, we used full semi-supervised Laplacian regularization, where the manifold was estimated using all available training data. We have not applied Laplacian regularization on the $\mathcal{Y}$ modality in any of the variants, though this may improve performance in that the statistical properties and dependencies of the different image variables may be better modeled. As we are primarily interested in the neuroscientific interpretation of f_x, we have chosen not to exploit these dependencies in this way.

We also evaluate the performance of the algorithms quantitatively. We have run five fold cross validation in which we hold out a portion of the data with correspondences at each fold. As KCCA attempts to maximize Pearson correlation, we first project the held out data using the learned regressors, and then measure their empirical correlation.

In all cases, we have used linear kernels on both the input and output spaces. This is so we may interpret the regressor, f_x, as a learned map of the brain regions implicated in various visual processing. The Laplacian matrix was computed using a Gaussian kernel with the bandwidth parameter set to the median distance between all pairs of training data (with and without correspondences). We have used the symmetric normalized Laplacian $\mathcal{L} = D^{-\frac{1}{2}}(D - W)D^{\frac{1}{2}}$, where D is the diagonal matrix whose entries are the row sums of the similarity matrix, W.

Table 4.1: Mean holdout correlations across the six variables in all experiments with five-fold cross-validation. Experiment 1 consists of KCCA using only data for which correspondences are known. Experiment 2 employs Laplacian regularization where the Laplacian matrix is estimated using only data for which correspondences are known. Finally, experiment 3 employs full semi-supervised Laplacian regularization. Semi-supervised Laplacian regluarization gives the best performance in all cases.

	Motion	Temporal Contrast	Human Body	Color	Faces	Language
Exp 1	-0.012 ± 0.081	0.042 ± 0.065	0.095 ± 0.086	-0.075 ± 0.069	0.173 ± 0.073	0.172 ± 0.070
Exp 2	0.065 ± 0.066	0.088 ± 0.084	0.274 ± 0.093	-0.002 ± 0.079	0.203 ± 0.075	0.231 ± 0.074
Exp 3	0.170 ± 0.074	0.116 ± 0.101	0.340 ± 0.043	0.128 ± 0.089	0.303 ± 0.054	0.365 ± 0.057

Model Selection

We have used two model selection criteria to optimize over the variables ε and γ. Both criteria are used as the inner loop of a grid search. In the first variant, we select the model parameters that maximize a five fold cross validation estimate of the empirical correlation (using only the training data). As this is both computationally and statistically inefficient, we have also evaluated a model selection criterion proposed in [82]. This consists of creating a random permutation of the correspondences and running the eigenproblem with the unpermuted data and with the permuted data. The parameter setting with the maximum norm of the difference of the spectra of the two eigenproblems is taken to be the optimum.

Results

The visual content of the stimulus is quantified in six variables: Motion, Temporal Contrast, Human Body, Color, Faces, and Language. We have repeatedly run all three variants of the experimental setup (Section 4.1.3) setting our output space to each individual variable. The results for the cross validation model selection are shown in Table 4.1, and the results for the spectral model selection are shown in Table 4.2. We have additionally run experiments with multi-variate output by grouping several of the variables into three groups: {Visual motion energy, Body, Color}; {Motion, Faces}; and {Motion, Visual motion energy, Color, Faces}. The results of these experiments using the spectral model selection are shown in Table 4.3.

As we have used linear kernels in all cases, we can interpret the output of the model by analyzing the weights assigned to different spatially localized brain regions. We show results for visual stimulus consisting of *Faces* in

Table 4.2: Mean holdout correlations across the six variables in all experiments with the spectral model selection criterion of [82]. Experiment 1 consists of KCCA using only data for which correspondences are known. Experiment 2 employs Laplacian regularization where the Laplacian matrix is estimated using only data for which correspondences are known. Finally, experiment 3 employs full semi-supervised Laplacian regularization. Semi-supervised Laplacian regluarization gives the best performance in all cases.

	Motion	Temporal Contrast	Human Body	Color	Faces	Language
Exp 1	-0.012 ± 0.081	0.042 ± 0.065	0.095 ± 0.086	-0.075 ± 0.069	0.173 ± 0.073	0.172 ± 0.070
Exp 2	0.065 ± 0.066	0.088 ± 0.084	0.274 ± 0.093	-0.002 ± 0.079	0.203 ± 0.075	0.231 ± 0.074
Exp 3	0.170 ± 0.074	0.116 ± 0.101	0.340 ± 0.043	0.128 ± 0.089	0.303 ± 0.054	0.365 ± 0.057

Table 4.3: Mean holdout correlations across the 3 multi-variate sets in all experiments with the spectral model selection criterion of [82]. Experiment 1 consists of KCCA using only data for which correspondences are known. Experiment 2 employs Laplacian regularization where the Laplacian matrix is estimated using only data for which correspondences are known. Finally, experiment 3 employs full semi-supervised Laplacian regularization. Semi-supervised Laplacian regluarization gives the best performance in all cases.

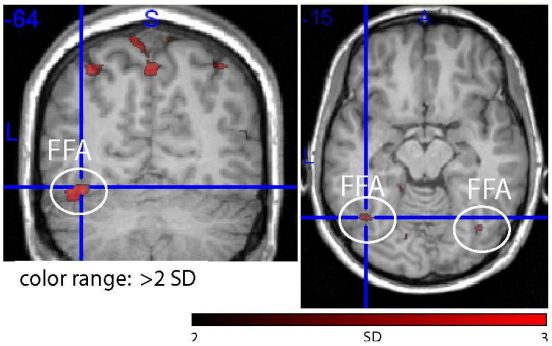
	Visual motion energy, Body, Color	Motion, Faces	Motion, Visual motion energy, Color, Faces
Experiment 1	0.1596 ± 0.0807	-0.0827 ± 0.0460	0.1167 ± 0.0785
Experiment 2	0.1873 ± 0.0879	0.0602 ± 0.0908	0.1498 ± 0.0827
Experiment 3	0.2844 ± 0.0716	0.1898 ± 0.0636	0.2528 ± 0.0579

Figure 4.1, *Human body* in Figure 4.2, *Color* in Figure 4.3, and *Motion* in Figure 4.4. In Figure 4.5 we show results from multivariate output consisting of *Motion* and *Faces*. We provide a neuroscientific evaluation in the next section.

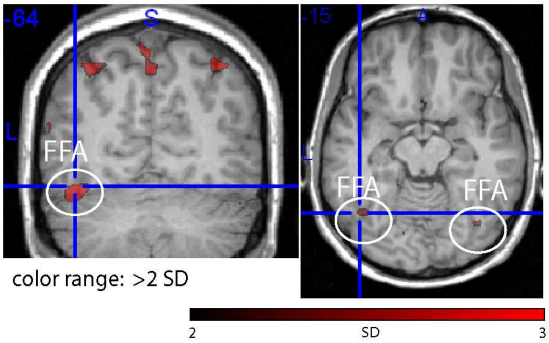
4.1.4 Discussion

We observe several trends in Tables 4.1, 4.2 and 4.3. First, our major hypotheses were confirmed: for every variate label, the performance improved with the Laplacian regularization on the labeled data, and performance was best in the semi-supervised condition. In the semi-supervised conditions (Experiment 3 as shown in Tables 4.1, 4.2 and 4.3) the additional data without correspondences is sufficiently close to the marginal distribution over $\mathcal{X}$ to improve results significantly, thus the additional data improves the results without any information about the correspondences of the data. Second, the model selection criterion worked very well; the criterion suggested by [82] performed equally as well as the five-fold cross-validation. Additionally, some variables can be better predicted than others, namely the presence of faces or human bodies in the viewing content, while some elicited relatively poorer performance in all experiments.

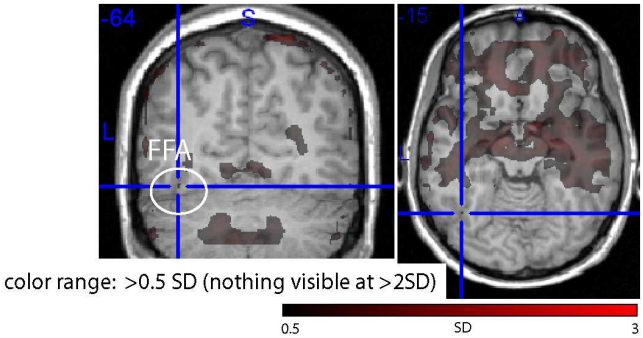
Figures 4.1 through 4.5 show slices taken through the anatomical image of one subject, with weight maps obtained from the different analyses of its functional data superimposed in red, wherein the maps were thresholded at 2 standard deviations in most cases, but had to be lowered in some cases to reveal any localized activity. We show examples of four of the single-variate labels for each of the three experiments, as well as one of the sets of multivariate experiments. In the multi-variate label example, we show the same weight map but at different brain volume coordinates in order to visualize the expected brain activations for each of the lables involved. The maps corresponding well to the known functional anatomy, and to activations obtained in the previous regression studies of free-movie-viewing data [11]. Faces obtained high weights in the fusiform cortex (fusiform face area, FFA) (Figure 4.1); Human Bodies dorso-lateral and ventral parts within the lateral occipital cortex (extrastriate body area (EBA) and fusiform body area (FBA)) (Figure 4.2); Color obtained high weights in the medial fusiform cortex where human V4 is located (Figure 4.3). The spatial layout of the weights thus corresponds well to the previous literature, and indicates that some of the analyses applied here yield results that are neuroscientifically meaningful and that can identify distinct cortical regions involved in the distinct tasks. Semi-supervised Laplacian regularization worked well in that



(a) Semi-supervised Laplacian regularized solution.

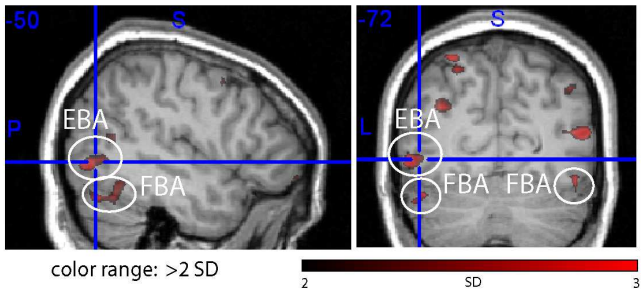


(b) Laplacian regularized solution.

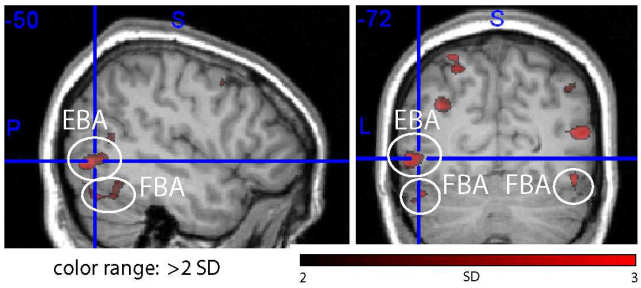


(c) KCCA without Laplacian regularization.

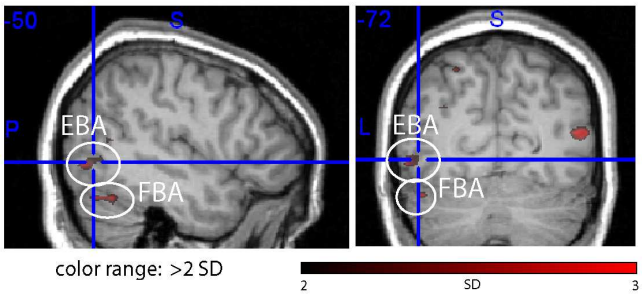
Figure 4.1: Faces: activation in the cortical region responsive to the visual perception of faces, the fusiform face area (FFA). Weight vectors are plotted over an anatomical image of the volunteers brain. Note that the semi-supervised Laplacian regularization led to the most specific and most significant weights in FFA.



(a) Semi-supervised Laplacian regularized solution.

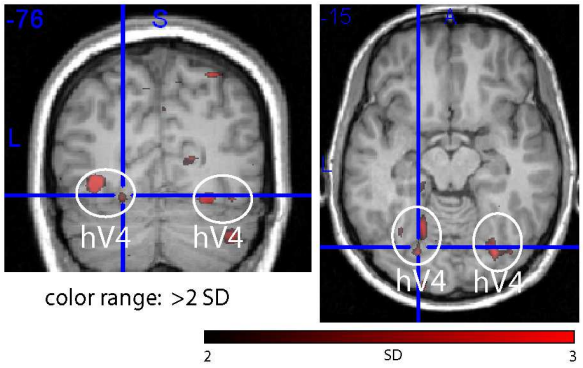


(b) Laplacian regularized solution.

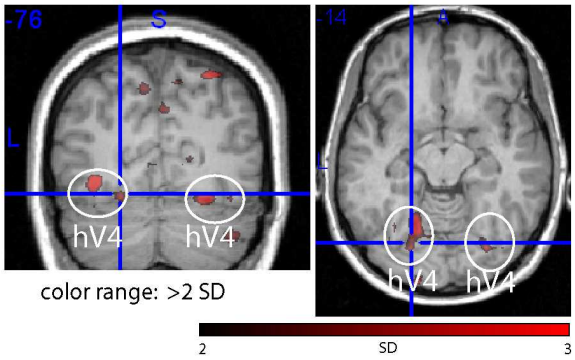


(c) KCCA without Laplacian regularization.

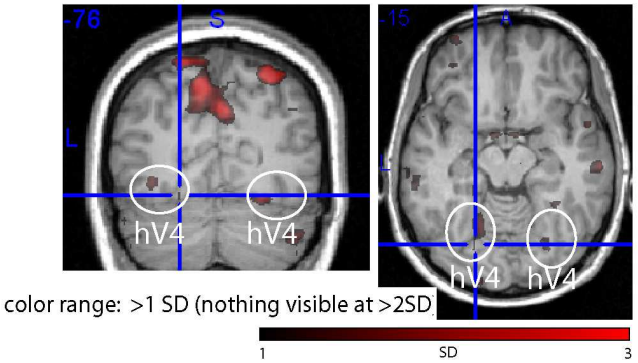
Figure 4.2: Human Body: activation in the cortical region responsive to the visual perception of human bodies, in the extrastriate body area (EBA) and in the fusiform body area (FBA). Same observation as in Figure 4.1.



(a) Semi-supervised Laplacian regularized solution.

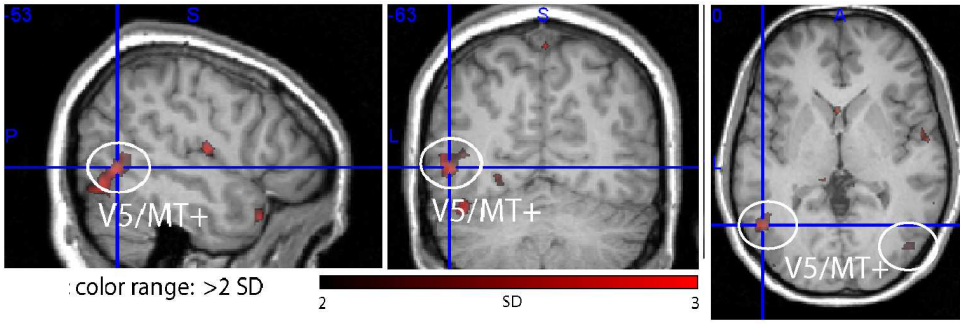


(b) Laplacian regularized solution.

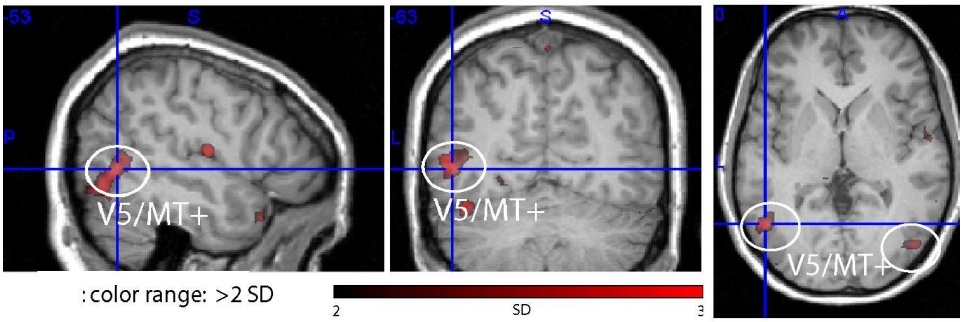


(c) KCCA without Laplacian regularization.

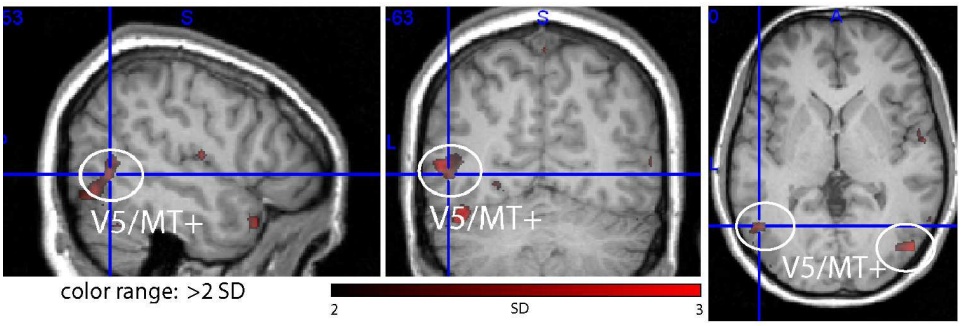
Figure 4.3: Color: activation in the color responsive cortex (human visual area 4, hV4). Same observation as in Figure 4.1.



(a) Semi-supervised Laplacian regularized solution.

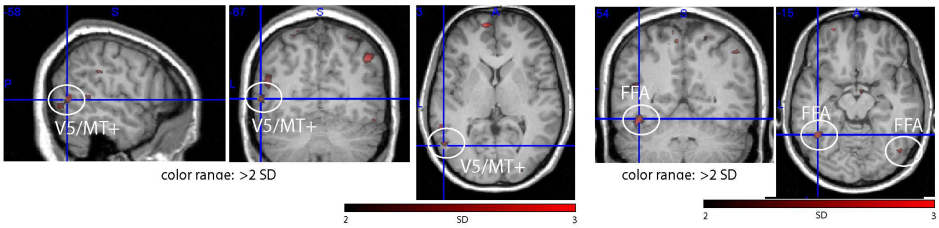


(b) Laplacian regularized solution.

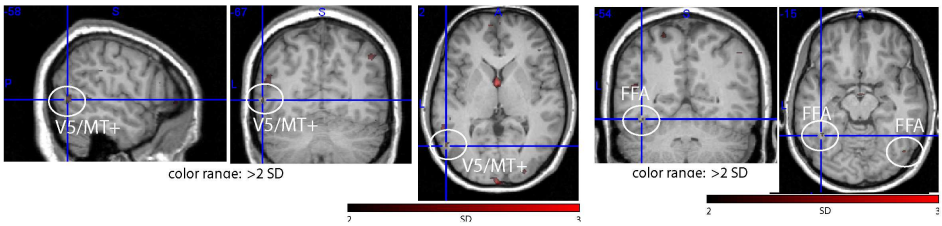


(c) KCCA without Laplacian regularization.

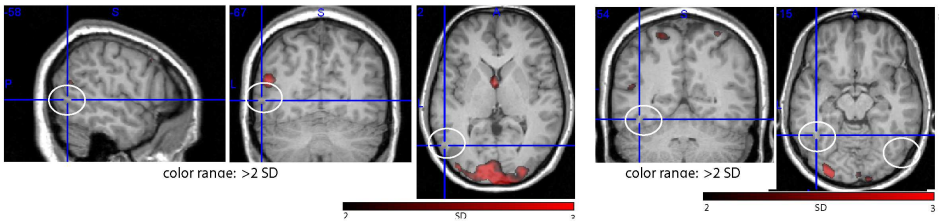
Figure 4.4: Motion: activation in the visual motion complex, area V5+/MT+. Same observation as in Figure 4.1.



(a) Semi-supervised Laplacian regularized solution.



(b) Laplacian regularized solution.



(c) KCCA without Laplacian regularization.

Figure 4.5: Multivariate - *Motion* and *Faces*: activations in the visual motion complex, area V5+/MT+ (left), and activation in the cortical region responsive to the visual perception of faces, the fusiform face area (FFA) (right). Same observation as in Figure 4.1.

weight maps thresholded at >2SD show relatively well defined activity of the regions previously shown to be involved with the features. For other analyses, e.g. KCCA without Laplacian regularization, we had to reduce the threshold to 0.5 or 1 (faces and color in the single-variate cases, respectively) to obtain activity in the areas in question, and the maps show additional, unspecific activity as well.

4.2 k-support Norm Regularization

This section is based on [GHS+13a, GDB+13, Bla13] and covers the application of the k-support norm as a regularizer for fMRI data, followed by its application to discrimination between neuromuscular dystrophies.

Functional magnetic resonance imaging (fMRI) is a wide spread modality within the field of neuroimaging, that measures brain activity by detecting associated changes in blood flow. The goal of fMRI data analysis is to detect correlations between brain activation and a task the subject performs during the scan. Many statistical methods have been proposed for analyzing fMRI data, including generalized linear model [14, 12], support vector machines [145], independent component analysis [11, 13] and kernel canonical correlation analysis [83, BSB+11]. All these methods have to deal with (a) data that lie in a high-dimensional space, with ten of thousands of voxels, (b) a small number of samples, due to the high cost and time consuming nature of the fMRI acquisition procedure, and (c) high levels of noise that arise from different sources, such as system noise and random neural activity.

Sparsity regularizers are key statistical methods for improving predictive performance in the event that the number of observations is substantially smaller than the dimensionality of the data, as is the case in fMRI analysis. The main methods considered here are the LASSO [151], the elastic net [176], and the k-support norm [5]. The former two are frequently applied sparsity regularizers developed in the statistics literature, while the latter is a recently introduced method that is mathematically related to the elastic net. The former two have previously been applied to fMRI analysis [37], while we are the first to apply the k-support norm to the best of our knowledge. We apply these methods to two different real data sets, the first consists of a healthy subject viewing a movie [11, 14, BSB09] while the second one consists of both cocaine addicted and healthy non-drug-using subjects performing a monetary reward task [75, 92]. Previous works that have explored sparsity regularization in fMRI are numerous and include [37, 133].

Table 4.4: A summary of the regularizers considered in this section.

Regularizer	$\Omega(w)$
LASSO [151]	$\lambda_1 \|w\|_1$
Elastic net [176]	$\lambda_1 \|w\|_1 + \lambda_2 \|w\|_2^2$
k-support [5]	$\lambda \|w\|_k^{sp}$ (see Equation (4.13))

4.2.1 Sparsity Regularization and the k-support Norm

Sparsity regularization is a key family of priors over linear functions that prevents overfitting, and aids interpretability of the resulting models [151, 176, 5, 37, 133]. Key to the mathematical understanding of sparsity regularizers is their interpretaion as convex relaxations to quantities involving the ℓ_0 norm, which simply counts the number of non-zero elements of a vector. Two of the most important sparsity regularizers, the LASSO [151] and the elastic net [176], are achieved by setting Ω to be the ℓ_1 norm of w or a linear combination of the ℓ_1 and squared ℓ_2 norms, respectively (Table 4.4). The elastic net has been employed in situations where there may be multiple correlated signals that should be combined to improve prediction accuracy, a case where the LASSO would yield a higher variance predictor.

While the LASSO can be interpreted as employing the convex hull of the ℓ_0 sparsity regularizer, the elastic net is looser than the convex hull of a norm that combines ℓ_2 regularization with sparsity [5]. However, one may employ the k-support norm, which is exactly the convex hull of that hybrid norm. The k-support norm can be computed as

$$\|w\|_k^{sp} = \left(\sum_{i=1}^{k-r-1} (|w|_i^{\downarrow})^2 + \frac{1}{r+1} \left(\sum_{i=k-r}^{d} |w|_i^{\downarrow} \right)^2 \right)^{\frac{1}{2}} \tag{4.13}$$

where $|w|_i^{\downarrow}$ is the ith largest element of the vector and r is the unique integer in $\{0, \ldots, k-1\}$ satisfying

$$|w|_{k-r-1}^{\downarrow} > \frac{1}{r+1} \sum_{i=k-r}^{d} |w|_i^{\downarrow} \geq |w|_{k-r}^{\downarrow}. \tag{4.14}$$

The k-support norm is closely related to the elastic net, in that it can be bounded to within a constant factor of the elastic net, but leads to slightly different sparsity patterns. One can see from Equation (4.13) that the norm trades off a squared ℓ_2 penalty for the largest components with an ℓ_1 penalty

for the smallest components. While initial experiments have shown promising results with the k-support norm for a range of machine learning problems [5], to the best of our knowledge this study is the first to apply the approach to fMRI.

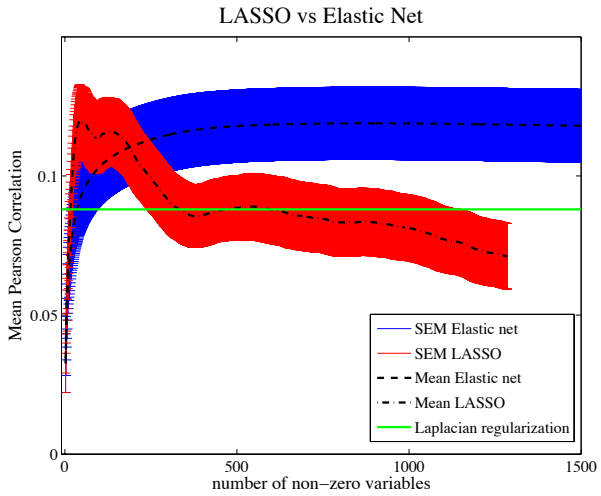
4.2.2 fMRI Analysis of Cocaine Addiction

The neuropsychological experiment for cocaine addiction data set has a block design, that included six sessions, with each of them having different conditions. The two varying conditions are the monetary reward (50¢, 25¢ and 0¢) and the cue shown (drug words, neutral words). The session consists of an initial screen displaying the monetary reward and then presenting a sequence of forty words in four different colors (yellow, blue, red or green). The subject was instructed to press one of four buttons matching the color of the word they had just read. The subjects were rewarded for correct performance depending on the monetary condition. In this paper, we focus on the monetary conditions only, and more specifically the session of 50¢ following [92]. The dataset consists of 16 cocaine addicted individuals and 17 control subjects. These were the subjects that complied to the following requirements: motion $<$ 2mm translation, $< 2^\circ$ rotation and at least 50% performance of the subject in an unrelated task [75]. For each subject a contrast map was computed using the statistical parametric mapping package SPM2 (`http://www.fil.ion.ucl.ac.uk/spm/`).

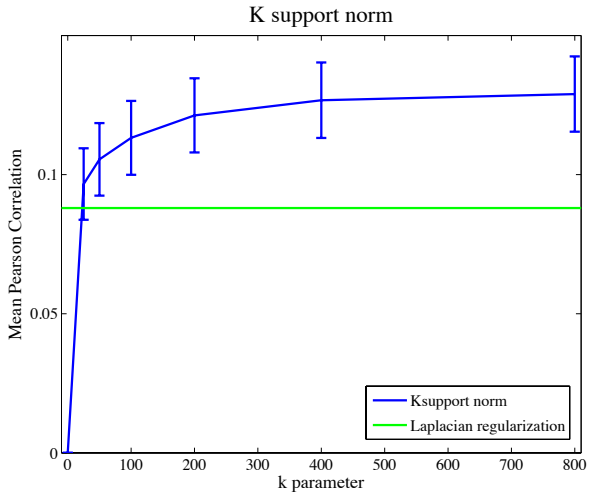
4.2.3 Results

Results are presented on two fMRI datasets. The first consists of a healthy subject in a free-viewing setting. Data collection was previously described in [11, 14], while the pre-processing followed [BSB09]. The discriminative task is the prediction of a "Temporal Contrast" variable computed from the content of a movie presented to the subject [BSB$^+$11]. This dataset was employed for the quantitative evaluation due to its larger sample size. The second dataset consists of control and cocaine addicted subjects [75, 92].

The performance of the different sparse regularization techniques, shown in Figure 4.6, is evaluated as the mean correlation over 100 trials of random permutation of the data described in [BSB09]. In each trial, 80% of the data are used to train the method, while the remaining 20% are used to evaluate the performance. More specifically, Figure 4.6(a) shows the mean correlation between LASSO and elastic net against the number of non-zero variables (i.e voxels), while Figure 4.6(b) shows the mean correlation for the k-support



(a) LASSO vs Elastic net



(b) k-support norm

Figure 4.6: Mean Pearson correlations between the label and prediction on the hold-out data over 100 trials for the dataset described in [BSB09] (higher values indicate better performance). Error bars show the standard deviation. The LASSO achieves its best performance with a sparsity level substantially lower than the elastic net, as it suppresses correlated voxels (Figure 4.6(a)). The k-support norm performs better than the LASSO, elastic net, or Laplacian regularization reported in [BSB$^+$11], and is a promising candidate for sparsity in fMRI analysis (Figure 4.6(b)). (Figure best viewed in color.)

norm against different k values–which are correlated with the number of non-zero coefficients. LASSO achieves a maximum mean correlation of 0.1198 for 44 non-zero variables, elastic net a maximum mean correlation of 0.1189 for 866 non-zero variables, while k-support norm a maximum of 0.129 for $k = 800$. This is substantially higher than was previously reported in [BSB+11].

We have additionally visualized the brain regions predicted when applying the LASSO and the k-support norm to the data from [75, 92]. For each, we have selected slices through the brain that maximize the sum of the absolute values of the weights predicted by the respective methods. These results are presented in Figure 4.7.

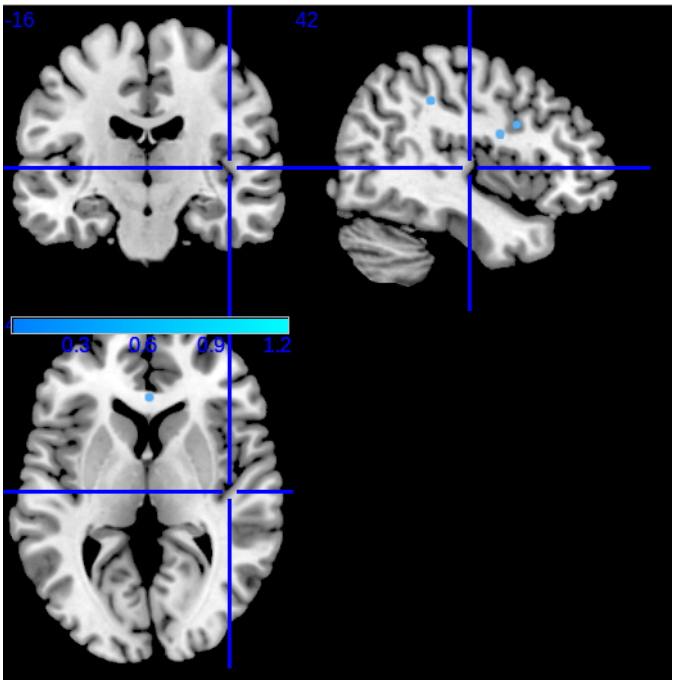
The main area of activity shown in Figure 4.7(b) is the rostral anterior cingulate cortex (rostral ACC). It has been shown to be deactivated during the drug Stroop as compared to baseline in cocaine users vs. controls even when performance, task interest and engagement are matched between the groups [75] and that its activity is normalized by oral methylphenidate [76]–which similarly to cocaine blocks the dopamine transporters increasing extracellular dopamine–an increase that was associated with lower task-related impulsivity (errors of commission). This region was responsive (showed reduction in drug cue reactivity) to pharmacotherapeutic interventions in cigarette smokers [45, 59], and may be a marker of treatment response in other psychopathology (e.g., depression). The LASSO does not show a meaningful sparsity pattern (Figure 4.7(a)).

4.3 Graph Kernels

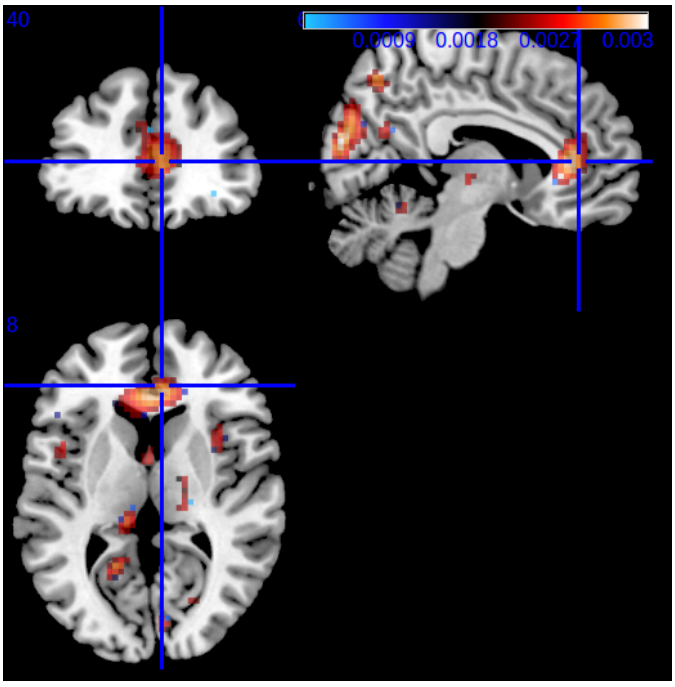
This section is based in part on [GHS+13b] and contains material previously included in the thesis of Katerina Gkirtzou, a doctoral student that I supervised [73].

Graphs are a powerful and natural way to represent complex data with integrated structure. Graphs have been used in numerous applications in a number of different fields, such as (i) computer vision and biomedical imaging analysis, (ii) bioinformatics, (iii) social networks analysis and (iv) chemoinformatics. In many applications, the exploration of the data requires the ability to efficiently compare graphs and to provide a similarity measurement, a problem known as *graph comparison*.

A first approach towards this problem is to quantify whether two graphs are identical, i.e. isomorphic. This leads to a binary similarity measure, which equals to 1 when the two graphs are isomorphic, otherwise it equals to 0. Although this idea is intuitive no efficient algorithm is known, as



(a) LASSO



(b) k-support norm

Figure 4.7: A visualization of the areas of the brain selected by the LASSO and by the k-support norm applied to the data described in [75]. The LASSO leads to overly sparse solutions that do not lend themselves to easy interpretation (Figure 4.7(a)), while the k-support norm does not suppress correlated voxels, leading to interpretable and robust solutions (Figure 4.7(b)).

neither a proof of NP-completeness nor membership in the class of polynomial time problems are known for the graph isomorphism problem [69, Chapter 7]. Other similarity measures are based on concepts related to isomorphism, such as subgraph isomorphism or the largest common subgraph. Subgraph isomorphism is analogous to graph isomorphism but it could be used also when two graphs have different sizes. Unlike, the graph isomorphism problem, the subgraph isomorphism problem has been proven to be NP-complete [69, Section 3.2.1]. A similarity measure can also be defined based on the size of the largest common subgraph in two graphs. Unfortunately, this problem is also known to be NP-hard [69, Section 3.3].

Despite being intuitive, these approaches suffer from intractable computational time. Another family of approaches, *graph kernels*, have been found to be useful across a wide range of applications in recent studies. They tackle both the problem of graph representation and graph comparison through the exploitation of the graph topology by decomposing the graph into substructures and aggregating statistics over these substructures. This strategy considers a measure of similarity between the graphs as a form of inner product. Graph kernels are commonly derived as an instance of the family of the R-convolution kernels [86], which are a generic way of constructing kernels of complex objects by decomposing them into discrete structures and comparing all pairs of decompositions. Every new decomposition would yield a new kernel. A first approach would be to decompose the graphs into all possible subgraphs. However, calculating all subgraphs is at least as hard as deciding whether two graphs are isomorphic [70]. So it is necessary to limit the decomposition of the graphs only into specific types of subgraphs that are computable in polynomial time [161, 143].

Although efficient graph kernels have been developed that have good performance on discretely labeled graphs, the literature on continuously or vector labeled graphs is still relatively undeveloped. We help to address this gap in the literature by proposing a framework for kernel construction that converts continuous labeled graphs into a sequence of discretely labeled graphs using a pyramid quantization strategy.

We define a graph as a triplet $G = (V, E, \mathcal{L})$, where V is the vertex set, E is the edge set and $\mathcal{L} : V \mapsto \Sigma$ is a function assigning a label from an alphabet Σ to each vertex in the graph. The neighborhood $N(v) = \{v' | (v, v') \in E\}$ of a vertex v is the set of all vertices adjacent to v, i.e. all vertices connected with a single edge. The degree $d(v)$ of a vertex v is the number of edges incident with v. Every graph has at most v vertices, e edges and a maximum degree of d. A walk in a graph is a sequence of adjacent vertices. A path is a walk that contains only distinct vertices, while a cycle is a closed walk. A

rooted tree is an acyclic graph with a specified root vertex. A subtree is a connected subset of distinct vertices that contains no cycles. The height of a rooted tree or subtree is the maximum distance between the designated root vertex and any other vertex in the tree or subtree respectively. Subtree patterns are labeled trees extracted from a labeled graph G for a given depth h and a given vertex v. Repetition of the same vertex is allowed in subtree pattern, but it is treated as distinct vertices, allowing a cyle-free pattern.

4.3.1 The Weisfeiler-Lehman test of isomorphism

Our proposed algorithm uses the discretly labeled subtree pattern features introduced by the Weisfeiler-Lehman kernel [143], which exploits the key concepts from the one dimensional variant of the Weisfeiler-Lehman test of isomorphism [171].

A key feature of the Weisfeiler-Lehman algorithm is its fast runtime, $\mathcal{O}(he)$ where h is the maximum number of iterations of the test (effectively a chosen parameter), and e the maximum number of edges [143].

4.3.2 The pyramid quantization strategy

The Weisfeiler-Lehman algorithm is efficient precisely because it makes use of a discrete labeling over nodes, which enables an efficient hashing scheme in order to scale linearly in the number of edges and in the height of subtree patterns. A problem occurs when extending this method to continuous labeled graphs: we no longer have a notion of an exact match of a discrete label, and a hash function that counts approximate matches would implicitly define a single quantization of the vector space to a discrete set of labels. A single quantization is inexact, and gives only a weak relationship to the potentially rich geometry of the original label space. It is also not clear what the resolution of the quantization should be to maximize performance. To overcome this, we propose a pyramid quantization strategy similar to the one used by [78, 77] to determine a logarithmic number of discrete labelings with increasing granularity for which we run the Weisfeiler-Lehman algorithm. In other words, we approximate a graph representation with continuous valued labels as a sequence of graphs with discrete labels of increasing granularity.

Given a vector labeled graph $G = (V, E, \mathcal{L})$, where $\mathcal{L} : V \mapsto \mathbb{R}^d$ is the function assigning a d-dimensional vector label to each vertex, we want to derive a hierarchical decomposition of $\mathbb{R}^d$ as multi-resolution quantizations. The multi-resolution quantizations will then be used to determine the discrete labeling of increasing granularity. This can be expressed

as a two step process, first we construct a set of quantization functions $Q^{(l)} : \mathbb{R}^d \mapsto \Sigma_0^{(l)}, 0 \leq l \leq L$ that will encode the continuous labels into a quantization of a given resolution $|\Sigma_0^{(l)}| = 2^l$. The quantization function $Q^{(l)}$ is generated for $l \in \{0, \ldots, L\}$ to determine multi-resolutions of increasing granularity, where $L = \lceil \log_2 D \rceil$, $D \leq |V| = v$ is the number of unique values in the image of the vertex set V under the label function $\mathcal{L}$. Note that the single quantization bin for $Q^{(0)}$ is big enough so that all vertices receive the same discrete label, while as the quantization resolution moves from coarser to finer, we end up with $Q^{(L)}$ that contains quantization bins that are small enough so each unique data point from the image of the set V under $\mathcal{L}$ falls into its own quantization bin. To achieve this hierarchical quantization in the experiments performed here, we have uses an agglomerative hierarchical clustering with Ward's minimum variance method [168].

The second step is to compose the quantization function $Q^{(l)}$ with the labeling function $\mathcal{L}$, $\forall l \in \{0, \ldots, L\}$, so we can approximate our initial vector labeled graph G as a sequence of graphs with discrete labels of increasing granularity:

$$\begin{aligned} G = (V, E, \mathcal{L}) \overset{Q^{(l)} \circ \mathcal{L}}{\mapsto} & \left(G^{(0)}, \ldots, G^{(L)} \right) \\ & = \left((V, E, \mathcal{L}^{(0)}), \ldots, (V, E, \mathcal{L}^{(L)}) \right), \end{aligned} \tag{4.15}$$

where $\mathcal{L}^{(l)} : V \mapsto \Sigma_0^{(l)}$ is defined to be $Q^{(l)} \circ \mathcal{L}$, and $\Sigma_0^{(l)}$ is the discrete label alphabet for a given level l of quantization. Note that the topology of the graph does not change in the sequence of graphs, only the continuous vector labels are discretized.

We note that quantization schemes of this type, when paired with a histogram intersection kernel and an appropriately weighted linear combination of kernel values across quantization levels, results in a multiplicative error bound on the optimal graph matching [78, Proposition 3]. We may therefore interpret the pyramid quantized Weisfeiler-Lehman graph representation as a function space that enables tight approximations to cost of the optimal matching over vector representations of subtree patterns.[1]

4.3.3 Cocaine addiction dataset

We evaluate the approach on a dataset [75, 92, GHS$^+$13a, GHS$^+$13b] that contains an approximately equal number of cocaine addicted individuals and

[1] In practice, we do not use the fixed weighting across quantization levels proposed by Grauman and Darrell [78], but instead discriminatively optimize over the function space.

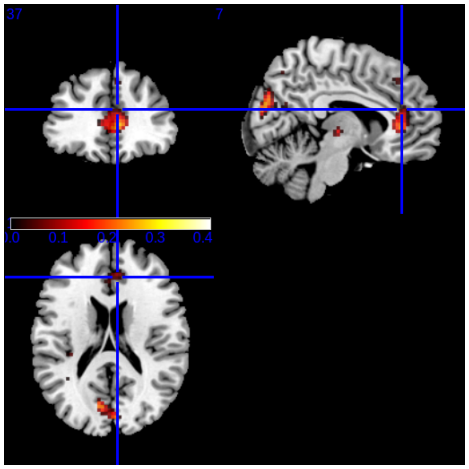
control subjects performing a neuropsychological experiment of block design, called a drug Stroop experiment. The classification task is to discriminate cocaine from control subjects. The data were preprocessed using statistical parametric mapping SPM2 [63] and a contrast map for each subject was produced. Only the subjects that complied to motion < 2mm translation, $< 2^\circ$ rotation and at least 50% performance of the subject in an unrelated task [75] were kept.

4.3.4 Graph construction

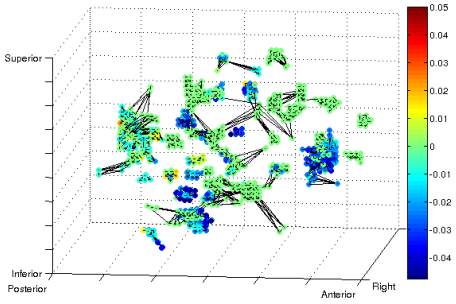
As our statistical estimator, we have made use of the Elastic Net [176]. This method is particularly appropriate in fMRI where nearby voxels are likely to be correlated, and regions responsible for a given function or behavior distributed across multiple voxels. Furthermore, it is typical that the majority of voxels in the brain are not discriminative of a specific output. We make use of the Elastic Net twice in our learning pipeline. In the first instance, we use the Elastic Net on the raw voxel values to determine a subset of voxels on which we build a graph representation. Our model selection step has typically chosen approximately 10^3 voxels for this stage. We subsequently compute subgraph statistics over this graph to generate a feature vector, $\phi^{(l)}_{(h)}(G^{(l)})$ for a given height h of subtree patterns and a given quantization level l for a graph G. Finally, we use the Elastic Net on these subgraph statistics over all quantization level, in order to determine our final prediction function, with a model selection step to determine appropriate values for λ_1 and λ_2.

To construct the graph representation, we have made use of k-nearest neighbor graphs on the voxels that were selected by an initial training of the Elastic Net. We symmetrize the k-nn relationship by considering the edges to indicate an undirected graph structure. While other models of connectivity are of interest [146, 169], we have found that the use of k-nearest neighbors to determine the graph topology yields good performance in general and illustrates the advantages of the pyramid Weisfeiler-Lehman approach. Furthermore, the subtree statistics considered here implicitly account for longer distance connections for sufficiently deep subtree patterns. We set the number of neighbors $k = 5$ in all experiments.

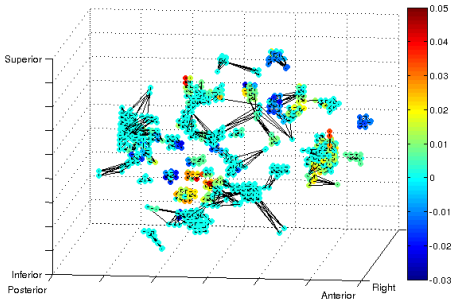
To enrich our graph representations of the fMRI contrast maps, we take advantage of the activation information. At each voxel selected by the Elastic Net for the construction of the graph, we label it with its activation. Since the activation has continuous values, our graph representation is transformed to a continuous labeled graph. Finally, since the initial fMRI



(a) A visualization of the areas of the brain selected by Elastic Net.



(b) Weisfeiler-Lehman - Control



(c) Weisfeiler-Lehman - Cocaine

Figure 4.8: A visualization of the areas of the brain selected by Elastic Net as well as a visualization of the learned functions on the quantized Weisfeiler-Lehman representation. The selected regions correspond to areas previously implicated as being related to addiction [75].

contrast maps are now represented as graphs with continuous labels on the vertices, we quantized the continuous activation labels and run the efficient Weisfeiler-Lehman algorithm in each quantization level to aggregate statistics of subtree patterns of different depth h.

4.3.5 Results

We use the same experimental setup, a random splitting scheme with 50 trials, to estimate the classification performance of *pyramid quantized Weisfeiler-Lehman graph representation* and the baseline method on the cocaine addiction dataset. In each trial, a random selection of 80% of the data are used for training, while the remaining 20% are used to estimate the performance.

Table 4.5: Mean accuracy over the hold-out data of 50 trials of the *pyramid quantized Weisfeiler-Lehman graph representation* for four different subtree pattern depths, $h \in \{0, 1, 2, 3\}$. Maximum performance is achieved with subtree patterns up to depth two.

Pyramid Quantized Weifeiler-Lehman				
h	**0**	**1**	**2**	**3**
Accuracy	54.00%	57.14%	**64.28%**	63.42%

In Table 4.5 we show the performance of the *pyramid quantized Weisfeiler-Lehman graph representation* for four different depths of subtree patterns, while Figure 4.8 shows a visualization of the learned function. Our approach achieves a mean accuracy of 64.28% for subtree patterns up to depth two, a significant improvement over the bag of words kernel ($h = 0$). We also compare our proposed technique with three other methods on the same dataset: (i) Gaussian kernel ridge regression, (ii) the Elastic Net with raw voxels as features, and (iii) the Elastic Net with raw voxels and *pyramid quantized Weisfeiler-Lehman* subtree features concatenated in a joint feature vector. In Figure 4.9 we show the mean accuracy of the final system and the standard error. *Pyramid quantized Weisfeiler-Lehman graph representation* outperforms the rest of the methods. With a Wilcoxon signed rank test between the Elastic Net with raw voxels (the best performing baseline system) and the *pyramid quantized Weisfeiler-Lehman graph representation* we determine that our proposed method is statistically significantly better ($p = 0.02$). Additionally, a reduction of over 14% in classification error is recorded between the Elastic Net on the raw voxels and our method.

Figure 4.8(a) shows the areas selected by the Elastic Net, while Figure 4.8(b) and Figure 4.8(c) show the visualizations of the learned functions for control and cocaine addicted subjects, respectively. Note that Elastic Net on the raw voxels selected the rostral anterior cingulate cortex (rostral ACC), an important region for addictive behavior [GHS$^+$13a, GHS$^+$13b].

Although our method works in an implicitly high dimensional space, we empirically observe that Elastic Net regularization controls the complexity at each stage of the pipeline. The first learning step selects approximately 1100 voxels. Using the *pyramid quantized Weifeiler-Lehman graph representation*, we generate a feature vector of length 6×10^5, but with a sparsity of $\sim 2\%$. The second application of Elastic Net selects only $\sim$ 2K dimensions.

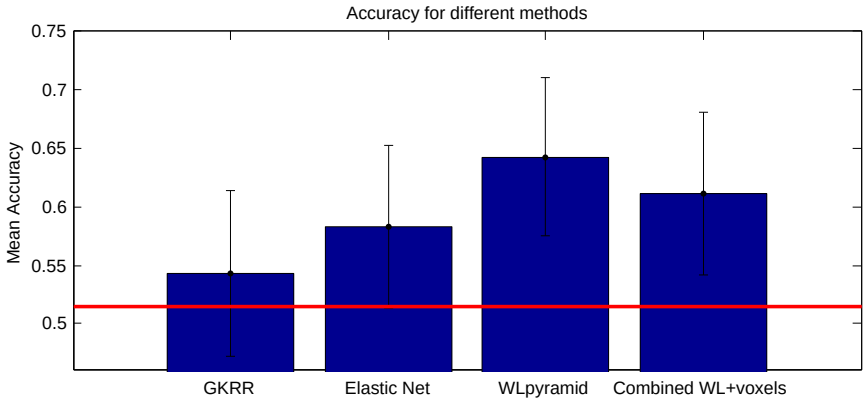


Figure 4.9: Mean accuracy and standard error on the cocaine addiction dataset. The compared methods are (left to right) Gaussian kernel ridge regression (GKRR), the Elastic Net on raw voxels, *pyramid quantized Weisfeiler-Lehman* (WLpyramid), and the Elastic Net with a concatenation of the raw voxels and the *pyramid quantized Weisfeiler-Lehman* features (Combined EN+WL). The horizontal red line indicates chance performance. The *pyramid quantized Weisfeiler-Lehman* features perform better than Gaussian kernel ridge regression and the Elastic Net on raw voxels with statistical significance.

In each step, the method retains complexity much lower than a "simple" linear function over tens of thousands of voxels as has been proposed in previous works.

Several broad observations are apparent from our quantitative results. From Table 4.5, we note that subtree patterns up to depth two seem to perform best, and that deeper subtree patterns begin to reduce average performance. This indicates that the big-$\mathcal{O}$ complexity of the graph representation is only slightly higher than using a simple linear function. The proposed method performs significantly better than the Gaussian kernel ridge regression and the Elastic Net baselines (see Table 4.5 and Figure 4.9). In our final experiment of combining the raw voxel values with the subtree pattern features, we found that performance decreased slightly from that of only considering subtree pattern features.

4.4 Discussion

In this chapter, we have discussed three contributions unified by their relationship to regularization and function classes, as well as their application to fMRI analysis. In Section 4.1 we discussed the application of semi-supervised Laplacian regularization to kernel canonical correlation analysis. Subsequently, in Section 4.2, we demonstrated the novel application of the k-support norm to the problem of fMRI analysis. Finally, we developed a pyramid quantization strategy for adapting kernels on discretely labeled graphs to the case of continuous labeled graphs in Section 4.3. In the next chapter, we discuss our contributions to representation and inference in the regularized risk framework.

Chapter 5

Representation and Inference

Inference is an essential step to achieving good prediction with a structured output model. Not all models are tractable, and the balance between an expressive yet tractable model whose parameters can be learned is the essence of modeling in this setting. The selection of an expressive model class that maintains efficient inference can result in high quality results, e.g. as demonstrated in [OB14] (Figure 5.1). Our contributions to efficient inference strategies include [BL09, Bla11, FB12, BZG13, OB14]. In this chapter, we highlight in particular contributions in branch-and-bound inference for object detection, and efficient inference in taxonomic prediction by the exploitation of computational advantages of a tensor product decomposition of the joint feature map.

5.1 Branch-and-Bound for Object/ROI Detection

One of the results of my doctoral research was the development of a branch-and-bound framework for object detection with bounding boxes [27, 113, 114, 24]. Subsequent to that work, I have extended this framework in several directions, including the incorporation of context and the efficient detection of multiple object instances [BL09, Bla11]. In this section, I focus on this latter aspect, which was presented in [Bla11].

Non-maximal suppression has been employed in many settings in vision and image processing. In image processing, objectives for edge and corner detection have been specified in terms of the eigenvalues of a matrix containing local oriented image statistics [84], while more recently general objectives for object detection have been trained discriminatively [162, 47, 115, 29, BL09, BVZ10, 67]. Often, an objective function specifies a property

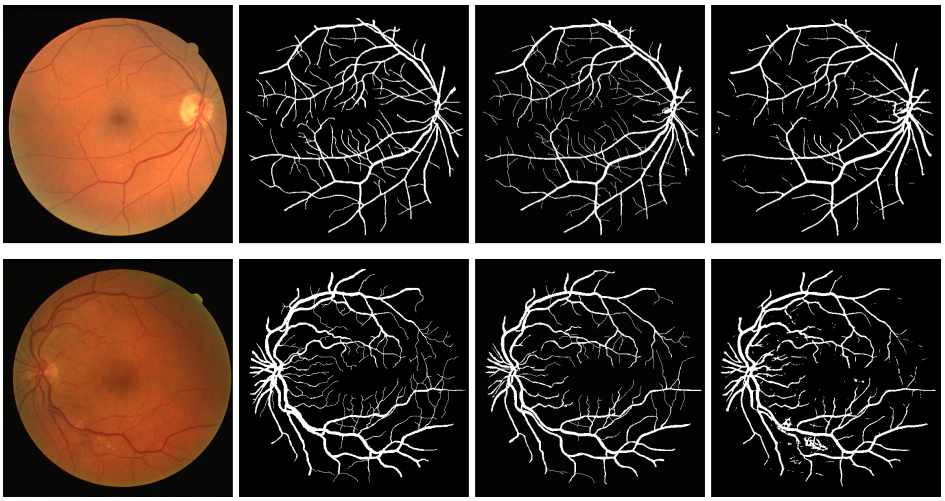


Figure 5.1: Examples of results obtained on healthy (top) and pathological (bottom) images from the benchmark DRIVE dataset [147]. From left to right: original images, ground truth labelings, 2nd human observer labelings, the segmentations achieved using the method described in [OB14].

of interest in image coordinates, but it is the arg maximum of the objective rather than scalar values that is of importance. From this perspective, an ideal objective would place all its mass on the true location and give zero output elsewhere. In practice, this is rarely the case, and the function output consists instead of hills and valleys characterizing intermediate belief in the fitness of a given location. Discriminative training of detection models can lead to the need for non-maximal suppression as more confident detections will have higher peaks than less confident ones. Without non-maximal suppression the next best-scoring detections will almost certainly be located on the upper slope of the peak corresponding with the most confident detection, while other peaks may be ignored entirely. One may interpret this as maximizing the log-likelihood of the detections assuming that they are independent, while in fact there is a strong spatial dependence on the scores of the output.

Here, we interpret commonly applied non-maximal suppression strategies as the maximization of a random-field model in which energies describing the joint distribution of detections are included. This insight enables us to characterize in general terms the maximization problem, and to make use of existing theoretical results on maximizing submodular (minimizing supermodular) functions. As a result, we can adopt an efficient optimiza-

tion strategy with strong approximation guarantees. This is of particular interest as maximizing a submodular function is in general NP-hard. The resulting optimization problem can be solved by a series of inter-related optimizations. Here, we follow Lampert et al. and approach the optimization using a branch-and-bound strategy that enables fast detections of typically tens of milliseconds on a standard desktop machine [113].

The branch-and-bound strategy we consider here is a best first search that makes use of a priority queue to manage which regions of the space of detections to explore. Furthermore, the inter-related optimizations resulting from branch-and-bound have a very benign structure in that each problem can use intermediate results stored in the priority queue by the previous optimization. We show empirically that, while reuse of these results does not always give an optimal increase in speed, that there is a very simple strategy for the selective reuse of intermediate results that does give optimal empirical performance. This is further illuminated by several theoretical results that motivate the strategy.

5.1.1 Related Work

Viola and Jones developed one of the best studied and widely used generic detection algorithms [162]. A key step in their algorithm can be interpreted as non-maximal suppression, in which they cluster highly overlapping detections and represent clusters by only one detection. Thus, peaks in the detection landscape are compressed to a single detection, suppressing other output.

A key question in such strategies is which metric to use when suppressing detections that are too close. A common approach in the recent object detection literature (e.g. [56, 158, 159]) is to make use of a detection specific overlap measure, such as the one used in the PASCAL VOC object detection challenge [52]. It has been noted that this overlap measure has several favorable properties compared to other measure such as invariance to scale and translation [88].

Desai et al. have taken an interesting approach in which the joint distribution between object detections is modeled linearly given features capturing statistics of the joint distribution of objects [49]. The model is trained discriminatively, but without approximation guarantees due to the greedy optimization employed in a cutting plane training algorithm. Their subproblem shares key characteristics with our random field characterization of non-maximal suppression, and the explicit characterization of a tractable family of models is a key contribution of the work described in this section.

The approaches cited above largely work by employing sliding windows or other window subsampling strategies, but alternatively, variants on Hough transform detections have also been used. Leibe et al. proposed a widely adopted model in which visual words vote for an object center [120]. Gall and Lempitsky have developed a state of the art detection framework using Hough forests [67]. Lehmann et al. have presented a line of work that extends these models to efficient detection [118, 119] where the second citation uses branch-and-bound for optimization of detection. The present work in contrast is agnostic to the exact model employed, and the branch-and-bound framework we employ has been applied to several variants of non-linear models that cannot be represented using Hough transforms [114].

Barinova et al. have proposed a principled method of non-maximal suppression that can be interpreted as an explicit approximation to a full probabilistic model [10]. Their work is to our knowledge the first to couple approximation results for the maximization of submodular functions with object detection. Their work, however, is (i) restricted to models for which one can build a Hough image whereas the class of functions for which we can design a practical bound is more general, and (ii) their approach is restricted to very low dimensional detection parametrizations because Hough images are expensive to build for more than a few dimensions. Such an approach additionally must recompute a Hough image after each detection, while the proposed non-maximal suppression model can reuse the same data-structures (such as integral images [162, 114]) for subsequent detections.

Maximization of a submodular function with monotonic properties is common to many problems in computer science, from robotics [91] to social network analysis [100] and sensor networks [80, 108], and has been studied extensively in the operations research literature (a toolbox by Andreas Krause contains many of the algorithms developed there [107]). Branch and bound has been employed to find optimal solutions to the (in general) NP-hard problem [74], but has not, to our knowledge, been applied to greedy optimization of supermodular functions with optimal approximation guarantees, as in this work. The variety of problems that share the same structure promises that analogous optimization approaches to that proposed in this work may have wider application across computer science domains.

5.1.2 The Energy

We consider a very general class of joint energy functions that contains both an appearance model of the object class of interest, as well as terms incorporating beliefs about the joint distribution of object detections. These

latter terms may be the result of a learning procedure, a prior over the joint positions of objects [49], or a set of constraints chosen *a priori* to disallow detections that have high overlap. We consider energies of the form

$$\max_y \sum_i \langle f, \phi(x, y_i) \rangle_{\mathcal{H}} - \mathbf{\Omega}(y). \tag{5.1}$$

Here we consider $\mathbf{\Omega}$ that factorizes into pairwise terms as well as higher order terms

$$\mathbf{\Omega}(y) = \sum_{ij} \Omega(y_i, y_j) + \underbrace{\sum_{c \in \mathcal{C}} \Omega_c(y_c)}_{\text{higher order terms}} \tag{5.2}$$

where x is an image, y_i is an object detection,[1] y is a collection of detections, ϕ is a joint kernel map, f is a function living in the RKHS defined by ϕ, Ω is a penalization term for detections that overlap too closely, and $c \in \mathcal{C}$ is a clique in the set of cliques contributing to the energy. In principle, higher order terms that are supermodular (see Section 5.1.3) do not affect the anaylsis in this paper. For simplicity, we will not treat them explicitly in the sequel.

We note that this form of energy for the detection of multiple objects may occur in diverse settings, such as object detection test time inference, detection cascades, and inference for cutting plane training of structured output learning [29, 97].

5.1.3 Minimization of a Supermodular Function

Many optimization approaches to random field models, such as graph cuts, rely on the submodularity of a function to be minimized. In the context of image segmentation, this is reflected in a general principle that neighboring pixels are likely to share the same label. Non-maximal suppression, however, enforces the exact opposite effect: neighboring detections are likely to have different labels, at least when the appearance term indicates an object is likely to be present in the vicinity.

In particular Equation (5.1) is the maximization of a submodular (minimization of a supermodular) function. Submodularity holds for a set function if for any two subsets of detections, A and B such that

$$A \subset B \tag{5.3}$$

[1] In the sequel we pay particular attention to detections parametrized by bounding boxes.

the following holds

$$f(A \cup \{y\}) - f(A) \geq f(B \cup \{y\}) - f(B). \tag{5.4}$$

This is easy to show as

$$\begin{aligned} f(A \cup \{y\}) - f(A) &= \langle f, \phi(x,y) \rangle_{\mathcal{H}} - \sum_{i \in A} \Omega(y_i, y) && (5.5) \\ \langle f, \phi(x,y) \rangle_{\mathcal{H}} - \sum_{i \in A} \Omega(y_i, y) &\geq \langle f, \phi(x,y) \rangle_{\mathcal{H}} - \sum_{i \in B} \Omega(y_i, y) && (5.6) \\ 0 &\geq - \sum_{i \in B \setminus A} \Omega(y_i, y). && (5.7) \end{aligned}$$

Supermodular higher order terms in Equation (5.2) will be negated, resulting in submodularity. Equation (5.1) is therefore very difficult to optimize globally for multiple detections as maximizing a submodular (minimizing a supermodular) function is in general NP hard.

As our proposed optimization methodology is based on branch-and-bound, the practical constraints of its application to global optimization are key. Branch and bound ceases to be efficient due to curse of dimensionality for approximately 6 or more dimensions. While a bounding box provides a low (four) dimensional parametrization for single object detection, joint optimization of even two boxes leads to a combinatoric explosion of the complexity of the algorithm and is infeasible already for relatively small images. However, as has been exploited by Barinova et al. [10], strong theoretical results about the maximization of submodular functions indicates that a greedy approach gives optimal approximation guarantees for submodular energies [132]. Consequently, our optimization strategy will be to find the best detection without taking into account the non-maximal suppression terms, and then iteratively find subsequent detections, taking into account non-maximal suppression terms only with previously selected detections. The next section addresses the specific implications of this approach for branch and bound strategies, in particular how the structure of the problem can be exploited to improve the computational efficiency of subsequent detections.

5.1.4 Branch and Bound Implementations

Efficient subwindow search (ESS) is a branch and bound framework for object detection that works by storing sets of windows in a priority queue [113, 114]. Sets of windows are specified by intervals indicating the minimum and

maximum coordinates of the four sides of the bounding box, and are ordered by an upper bound on the maximum score of any window within the set. This upper bound, $\hat{f}$, must satisfy two properties in order to guarantee the optimality of the result:

$$\hat{f}(Y) \geq f(y) \quad \forall y \in Y \tag{5.8}$$

$$\hat{f}(\{y\}) = f(y) \tag{5.9}$$

where Y is a set of bounding boxes specified by intervals for the sides of the box, and y is an individual window. The first property states that the upper bound is a true bound, while the second states that the score for a set containing exactly one window should be the true score of the window. Given these properties, when a state containing only one window is dequeued, we are guaranteed that this window has the maximal score of all windows in the image.

As we are pursuing a greedy optimization strategy, we wish to be able to compute upper bounds of the augmented quality function that contains both the unary terms, and the pairwise non-maximal suppression terms. Here, we discuss how to do so for a class of pairwise terms that are monotonic functions of the ratio of the areas of intersection and union of the two windows [52]

$$\Omega(y_i, y_j) = g\left(\frac{\text{Area}(y_i \cap y_j)}{\text{Area}(y_i \cup y_j)}\right) \tag{5.10}$$

where g is any non-negative monotonic function. Consequently, for the kth detection we require an upper bound for

$$\langle f, \phi(x, y_k)\rangle_{\mathcal{H}} - \sum_{i=1}^{k-1} \Omega(y_i, y_k) \tag{5.11}$$

where detections are ordered by their selection by the greedy optimization strategy. We may do so by taking the sum of two bounds, that of the unary terms, the construction of which is discussed for a number of linear and non-linear function classes in [114], and that of the non-maximal suppression term. The bound on the non-maximal suppression terms can be computed as

$$\max_{y \in Y} -g\left(\frac{\text{Area}(y_i \cap y)}{\text{Area}(y_i \cup y)}\right) \leq -g\left(\frac{\min_{y \in Y} \text{Area}(y_i \cap y)}{\max_{y \in Y} \text{Area}(y_i \cup y)}\right) \tag{5.12}$$

$$\leq -g\left(\frac{\min_{y \in Y} \text{Area}(y_i \cap y)}{(\max_{y \in Y} \text{Area}(y)) + \text{Area}(y_i) - (\min_{y \in Y} \text{Area}(y_i \cap y))}\right) \tag{5.13}$$

The computation of the bounds for area of overlap require only constant time given sets of windows specified by intervals.

A key property of greedy optimization of bounds of this form is that the objective for subsequent detections differs only by the subtraction of one additional Ω term. Since Ω is non-negative, this means that any valid bound for an earlier detection remains a valid upper bound for a subsequent detection (Equation (5.8)). This suggests that the computation required to find an earlier detection may be leveraged to more efficiently discover subsequent detections by *keeping the priority queue expanded by an earlier detection*. We also note, however, that Equation (5.9) may be violated if we simply continue the ESS branch-and-bound procedure without modification. This is because a state may be pushed into the priority queue containing only one window, but that does not consider non-maximal suppression terms resulting from detections discovered after that state was pushed into the queue. We can account for this by modifying the ESS algorithm in two ways: (i) we augment a state in the priority queue to store not only the upper bound and intervals specifying the set of bounding boxes, but also to store the number of previous detections considered in the computation of the upper bound, and (ii) we modify the termination criterion to check that the number of detections used for computation of the upper bound is equal to the number of detections found up to that point. If not, the bound is recalculated using all previous detections, and the state is re-inserted into the queue. We make a further assumption on the form of g for the purposes of subsequent analysis:

$$g(x) = \begin{cases} 0 & \text{if } x < \gamma \\ \infty & \text{otherwise} \end{cases} \tag{5.14}$$

where γ is a threshold on the overlap score (e.g. 0.5) above which multiple detections are disallowed. This results in the same non-maximal suppression criterion as used in recent state of the art detection strategies [56, 158, 159].

With these modifications, we can define a family of branch-and-bound strategies for multiple object detections. For each subsequent detection, a strategy may either reset the priority queue to contain a single state containing all possible windows in an image, or it may use a priority queue expanded from a previous detection (Figure 5.2). Each of these strategies will result in the *same set of detections*. Consequently, the goal is to determine a strategy or subset of strategies that reduces the expected computation time[2] of all detections. We fix the number of detections to 10 in this work and note that

[2]We use here the number of dequeuing operations required as a platform independent

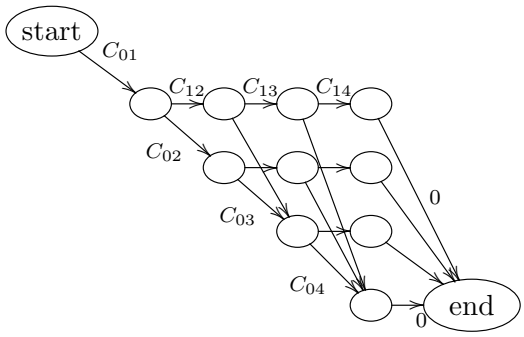


Figure 5.2: Mapping of the selection of an optimal strategy to a shortest path problem. The resulting graph is constructed here for four detections. Horizontal moves correspond to keeping an existing priority queue for a subsequent detection, while diagonal moves correspond to resetting the priority queue to the root node containing the set of all bounding boxes. C_{ij} corresponds to the cost of computing the jth detection using the priority queue carried on from the ith detection. C_{0j} corresponds to the cost when resetting the priority queue prior to computing the jth detection. All edges pointing towards a given node have the same cost. This construction demonstrates that the complexity of computing the optimal strategy *given the branch-and-bound costs* are $\mathcal{O}(n^2)$ for n detections (see text). These costs are not known at test time, but we show empirically that optimal strategies have a very simple form (Section 5.1.6).

a strong pattern is apparent in the empirically observed computation times indicating that results are likely to generalize to other numbers of detections in real data.

5.1.5 Theoretical Results

Branch and bound can be characterized as a best-first search strategy over a DAG whose nodes are isomorphic to a Hasse diagram with direction assigned by set inclusion. We use the notation $\mathcal{Y}$ to indicate the maximal (root) element of the Hasse diagram containing all possible windows, Y to indicate a set of windows ($Y \subset \mathcal{Y}$, $|Y| > 1$), and y to indicate an individual window ($y \in \mathcal{Y}$). In practice, a subset of possible edges are considered corresponding to those such that Y can be represented by intervals. Furthermore, we consider a deterministic rule for splitting Y into two subsets following [113]. We denote the set of nodes visited by the best-first search from the root node with an upper-bound $\hat{f}$ as $S_{\hat{f}} \subset \mathcal{P}(\mathcal{Y})$, where $\mathcal{P}(\mathcal{Y})$ denotes the power set of $\mathcal{Y}$.

Theorem 5.1.1. *For valid upper bounds $\hat{f}_1$ and $\hat{f}_2$,*

$$\hat{f}_1(Y) \geq \hat{f}_2(Y) \quad \forall Y \implies S_{\hat{f}_2} \subseteq S_{\hat{f}_1} \tag{5.15}$$

Proof. Best first search expands all nodes with upper bound greater than the value of the true detection $f(y^*)$. $\hat{f}_2(Y) \geq f(y^*) \implies \hat{f}_1(Y) \geq f(y^*)$, but there may be additional Y for which $\hat{f}_2(Y) < f(y^*) \wedge \hat{f}_1(Y) \geq f(y^*)$. □

Corollary 1. *$S_{\hat{f}_k} \subseteq S_{\hat{f}_i}$, where $k > i$ and $\hat{f}_k$ is a bounding function for the greedy optimization subproblem corresponding to detection k.*

Corollary 1 implies that there is a strict ordering of the number of nodes expanded by different objectives. As any priority queue expanded up to the point of an earlier detection will contain elements computed with a lose upper bound, we conclude that there is a potential computational advantage to resetting the priority queue to the root node for a subsequent detection. However, we also note that if the values of the function change only slightly, there will be a computational overhead to expanding the same nodes over again. Consequently, there may instead be a computational advantage to keeping an existing priority queue.

Stated simply, if we reset the queue to the root node we may have to re-expand nodes that had already been expanded in the previous round. If

measure of the computation time. We note in particular that the bound computation is constant for the family of Ω considered here, making this a natural unit of measurement.

we don't reset the queue, we may have to go through a large number of nodes that have been expanded, but violate the non-maximal suppression condition in Equation (5.14).

Theorem 5.1.2. *The number of nodes to be re-expanded on reset of a queue for detection k is upper bounded by the sum of nodes expanded by other strategies up to that point.*

Proof. Nodes that have been previously expanded in round i can be categorized as belonging to one of two groups: (i) those for which $\hat{f}_i(Y) \geq f(y^*) \wedge \hat{f}_k(Y) \geq f(y^*)$ and (ii) those for which $\hat{f}_i(Y) \geq f(y^*) \wedge \hat{f}_k(Y) < f(y^*)$. All nodes in the first case will be expanded by both strategies, while nodes in the second case will be expanded by the previous detections, but not by the current detection. □

The proof of Theorem 5.1.2 also indicates that in subsequent rounds after a reset, the marginal number of nodes to be expanded is strictly ordered, the older the priority queue, the more nodes will need to be expanded. This implies that once a priority queue has been reset and expanded until a subsequent detection is found, it will be superior to keep using that priority queue rather than one expanded from a previous set of detections.

These theoretical results indicate that for n detections, there are at most 2^{n-1} possible strategies of interest: for each detection after the first, we may either keep the existing priority queue with all expanded states, or we may reset the queue to the root node. If we were to know ahead of time all costs associated with a given choice, we could use a single-source shortest path algorithm to determine the optimal strategy. Figure 5.2 shows a mapping of the problem to a graph for four detections. As the graph is a DAG, the complexity of this procedure is $\mathcal{O}(V)$, where V is the number of vertices. For our graph construction, $V = \frac{n(n+1)}{2} + 2 = \mathcal{O}(n^2)$ resulting in an overall complexity of $\mathcal{O}(n^2)$ for n detections. This allows us *post hoc* to efficiently determine the optimal strategies in our empirical analysis.

This result unfortunately does not allow us to determine the lowest cost approach without precomputing all costs. Possible approaches would be to compute the empirical costs of these strategies for a sample of data, or to use a branch-and-bound strategy in the shortest path algorithm to avoid computing all edge costs. However, we show in Section 5.1.6 that *all* optimal strategies selected by this analysis on the PASCAL VOC data set have a simple form. This form consists of resetting the queue for a fixed number of initial detections, and then keeping the resulting priority queue without

Table 5.1: Statistics of the number of resets to the root node required by optimal strategies. Statistics are reported across classes.

	$\gamma = 0.25$	$\gamma = 0.50$	$\gamma = 0.75$
min	3	2	1
median	4	3	2
max	4	4	3

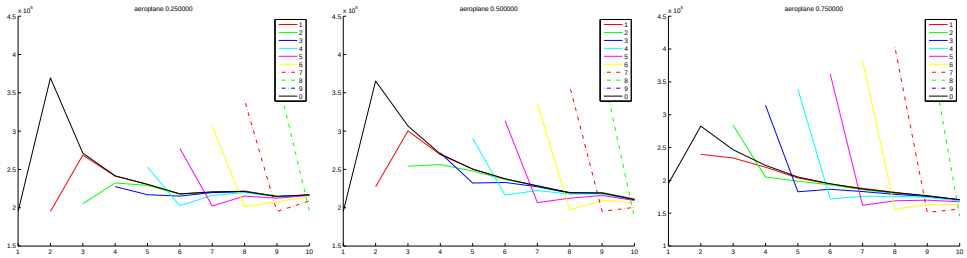
any resets for all subsequent detections. In practice, this indicates that only $n - 1$ of the possible 2^{n-1} strategies are of interest.

5.1.6 Empirical Results

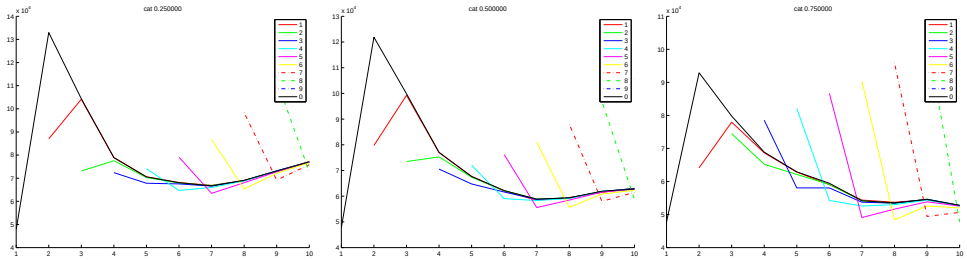
We present results for a modified implementation of the publicly available ESS code described in [114]. We use the feature extraction and trained models downloaded from the author's webpage. All results are reported on the test set of the PASCAL VOC 2007 data set [52], with a different objective trained for each of the 20 classes. Figure 5.3 shows the number of splits required for several selected classes, as well as the average across all classes for varying values of γ (Equation (5.14)). Figure 5.4 shows the number of splits conditioned on the presence or absence of the class of interest averaged across all classes. Table 5.1 shows statistics of the optimal strategy found by a shortest path search. For all classes, the optimal strategy consists of resetting the priority queue to the root node for a number of initial detections followed by re-using the existing priority queue for all subsequent detections. Table 5.2 shows the ratio of the amount of computation required by two simple strategies compared to the optimal strategy.

5.1.7 Discussion

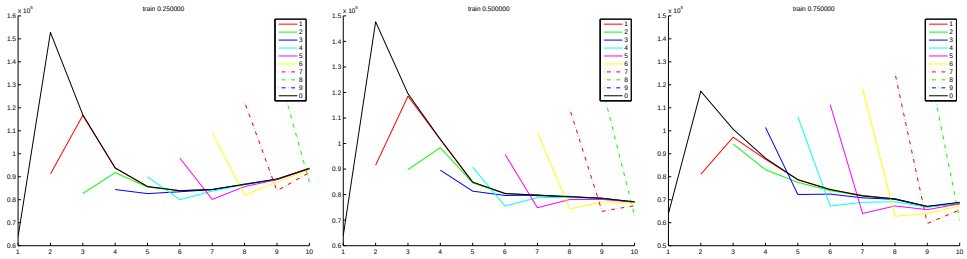
Several broad conclusions can be drawn from the experiments reported in Section 5.1.6. The first, and most important for practical application of branch-and-bound to object detection with non-maximal suppression, is that there is a regime in which resetting the priority queue is more efficient than keeping an existing queue. However, after a few detections, ranging from one to four depending on the class of interest (Table 5.1), it is better to keep an existing priority queue for all subsequent detections. The proof of Theorem 5.1.2 indicates that more recently reset priority queues are *always* preferable to older queues. This has advantages, both in terms of the simplicity of the set of useful strategies, as well as in terms of reducing memory



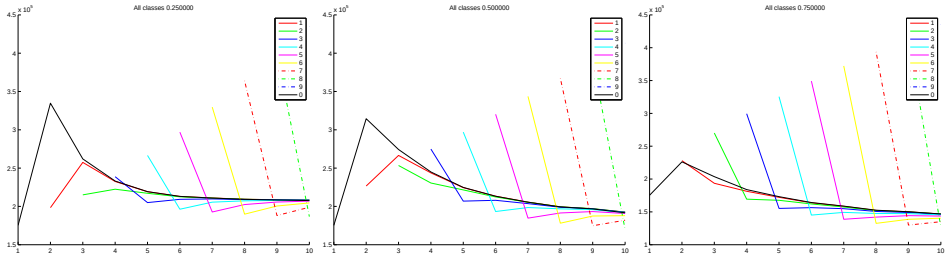
(a) Aeroplane, $\gamma = 0.25$ (b) Aeroplane, $\gamma = 0.50$ (c) Aeroplane, $\gamma = 0.75$



(d) Cat, $\gamma = 0.25$ (e) Cat, $\gamma = 0.50$ (f) Cat, $\gamma = 0.75$



(g) Train, $\gamma = 0.25$ (h) Train, $\gamma = 0.50$ (i) Train, $\gamma = 0.75$



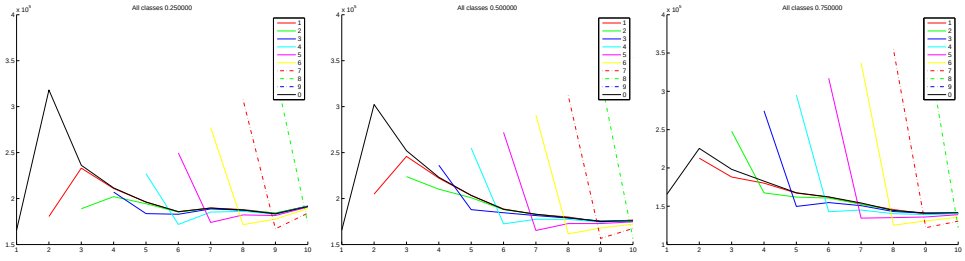
(j) All classes, $\gamma = 0.25$ (k) All classes, $\gamma = 0.50$ (l) All classes, $\gamma = 0.75$

Figure 5.3: Number of splits per subsequent detection when resetting the priority queue at different detections vs. keeping an existing priority queue. x-axis: detection number, y-axis: average number of splits across all images in the VOC2007 test set.

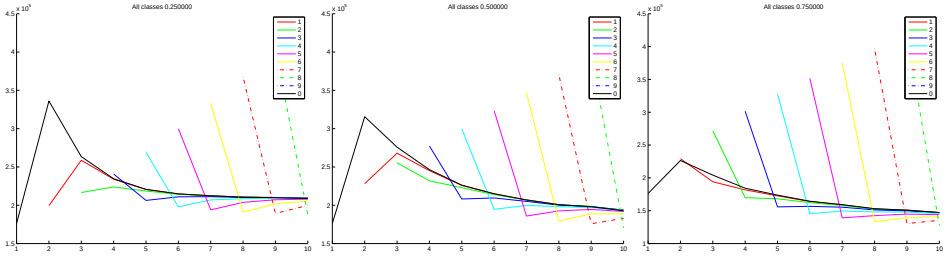
Table 5.2: Ratios of the amount of computation required by two simple strategies to the optimal strategy. The first, naïve strategy consists of resetting the priority queue to the root node at each subsequent detection. The second strategy consists of keeping a single priority queue for all detections without any resets to the root node. Statistics are reported across classes.

$\gamma = 0.25$	**all reset**	**no reset**	$\gamma = 0.50$	**all reset**	**no reset**
min	1.36	1.17	min	1.38	1.16
median	1.48	1.22	median	1.52	1.20
max	1.94	1.28	max	2.19	1.28

$\gamma = 0.75$	**all reset**	**no reset**
min	1.59	1.14
median	2.04	1.16
max	3.15	1.20



(a) $+$, $\gamma = 0.25$ (b) $+$, $\gamma = 0.50$ (c) $+$, $\gamma = 0.75$



(d) $-$, $\gamma = 0.25$ (e) $-$, $\gamma = 0.50$ (f) $-$, $\gamma = 0.75$

Figure 5.4: Number of splits per subsequent detection when resetting the priority queue at different detections vs. keeping an existing priority queue. x-axis: detection number, y-axis: average number of splits across all images and classes in the VOC2007 test set *conditioned on the presence or absence of an object of interest* (denoted $+$ and $-$, respectively).

usage.

Varying behaviors were found when using differing values for γ. In general, the lower the value of γ (more strict non-maximal suppression) the more likely resetting the priority queue is beneficial. As γ increases from 0.25 to 0.75 the median number of resets taken by the optimal strategy for a given class decreases from 4 to 2. This makes intuitive sense as lower values of γ result in strictly higher numbers of nodes in the search graph that will be suppressed in subsequent branch-and-bound optimizations. A large number of expanded nodes around a peak will result in wasted computation as they are subsequently pruned by non-maximal suppression. Conversely, the higher the overlap threshold (less strict non-maximal suppression), the more likely keeping the existing priority queue is helpful.

Conditioning on the class label does not seem to show a large difference in the average number of splits per detection (Figure 5.4). This supports the idea that strategies may be fixed ahead of time.

The marginal cost of the first detection after resetting the priority queue to the root node is not strictly increasing (see e.g. Figure 5.3(d)), but is empirically observed to do so for many classes, and in the average performance across all classes (Figures 5.3(j)-5.3(l)). This result is in line with Theorem 5.1.2 which says that the upper bound on subsequent detections is increasing. This is especially apparent after the first few detections.

Finally, Table 5.2 indicates that of the simple strategies consisting of either always resetting the priority queue or never resetting the priority queue, it is preferable to never reset the priority queue. Our experiments showed that the amount of required computation for 10 detections was higher for each class and overlap threshold when using the resetting strategy than the simple strategy of always keeping the same priority queue.

Commonly applied non-maximal suppression strategies can be interpreted as optimization of a random field model in which non-maximal suppression is captured by pairwise terms encoding the joint distribution of object detection. We have shown in this work how to adapt a branch-and-bound strategy to optimize jointly over multiple detections with non-maximal suppression terms. An optimal approximation result allowed us to frame this as the subsequent application of inter-related branch-and-bound optimizations, enabling us to reuse computations across multiple detections. It is possible to frame the search for a computationally optimal strategy as a shortest path problem on a DAG with $\mathcal{O}(n^2)$ vertices, resulting in efficient *post hoc* computation of the optimal strategies. We have observed that these strategies have a very simple form: although every length $n-1$ bit string encodes a valid strategy resulting in 2^{n-1} possible strategies, all

empirically optimal strategies consisted of first resetting the priority queue for a small number of detections, followed by keeping an existing priority queue. Furthermore, simply keeping a single priority queue for all detections resulted in only a modest increase in the total amount of required computation over the optimal strategy. This indicates that simple strategies can significantly improve computational performance over the naïve application of branch-and-bound in serial.

5.2 Taxonomic Multi-class Prediction

This section is based on [BZG13].

In many fields where large numbers of objects must be categorized, including computer vision, bioinformatics, and document classification, an underlying taxonomic structure is applied. While such taxonomies are useful visualization tools to organize data, and to talk about inter-relationships between (sub)categories, it is less clear whether taxonomies can help to perform structured learning, or whether learned taxonomies outperform those imposed by domain experts.

Several learning algorithms have been developed that make use of user-imposed taxonomies, with the main goal being to improve discriminative performance by using hierarchical structure. For example, [177] proposed a learning framework that incorporated semantic categories, and [23] implemented structured output prediction based on a fixed taxonomic structure. For the most part, these previous works have have found that taxonomic structure results in slight improvements in performance at best, while sometimes decreasing performance. The empirical results in this paper give strong evidence that this may be the result of the user-imposed taxonomy not being aligned to the feature similarities in the data.

In this paper, we make use of a non-parametric dependence measure, the Hilbert-Schmidt Independence Criterion (HSIC), to learn taxonomies. We establish the equivalence between taxonomies and tree structured covariance matrices, and show that the latter constitute a natural way to encode taxonomies in structured prediction problems (indeed, the HSIC is a regularizer for structured output SVM when taxonomies are used). Moreover, we use this tree structured covariance representation to develop a highly efficient algorithm for structured prediction with taxonomies, such that it can be used in large scale problems.

A number of approaches have been proposed for the discovery of taxonomic structure and relationships between classes. Dependency graphs and

co-occurrences were modeled in [25, 112]. [152] proposed to perform a top-down greedy partitioning of the data into trees. Hierarchical clustering has been employed in [53, 79]. Marszałek and Schmid first made use of a semantic hierarchy [127], and later proposed to do a non-disjoint partition into a "relaxed hierarchy" which can then be used for prediction [128]. [175] assume a given taxonomy and then uses a group lasso structured sparsity regularizer with overlapping blocks conforming to the taxonomic structure. In contrast, we do not make the assumption implicit in the group lasso that individual features are exactly aligned with category concepts. [MBZT12] perform hierarchical categorization using a taxonomic feature map and loss, but perform an explicit feature map and do not gain the computational advantages arising from the use of tree structured covariance matrices. [129] consider structured prediction of hierarchically organized image labels using a latent variable method to estimate missing annotations in a weakly supervised setting. None of these methods has identified the relationship between hierarchical prediction and tree-structured covariance matrices. [23] made use of a learning framework that is perhaps the most similar to that employed here, based on structured output prediction. However, they did not learn the taxonomy using a non-parametric dependence measure as we do, but instead used a fixed taxonomic structure.

While these works all make use of some clustering objective distinct from the learning procedure, in contrast, this work employs the Hilbert-Schmidt Independence Criterion, which interestingly is coupled with the learning algorithm in its interpretation as a direct optimization of the function prior in ℓ_2 regularized risk with a taxonomic joint kernel map (cf. Equation (5.28) and Section 5.2.5).

Recent works addressing the machine learning aspects of taxonomic prediction include [170], which embeds a taxonomic structure into Euclidean space, while in contrast our method can efficiently learn from taxonomic structures without this approximation. [18] learn a tree structure in order to improve computational efficiency by only evaluating a logarithmic number of classifiers, while [68] relax this tree structure to a directed acyclic graph. Such greedy methods are advantageous when the number of categories is too large to evaluate exactly, while the current paper addresses the problem of efficient learning when exact evaluation is desired.

In experiments on the PASCAL VOC [52], Oxford Flowers [134], and WIPO-alpha [173] datasets, we show that learned taxonomies substantially improve over hand-designed semantic taxonomies in many cases, and never perform significantly worse. Moreover, we demonstrate that learning using taxonomies is widely applicable to large datasets, thanks to the efficiency of

our algorithm.

Our paper is organized as follows: in Section 2, we review structured output SVMs, following [154]. We proceed in Section 3 to establish the equivalence of taxonomies and tree structured covariance matrices. In Section 4, we show how tree structured covariance matrices may be incorporated into a structured output learning algorithm, and in particular that this representation of taxonomic structure results in substantial computational advantages. In Section 5, we determine how to learn edge lengths of a taxonomy given a fixed topology using the Hilbert-Schmidt Independence Criterion. Finally, Section 6 contains our experimental results.

5.2.1 Taxonomic Prediction

Given a training set of data $\mathcal{S} = \{(x_1, y_1), \ldots, (x_n, y_n)\} \in (\mathcal{X} \times \mathcal{Y})^n$, a structured output SVM with slack rescaling [154, 97] optimizes the following learning objective

$$\min_{w \in \mathbb{R}^d, \xi \in \mathbb{R}} \frac{1}{2}\|w\|^2 + C\xi \tag{5.16}$$

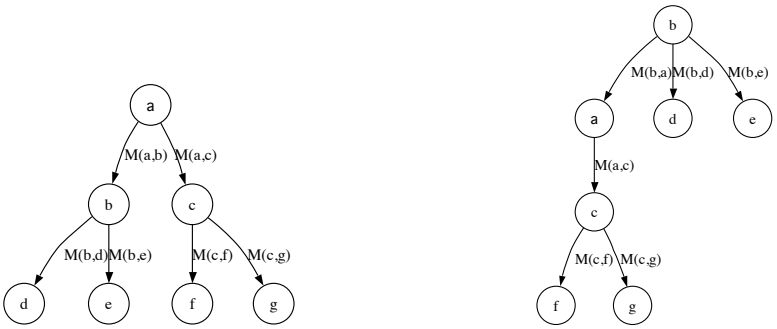
$$\text{s.t.} \sum_i \max_{\tilde{y}_i \in \mathcal{Y}} \left(\langle w, \phi(x_i, y_i) - \phi(x_i, \tilde{y}_i)\rangle - 1\right) \Delta(y_i, \tilde{y}_i) \geq -\xi \tag{5.17}$$

$$\xi \geq 0 \tag{5.18}$$

where ϕ is a joint feature map, and $\Delta(y_i, \tilde{y}_i)$ measures the cost of the erroneous prediction $\tilde{y}_i$ when the correct prediction should be y_i.

Cai and Hofmann proposed a special case of this learning framework in which $\mathcal{Y}$ is taxonomically structured [36]. In that setting, $\phi(x_i, y_i)$ decomposes as $\phi_y(y_i) \otimes \phi_x(x_i)$ and $\phi_y(y_i)$ is a binary vector that encodes the hierarchical relationship between classes. In particular, a taxonomy is defined to be an arbitrary lattice (e.g. tree) whose minimal elements (e.g. leaves) correspond to the categories. $\phi_y(y_i)$ is of length equal to the number of nodes in a taxonomy (equal to the number of categories plus the number of ancestor concepts), and contains non-zero entries at the nodes corresponding to predecessors of the class node. It is straightforward to extend this concept to non-negative entries corresponding to the relative strength of the predecessor relationship. The loss function employed may depend on the length of the shortest path between two nodes [167], or it may be the length of the distance to the nearest common ancestor in the tree [36].

We show in the next two sections that structured prediction with taxonomies is intimately tied to the concept of tree-structured covariance matrices.



(a) A binary rooted tree. Edges are annotated by their length. The tree metric is defined by the sum of the path lengths between two leaf nodes.

(b) Rerooting the tree by setting node "b" to the root. Distances between leaf nodes are preserved regardless of the rooting.

Figure 5.5: An arbitrarily rooted binary tree may be rerooted without changing the pairwise distances between leaf nodes. Furthermore, rerooting has no effect on the value of $HSIC_{\text{cov}}$ (Section 5.2.5 and Theorem 5.2.5).

5.2.2 Tree-structured Covariance Matrices

Here we consider the structure of a covariance matrix necessary to encode taxonomic structure [38, 42].

Definition 5.2.1 (Partition property)**.** A binary matrix V of size $k \times (2k-1)$ has the partition property for trees of size k (i.e. having k leaves) if it satisfies the following conditions:

1. V contains the vector of all ones as a column
2. for every column w in V with more than one non-zero entry, it contains two columns u and v such that $u + v = w$.

We now use this definition to construct a tree structured covariance matrix

Definition 5.2.2 (Tree covariance representation)**.** A matrix B is a tree-structured covariance matrix if and only if $B = VDV^T$ where D is a diagonal matrix with nonnegative entries and V has the partition property.

This definition is chosen to correspond to [42, Theorem 2]. Such an encoding of tree-structured covariance matrices separates the specification

of the topology of the tree, which is encoded in V, from the lengths of the tree branches, which is specified in D. As a concrete example, the tree structured covariance matrix corresponding to Figure 5.5(a) is

$$V = \begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{pmatrix}, \tag{5.19}$$

$$D = \text{diag}[0, M(a,b), M(a,c), M(b,d), M(b,e), M(c,f), M(c,g)]^T,$$

$$B = \begin{pmatrix} M(a,b)+M(b,d) & M(a,b) & 0 & 0 \\ M(a,b) & M(a,b)+M(b,e) & 0 & 0 \\ 0 & 0 & M(a,c)+M(c,f) & M(a,c) \\ 0 & 0 & M(a,c) & M(a,c)+M(c,g) \end{pmatrix}$$

Section 5.2.3 derives a mapping between tree structured covariance matrices and tree metrics, giving a one-to-one relationship and implicitly showing the NP-hardness of optimizing over tree-structured covariance matrices with arbitrary topology.

5.2.3 Properties of Tree-structured Covariances and Tree Metrics

In the sequel, the following lemma will be useful

Lemma 1. *B_{ij} contains the weighted path length from the root to the nearest common ancestor of nodes i and j.*

Proof. Each column of V can be associated with a node in the tree. Each row of V contains a set of binary variables that are equal to 1 iff a corresponding node in the tree is on the path to the leaf associated with that row. As V is binary, $B_{ij} = V_{i:}DV_{j:}^T$ sums over those elements, m, of D for which $V_{im} = V_{jm} = 1$. These elements are exactly the lengths of the branches leading to the common ancestors of nodes i and j. □

Definition 5.2.3 (Four point condition)**.** A metric M satisfies the four point condition if the following holds

$$M(a,b) + M(c,d) \leq \max(M(a,c) + M(b,d), M(a,d) + M(b,c)) \quad \forall a,b,c,d \tag{5.20}$$

Theorem 5.2.4 (Equivalence of the partition property and the 4 point condition)**.** *The following statements are equivalent*

1. *M is a tree metric.*
2. *M satisfies the four point condition.*
3. *$M(i,j) = B_{ii} + B_{jj} - 2B_{ij}$ where $B = VDV^T$ is a tree-structured covariance matrix.*

Proof. $1 \iff 2$ is shown in [35].

$3 \implies 1$: Using Lemma 1, $M(i,j)$ is the length of the path from the root to node i (B_{ii}) plus the length of the path from the root to node j (B_{jj}) minus two times the length of the path to the nearest common ancestor of nodes i and j (B_{ij}). $B_{ii} - B_{ij}$ is therefore the length from node i to the nearest common ancestor of i and j, and $B_{jj} - B_{ij}$ is the length from node j to their nearest common ancestor. $M(i,j)$ is simply the sum of the two subpaths.

$1 \implies 3$ is a consequence of [42, Theorem 2]. □

We note that [35] considered unrooted trees while Definition 5.2.1 and Lemma 1 makes use of the root of a tree. This can be rectified by choosing a root arbitrarily in an unrooted tree (Figure 5.5). Such a choice corresponds to a degree of freedom in the construction of B that is customarily eliminated by data centering, or by working in a canonical basis as in Definition 5.2.1. This is formalized in Theorem 5.2.5.

Theorem 5.2.5 (Centering trees with different roots but identical topology)**.** *Trees with different roots but identical topology project to the same covariance matrix when centered:*

$$H_k B_1 H_k = H_k B_2 H_k, \tag{5.21}$$

where B_1 and B_2 have identical topology and edge weights, but different roots, and $H_k = I - \frac{1}{k} e_k e_k^T$ is a centering matrix, e_k being the length k vector of all ones.

Proof. We first note that the linear operator defined in part 3 of Theorem 5.2.4, $B_{ii} + B_{jj} - 2B_{ij}$, projects to the same metric all tree structured covariance matrices with identical topology and edge weights, but potentially different roots. This is clear as $M(i,j)$ is simply the sum of weights along the unique path from node i to node j. Consequently, this operator applied to $B_1 - B_2$ yields the zero matrix, yielding a system of linear equations describing the null space of the operator. The null space can be summarized in compact matrix notation as follows

$$C e_k e_k^T + e_k e_k^T C \tag{5.22}$$

where C is an arbitrary diagonal matrix. We can consequently write any matrix with a fixed topology and edge weights as the summation of the component that lies in the null space of the operator, and the component that is orthogonal to the null space

$$B_1 = B_\perp + C_1 e_k e_k^T + e_k e_k^T C_1, \tag{5.23}$$

where $B_\perp$ is the component that is orthogonal to the null space, and is identical for all matrices with the same tree topology and edge weights.

We have that $H_k e_k e_k^T = e_k e_k^T H_k = \mathbf{0}$, which yields $H_k(C e_k e_k^T + e_k e_k^T C) H_k = \mathbf{0}$. This in turn implies that

$$\begin{aligned} H_k(B_1 - B_2)H_k =& H_k(B_\perp + C_1 e_k e_k^T + e_k e_k^T C_1 - \\ & B_\perp - C_2 e_k e_k^T - e_k e_k^T C_2) H_k = \mathbf{0} \end{aligned} \tag{5.24}$$

$$H_k B_1 H_k = H_k B_2 H_k. \tag{5.25}$$

□

5.2.4 Structured Prediction with Tree-structured Covariances

Given the concepts developed in Section 5.2.2, we find now that the specification of joint feature maps and loss functions for taxonomic prediction is much simplified. We may assume that a taxonomy is specified that encodes the loss function Δ for a given problem, which need not be the same as a taxonomy for specifying the feature map ϕ. For the minimal path distance, $\Delta(y, \tilde{y}) = M(y, \tilde{y})$ for M defined as in Theorem 5.2.4. For Δ equal to the distance to the nearest common ancestor, we may use $B_{\tilde{y}\tilde{y}} - B_{y\tilde{y}}$. We have used the minimal path distance in the experimental section whenever taxonomic loss has been employed. The standard taxonomic structured loss functions therefore only require as an input a tree-structured covariance matrix B_{loss}, which need not be the same matrix as the one used to define a feature map (0-1 loss is recovered by using the identity matrix).

We now turn to the tree-structured joint kernel map (cf. Section 5.2.1). Given a tree-structured covariance matrix B and its decomposition into $B = VDV^T$, we may compactly define $\phi_y : \mathcal{Y} \mapsto \mathbb{R}^{2k-1}$ as the function that selects the kth column of $D^{\frac{1}{2}} V^T$ when y specifies that the sample belongs to the kth class.[3] Making use of the representer theorem for structured prediction with joint kernel maps [110], we know that the solution to our structured prediction objective lies in the span of our training input data

[3] A rooted tree with k leaves can be encoded with at most $2k - 1$ nodes (Figure 5.5).

$X \subset \mathcal{X}$ crossed with the output space, $\mathcal{Y}$. Assuming a kernel matrix K_x with associated reproducing kernel Hilbert space $\mathcal{F}$ such that the i, jth entry of K_x corresponds to $\langle \phi_x(x_i), \phi_x(x_j) \rangle_{\mathcal{F}}$, we have that the solution may be written

$$\sum_{1 \leq i \leq n} \sum_{y \in \mathcal{Y}} \alpha_{iy} \phi(x_i, y) \tag{5.26}$$

and that the corresponding joint kernel matrix decomposes as $K_x \otimes B$. Although the size of the joint kernel matrix is $n \cdot k \times n \cdot k$, we may make use of several properties of the Kronecker product to avoid high memory storage and costly matrix operations.

Looking specifically at Tikhonov regularized risk:

$$\min_g \lambda \|g\|_{\mathcal{H}}^2 + \ell(g, \mathcal{S}) = \min_\alpha \lambda \alpha^T (K_x \otimes B) \alpha + \ell(\alpha, \mathcal{S}) \tag{5.27}$$

where ℓ is some loss function (we have overloaded the notation in the kernelized case). Interestingly, we may use the identity from Theorem 2.3 of [125]

$$\alpha^T (K_x \otimes B) \alpha = \mathrm{Tr}[K_x \tilde{\alpha}^T B \tilde{\alpha}] \tag{5.28}$$

where $\tilde{\alpha} \in \mathbb{R}^{n \times k}$ is the matrix such that $\mathrm{vec}\, \tilde{\alpha} = \alpha$.

In the case of a structured output SVM, where we have a quadratic regularizer with linear constraints, we can make use of many optimization schemes, that, e.g. require repeated efficient multiplication of a vector with the Hessian:

$$(K_x \otimes B) \alpha = \mathrm{vec}\, B \tilde{\alpha} K_x. \tag{5.29}$$

Using the popular SVMstruct framework [154, 97] in this case generates a large number of non-sparse constraints and is very memory inefficient, requiring the storage of a number of kernel values proportional to the number of tuples in $\mathcal{X} \times \mathcal{Y} \times \mathcal{X} \times \mathcal{Y}$.[4] This indicates that the resulting memory requirements for such a scheme are $\mathcal{O}(n^2 k^2)$, while making use of optimization with Equation (5.29) requires only $\mathcal{O}(n^2 + k^2 + nk)$ memory, and standard large scale kernel learning methods may be applied off-the-shelf to reduce the dominating $\mathcal{O}(n^2)$ component [33]. We have used a cutting plane training to efficiently train our taxonomic predictors, giving the same convergence guarantees as SVMstruct, but with substantially less expensive computation for cutting plane inference.

[4] This follows from an analogous argument to the one used in binary classification that the storage requirements of a SVM are proportional to the Bayes rate, and therefore linear in the number of i.i.d. training samples.

Cutting plane optimization requires finding a setting of $\tilde{y}$ that minimizes the right hand side of Equation (5.17). In the kernelized setting, we substitute for w as in Equation (5.27), and search for parameters $\beta \in \mathbb{R}^{nk \times 1}$ and $\delta \in \mathbb{R}$ that give the kernel coefficients and offset of the linear constraint

$$\delta - \alpha^T (K_x \otimes B)\beta \geq \xi. \tag{5.30}$$

Using Equation (5.29) enables us to solve this cutting plane iteration efficiently, both in terms of computation and memory usage.

In the next section, we discuss how to learn taxonomies from data that are suitable for learning in this structured prediction model.

5.2.5 Optimizing Tree-structured Covariances with the Hilbert-Schmidt Independence Criterion

In this section, we show how a non-parametric dependence test may be employed to learn taxonomies that can then be employed in the construction of a joint feature map for taxonomic prediction.

The Hilbert-Schmidt Independence Criterion (HSIC) is a kernel statistical measure that may be used to measure the dependence between empirical data observations and matrices that encode the hypothesized taxonomic structure of a data set [25]. The HSIC is defined to be the Hilbert-Schmidt norm of the cross covariance operator C_{xy} between mappings from the input space $\mathcal{X}$ and from the label space $\mathcal{Y}$. For characteristic kernels [66],[5] this is zero if and only if X and Y are independent. Given a finite sample of size n from $\Pr_{X,Y}$, the HSIC is

$$HSIC := \mathrm{Tr}[H_n K H_n L] \tag{5.31}$$

where K is the Gram matrix for samples from $\Pr_X$ with (i,j)th entry $k(x_i, x_j)$, and L is the Gram matrix with kernel $l(y_i, y_j)$.

To define our kernel matrix on the output space, we consider a family of functions proposed several times in the literature in the context of HSIC [25, 144]. In particular, we define the kernel in terms of a label matrix $\Pi \in \{0,1\}^{k \times n}$, and a covariance matrix, $B \in \mathbb{R}^{k \times k}$, that encodes the relationship between classes. Given these matrices, $L = \Pi^T B \Pi$. The HSIC with this kernel over $\mathcal{Y}$ is

$$HSIC_{\mathrm{cov}} := \mathrm{Tr}[H_n K H_n \Pi^T B \Pi]. \tag{5.32}$$

As pointed out by [26], $H_k \Pi H_n = \Pi H_n$, which in conjunction with Theorem 5.2.5 indicates that $HSIC_{\mathrm{cov}}$ is identical regardless of how the tree is

[5] e.g. the Gaussian Kernel on $\mathbb{R}^d$.

rooted (cf. Figure 5.5). We note that L is characteristic over $\mathcal{Y}$ whenever $\text{rank}[B] \geq k-1$ and the null space of B is empty or contains e_k.

When K_x is centered, the functional form of Equation (5.28) is identical to Equation (5.32), indicating that the regularizer is $HSIC_{\text{cov}}$ with $\tilde{\alpha}$ in place of Π. While our derivation has focused on tree-structured covariance matrices, this novel theoretical result is applicable to arbitrary covariances over $\mathcal{Y}$, indicating a tight coupling between non-parametric dependence tests and regularization in structured prediction.

With this fundamental relationship in place, we consider in turn optimizing over tree structured covariance matrices with fixed and arbitrary topology. The learned taxonomies may then be employed in structured prediction.

Optimization Over Tree-structured Covariance Matrices

Theorem 5.2.2 gives a convenient decomposition of a tree structured covariance matrix into a binary matrix encoding the topology of the tree and a positive diagonal matrix encoding the branch lengths. One such consequence of the existence of this decomposition is

Theorem 5.2.6. *The set of trees with identical topology is a convex set.*

Proof. [42] Given two tree structured covariance matrices with the same topology, $B = VDV^T$ and $\tilde{B} = V\tilde{D}V^T$, any convex combination can be written

$$\eta B + (1-\eta)\tilde{B} = V\left(\eta D + (1-\eta)\tilde{D}\right)V^T \tag{5.33}$$

for arbitrary $0 \leq \eta \leq 1$. □

Optimization of such covariance matrices with fixed topology is consequently significantly simplified. For D^* maximizing the HSIC subject to a norm constraint, a closed form solution is given by

$$D^* \propto \text{diag}\left[V^T\Pi^T H_n K_x H_n \Pi V\right]. \tag{5.34}$$

We note that this optimization is analogous to that in [25] for tree structured covariance matrices with arbitrary topology. In that work, a closed form solution for arbitrary positive definite matrices was found, which was later projected onto the space of tree-structured matrices using a *numerical taxonomy* algorithm with tight approximation bounds. We have employed the method of [25] for comparison in the experimental results section. Theorems 5.2.4 and 5.2.5 justify the equivalence of our procedures for learning tree-structured covariance matrices with both fixed and arbitrary covariance matrices.

5.2.6 Experimental Results

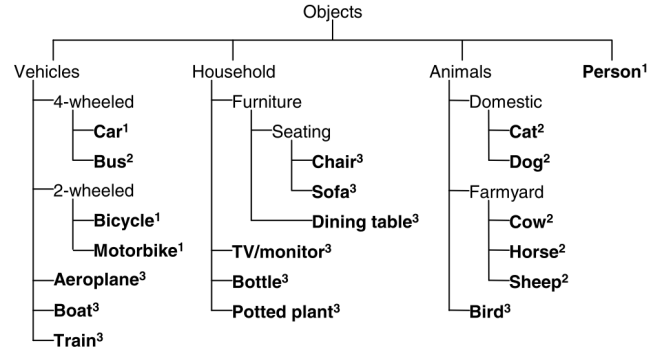
We perform an empirical study on two popular computer vision datasets, PASCAL VOC [52] and Oxford Flowers [134], and on the WIPO text dataset [173].

PASCAL VOC

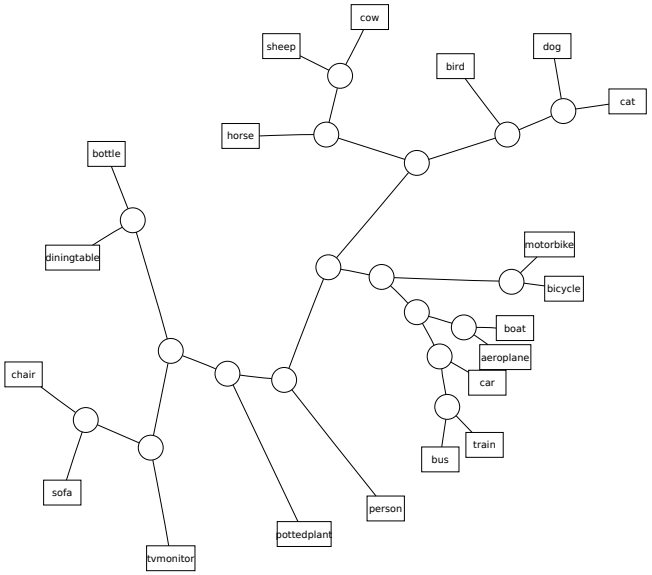
We evaluate the performance of semantic vs. visual taxonomies on the PASCAL VOC 2007 dataset. To construct features for this data, we have employed results from the best performing submission to the 2007 classification challenge, `INRIA_Genetic`, which won all but one category. Our feature vector is constructed by concatenating variance normalized class prediction scores, after which a Gaussian kernel is applied, setting the σ parameter to the median of the pairwise distances in the feature space. As the parameters of the prediction functions were trained on data separate from the test images, this is a proper kernel over the test data set. By construction, we are certain that the relevant visual information is contained within this feature representation, indicating that it is appropriate to use it to optimize the taxonomic structure. Furthermore, the `INRIA_Genetic` method did not make use of taxonomic relationships, meaning that no imputed class relationships will influence the taxonomy discovery algorithm.

The semantic taxonomy was transcribed from the one proposed by the competition organizers [52]. As they do not provide edge lengths for their taxonomy (i.e. relative similarities for each subclass), we have learned these optimally from data using Equation (5.34). We have also learned a taxonomy with unconstrained topology, which is presented in Figure 5.6. Interestingly, the semantic topology and the learned topology are very close despite the learning algorithm's not having access to any information about the topology of the semantic taxonomy.

We have performed classification on the PASCAL VOC data set using the taxonomic prediction method described in Section 5.2.1. We trained on the first 50% of the competition test set, and report results as ROC curves on the second 50%. We emphasize that the results are designed for comparison between semantic and learned visual taxonomies, and are not for comparison within the competition framework. We additionally compare to the multi-class prediction method proposed by [44]. Results are shown in Figure 5.7.

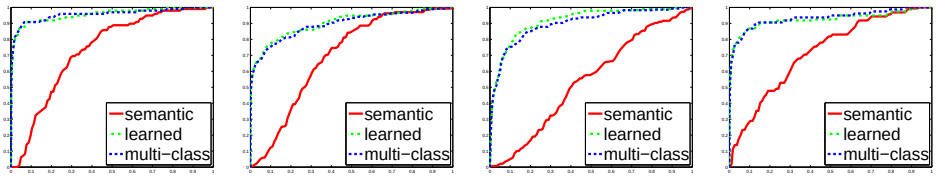


(a) Semantic taxonomy from [52].



(b) Learned visual taxonomy.

Figure 5.6: The semantic and learned taxonomies for the PASCAL VOC dataset. The semantic and visual taxonomies are very close, despite that the construction of the visual taxonomy made no use of the semantic relationships.



(a) aeroplane (b) bicycle (c) bird (d) boat

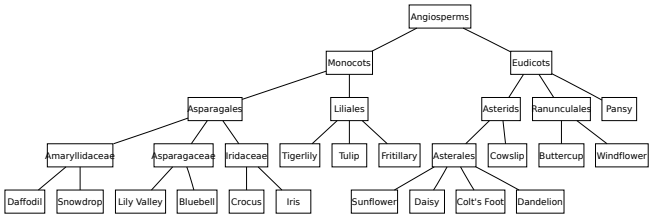
Figure 5.7: ROC curves for the PASCAL VOC dataset. The learned visual taxonomy performs consistently better than the semantic taxonomy. Multi-class classification was performed with a multi-label generalization of [44]. Only the first four classes are shown. The other classes show qualitatively the same relationship between methods.

Oxford Flowers

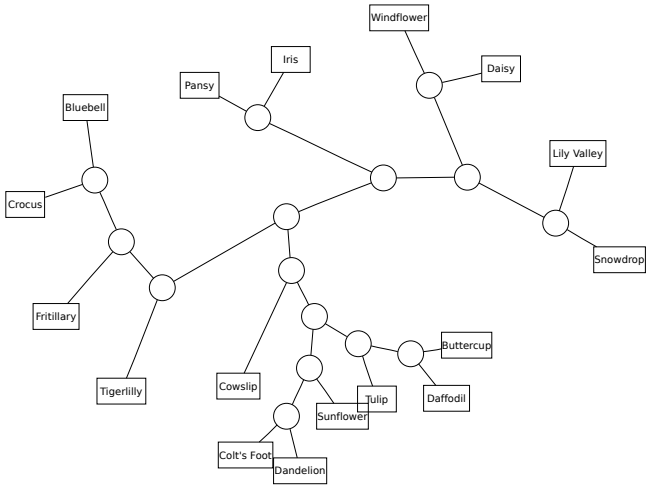
In the second set of experiments, we have compared semantic to visual taxonomies on the Oxford Flowers data set. To construct a rich image representation, we have made use of the features designed by the authors of the dataset. The image representations consist of information encoding color, shape, (local) gradient histograms, and texture descriptors [134]. These features have resulted in high performance on this task in benchmark studies. We have constructed kernel matrices using the mean of Gaussian kernels as described in [72].

The topology of the semantic taxonomy was constructed using the Linnaean biological taxonomy, while edge distances were computed by optimizing D according to Equation (5.34). The topologies of the semantic taxonomy and the learned visual taxonomy are given in Figure 5.8.

We have additionally performed classification using the semantic and learned visual taxonomies. We have applied the taxonomic prediction method described in Section 5.2.1. The results are presented in Table 5.3. In line with previous results on taxonomic prediction, the performance of the taxonomic method with a visual taxonomy performs comparably to 1-vs.-rest classification (here we report the results from [72], which use an identical kernel matrix to our method). However, we note that the semantic taxonomy performs very poorly, while the learned taxonomy maintains good results. We hypothesize that this is due to the strong mismatch between the semantic relationships and the visual ones. In this case, it is inappropriate to make use of a semantic taxonomy, but our approach enables us to gain the benefits of taxonomic prediction without requiring an additional information source to construct the taxonomy.



(a) Semantic taxonomy constructed using biological information.



(b) Learned taxonomy.

Figure 5.8: Semantic and visual taxonomies on the Oxford Flowers dataset. The topologies of the two taxonomies differ significantly, indicating a strong mismatch between the semantic hierarchy and visual similarity.

Table 5.3: Classification scores for the Oxford Flowers data set. The semantic taxonomy (Figure 5.8(a)) gives comparatively poor performance, likely due to the strong mismatch between the biological taxonomy and visual similarity. The learned visual taxonomy (Figure 5.8(b)), however, maintains good performance compared with one-vs.-rest classification.

One vs. rest [72]	**Semantic Taxonomy**	**Learned Taxonomy**
84.9 ± 1.9	56.3 ± 6.3	$\mathbf{87.7 \pm 2.6}$

Table 5.4: Losses on the WIPO data set (lower is better). The columns correspond to varying covariance structures, while the rows correspond to different loss functions. For the covariance structures, I corresponds to a standard multi-class feature map [44], B^* is learned using the method proposed in [25] for learning taxonomies without fixed topology, and D^* is learned from Equation (5.34). Each system was trained with a structured output support vector machine optimizing the loss on which it is evaluated.

	I	B^*	$H_k V D^* V^T H_k$	$V D^* V^T$
0-1	0.281 ± 0.027	$\mathbf{0.278 \pm 0.042}$	0.284 ± 0.037	0.362 ± 0.028
taxonomic	0.950 ± 0.100	$\mathbf{0.833 \pm 0.179}$	1.125 ± 0.071	1.120 ± 0.028

Text Categorization

We present timing and accuracies on the WIPO data set [173], a hierarchically structured document corpus that is commonly used in taxonomic prediction [36]. Kernel design was performed simply using a bag of words feature representation combined with a generalized Gaussian χ^2 kernel with the bandwidth parameter set to the median of the pairwise χ^2 distances. The topology, V, of the tree structure was constructed using the taxonomy provided by the data set organizers. The loss function, Δ, was either set to 0-1 loss, or the taxonomic distance between two concepts. The taxonomic distance between two concepts was measured as the unweighted path length between the two leaves in the taxonomy (i.e. not making use of the learned taxonomy but instead fixing edge lengths to 1).

We have computed results using a number of covariance structures, as well as a number of loss functions. Table 5.4 lists these settings and shows their numerical accuracies. We emphasize that the results correspond to the learning setting proposed by [36] when the covariance matrix is tree-structured. Any differences in performance for this column are due to our using a more recent version of the data set with a comparatively naïve feature representation, while Cai and Hofmann made use of an unspecified kernel function computed using a proprietary software system [36].

We focus on the efficiency of the optimization using our strategy, and the kernelized variant of SVMstruct [154, 97]. We compare the empirical time per cutting plane iteration in Figure 5.9. We note that timing results are presented as a fraction of the first training iteration to account

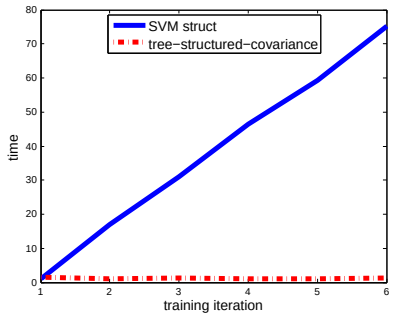


Figure 5.9: Computation time for constraint generation using the proposed method of optimization vs. the popular SVMstruct optimization package [154, 97]. The proposed optimization is several orders of magnitude faster than SVMstruct for this problem, and has constant computation time per iteration, while SVMstruct has computation that grows linearly with the training iteration.

for differences in vector and matrix libraries employed in our implementation vs. SVMstruct. Nevertheless, our implementation was several orders of magnitude faster than SVMstruct at all iterations of training due to the avoidance of naïve looping over cached kernel values as employed by their general purpose framework. In the SVMstruct implementation of taxonomic prediction, the joint kernel function was implemented by multiplying K_{ij} by $B_{y_i y_j}$, which were both kept in memory to optimize computation time. The computation time of our algorithm is constant per iteration, in contrast to SVMstruct, which grows approximately linearly with high slope as the number of support vectors grows. In later training iterations, a single kernelized cutting plane iteration of SVMstruct can take several minutes, while our method takes only several milliseconds. The number of cutting plane iterations required by both methods is identical.

5.2.7 Discussion

In this section, we have compared taxonomies learned from data with semantic taxonomies provided by domain experts, where these taxonomies are used to impose structure in learning problems. While a semantic taxonomy provides a measure of prior information on class relationships, this may be unhelpful to the desired learning outcome when the features available are not in accord with this structure. Indeed, in such cases, we have shown that

the imposition of prior taxonomic information may result in a significant performance penalty.

By contrast, we have observed that learned taxonomies based on feature similarity can do significantly better than hand-designed taxonomies, while never performing significantly worse than alternatives. Moreover, we have shown that the taxonomic structure may be encoded in a tree-structured covariance: as a result, we were able to develop a highly computationally efficient learning algorithm over taxonomies.

5.3 Discussion

In this chapter we have presented contributions to the use of inference techniques in the structured prediction setting. We have focused on two contributions. The first makes use of branch-and-bound inference for the detection of multiple objects in an image. We have shown in a very general presentation that this leads to a supermodular minimization problem, which is NP-hard in general. We therefore employ an iterative algorithm for efficient inference that has known approximation guarantees. The second contribution is focused on taxonomic multi-class prediction. We have shown that the joint kernel matrix decomposes as a Kronecker product between a kernel over the input space, and a kernel over the output space. We substantially increase the speed of inference over a naïve application of a joint kernel representation by application of [125, Theorem 2.3]. In the next chapter we conclude the manuscript.

Chapter 6

Conclusions

In this manuscript, I have summarized my contributions to empirical risk minimization for learning from visual data since my doctorate. Visual data are characterized by the spatial coherence of the solution. This manifests itself in several related ways depending on the specific setting and application. In bounding box based detection of visual objects, we may exploit the spatial structure of the problem to increase the efficiency of inference (Chapter 5) and to appropriately define learning objectives that maximize performance based on representations that account for spatial structure (Chapter 3). In fMRI analysis, structured sparsity regularization can account for correlated signals, which are frequently spatially contiguous, or we may employ a graph representation that incorporates spatial relationships explicitly in the function space (Chapter 4). In medical image segmentation, we may incorporate sophisticated spatial priors that capture the complex patterns present in biological structures (Chapter 5 and [OB14]).

The presentation of our contributions is unified through the principle of regularized empirical risk minimization. We have used the language of empirical risk minimization to categorize our contributions into those of risk, regularization, and inference. These concepts have been presented in the chapters entitled *Empirical Risk*, *Function Classes and Regularization*, and *Representation and Inference*.

We have demonstrated a range of methodological contributions with application to high-level object category recognition and detection, medical image segmentation, and fMRI analysis. Methodological contributions have included the development of novel structured output prediction training objectives, efficient algorithms for their optimization, regularization techniques for semi-supervised canonical correlation analysis and structured sparsity

regularization, branch-and-bound optimization strategies for object detection, and efficient optimization of taxonomic structured prediction.

In future work, we plan to make further contributions to the development of tractable structured output prediction algorithms. We plan to further contribute to the theoretical characterization of structured prediction algorithms, in particular their statistical properties. Contributions to medical image analysis are planned, including a strong focus on fMRI analysis and medical image segmentation. We plan to make use of structured prediction frameworks to improve the accuracy of medical image analysis systems in these application areas. An additional research area that we plan to work on is non-parametric statistical tests for the discovery of statistical structures in data [ZGB13, BGB14].

Bibliography

[1] Shivani Agarwal and Partha Niyogi. Generalization bounds for ranking algorithms via algorithmic stability. *Journal of Machine Learning Research*, 10:441–474, June 2009.

[2] B. Alexe, T. Deselaers, and V. Ferrari. What is an object? In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, June 2010.

[3] Dragomir Anguelov, Ben Taskar, Vassil Chatalbashev, Daphne Koller, Dinkar Gupta, Geremy Heitz, and Andrew Ng. Discriminative learning of Markov random fields for segmentation of 3D scan data. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pages 169–176, 2005.

[4] Andreas Argyriou. A study of convex regularizers for sparse recovery and feature selection. Technical report, 2010.

[5] Andreas Argyriou, Rina Foygel, and Nathan Srebro. Sparse prediction with the k-support norm. In P. Bartlett, F.C.N. Pereira, C.J.C. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1466–1474. 2012.

[6] Francis Bach, Rodolphe Jenatton, and Julien Mairal. *Optimization with Sparsity-Inducing Penalties*. Now Publishers Inc., Hanover, MA, USA, 2011.

[7] Francis R. Bach and Michael I. Jordan. Kernel independent component analysis. *JMLR*, 3:1–48, 2002.

[8] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv. Optimal decoding of linear codes for minimizing symbol error rate. *IEEE Transactions on Information Theory*, 20(2):284–287, 1974.

[9] Gökhan H. Bakır, Thomas Hofmann, Bernhard Schölkopf, Alexander J. Smola, Ben Taskar, and S. V. N. Vishwanathan. *Predicting Structured Data*. The MIT Press, 2007.

[10] O. Barinova, V. Lempitsky, and P. Kohli. On the detection of multiple object instances using Hough transforms. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

[11] A. Bartels and S. Zeki. The chronoarchitecture of the human brain–natural viewing conditions reveal a time-based anatomy of the brain. *NeuroImage*, 22(1):419 – 433, 2004.

[12] A. Bartels and S. Zeki. Functional brain mapping during free viewing of natural scenes. *Human Brain Mapping*, 21(2):75–85, 2004.

[13] A. Bartels and S. Zeki. Brain dynamics during natural viewing conditions–a new guide for mapping connectivity in vivo. *NeuroImage*, 24(2):339–349, 2005.

[14] A. Bartels, S. Zeki, and N. K. Logothetis. Natural vision reveals regional specialization to local motion and to contrast-invariant, global flow in the human brain. *Cereb. Cortex*, 2007.

[15] Peter L. Bartlett, Michael I. Jordan, and Jon D. McAuliffe. Convexity, classification, and risk bounds. *Journal of the American Statistical Association*, 101(473):138–156, 2006.

[16] Herbert Bay, Tinne Tuytelaars, and Luc J. Van Gool. SURF: Speeded up robust features. In *ECCV*, pages 404–417, 2006.

[17] Mikhail Belkin, Partha Niyogi, and Vikas Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of Machine Learning Research*, 7:2399–2434, 2006.

[18] Samy Bengio, Jason Weston, and David Grangier. Label embedding trees for large multi-class tasks. In *NIPS*, pages 163–171. 2010.

[19] Yoshua Bengio. *Learning Deep Architectures for AI*. Now Publishers Inc., Hanover, MA, USA, 2009.

[20] Albert Benveniste, Michel Métivier, and Pierre Priouret. *Adaptive Algorithms and Stochastic Approximations*. Springer-Verlag, New York, NY, 1990.

[21] D. P. Bertsekas. *Nonlinear Programming*. Athena Scientific, 1999.

[22] P. J. Bickel, Y. Ritov, and A. B. Tsybakov. Simultaneous analysis of lasso and Dantzig selector. *Annals of Statistics*, 37(4):1705–1732, 2009.

[23] A. Binder, K.-R. Müller, and M. Kawanabe. On taxonomies for multi-class image categorization. *IJCV*, 2012.

[24] Matthew B. Blaschko. *Kernel Methods in Computer Vision: Object Localization, Clustering, and Taxonomy Discovery*. PhD thesis, Technische Universität Berlin, 2009.

[25] Matthew B. Blaschko and Arthur Gretton. Learning taxonomies by dependence maximization. In *Advances in Neural Information Processing Systems 21*, pages 153–160. 2008.

[26] Matthew B. Blaschko and Arthur Gretton. Taxonomy inference using kernel dependence measures. Technical Report 181, Max Planck Institute for Biological Cybernetics, 2008.

[27] Matthew B. Blaschko, Thomas Hofmann, and Christoph H. Lampert. Efficient subwindow search for object localization. Technical Report 164, Max Planck Institute for Biological Cybernetics, 2007.

[28] Matthew B. Blaschko and Christoph H. Lampert. Correlational spectral clustering. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8, 2008.

[29] Matthew B. Blaschko and Christoph H. Lampert. Learning to localize objects with structured output regression. In David Forsyth, Philip Torr, and Andrew Zisserman, editors, *Computer Vision – ECCV 2008*, volume 5302 of *Lecture Notes in Computer Science*, pages 2–15. Springer Berlin Heidelberg, 2008.

[30] Matthew B. Blaschko, Christoph H. Lampert, and Arthur Gretton. Semi-supervised Laplacian regularization of kernel canonical correlation analysis. In Walter Daelemans, Bart Goethals, and Katharina Morik, editors, *Machine Learning and Knowledge Discovery in Databases*, volume 5211 of *Lecture Notes in Computer Science*, pages 133–145. Springer, 2008.

[31] Antoine Bordes, Léon Bottou, Patrick Gallinari, and Jason Weston. Solving multiclass support vector machines with larank. In *ICML*, 2007.

[32] Antoine Bordes, Seyda Ertekin, Jason Weston, and Léon Bottou. Fast kernel classifiers with online and active learning. *Journal of Machine Learning Research*, 6:1579–1619, December 2005.

[33] Léon Bottou, Olivier Chapelle, Dennis DeCoste, and Jason Weston, editors. *Large Scale Kernel Machines*. MIT Press, Cambridge, MA., 2007.

[34] Olivier Bousquet and André Elisseeff. Stability and generalization. *Journal of Machine Learning Research*, 2:499–526, 2002.

[35] P. Buneman. The recovery of trees from measures of dissimilarity. In D. G. Kendall and P. Tautu, editors, *Mathematics in the Archeological and Historical Sciences*, pages 387–395. Edinburgh University Press, 1971.

[36] L. Cai and T. Hofmann. Hierarchical document categorization with support vector machines. In *CIKM*, 2004.

[37] M.K. Carroll, G.A. Cecchi, I. Rish, R. Garg, and A.R. Rao. Prediction and interpretation of distributed neural activity with sparse models. *NeuroImage*, 44(1):112 – 122, 2009.

[38] L. L. Cavalli-Sforza and A. W. F. Edwards. Phylogenetic analysis: Models and estimation procedures. *American Journal of Human Genetics*, 19:223–257, 1967.

[39] Olivier Chapelle. Training a support vector machine in the primal. *Neural Computation*, 19(5):1155–1178, May 2007.

[40] Olivier Chapelle, Bernhard Schölkopf, and Alexander Zien. *Semi-Supervised Learning*. The MIT Press, 2010.

[41] Dan Claudiu Cireşan, Ueli Meier, Luca Maria Gambardella, and Jürgen Schmidhuber. Deep, big, simple neural nets for handwritten digit recognition. *Neural Computation*, 22(12):3207–3220, December 2010.

[42] H. Corrada Bravo, S. Wright, K. Eng, S. Keleş, and G. Wahba. Estimating tree-structured covariance matrices via mixed-integer programming. In *AISTATS*, 2009.

[43] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[44] K. Crammer and Y. Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *JMLR*, 2:265–292, 2002.

[45] Culbertson CS, Bramen J, Cohen MS, et al. Effect of bupropion treatment on brain activation induced by cigarette-related cues in smokers. *Archives of General Psychiatry*, 68(5):505–515, 2011.

[46] Paul Dagum and Michael Luby. Approximating probabilistic inference in Bayesian belief networks is NP-hard. *Artif. Intell.*, 60(1):141–153, 1993.

[47] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2005.

[48] Jeffrey Dean, Greg Corrado, Rajat Monga, Kai Chen, Matthieu Devin, Quoc Le, Mark Mao, Marc'Aurelio Ranzato, Andrew Senior, Paul Tucker, Ke Yang, and Andrew Ng. Large scale distributed deep networks. In P. Bartlett, F.C.N. Pereira, C.J.C. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1232–1240. 2012.

[49] C. Desai, D. Ramanan, and C. Fowlkes. Discriminative models for multi-class layout. In *Proceedings of the International Conference on Computer Vision*, 2009.

[50] Francesco Dinuzzo and Bernhard Schölkopf. The representer theorem for Hilbert spaces: a necessary and sufficient condition. In P. Bartlett, F.C.N. Pereira, C.J.C. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 189–196. 2012.

[51] Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification*. Wiley, 2nd edition, 2000.

[52] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The Pascal visual object classes (VOC) challenge. *International Journal of Computer Vision*, 88(2):303–338, June 2010.

[53] X. Fan. Efficient multiclass object detection by a hierarchy of classifiers. In *CVPR*, 2005.

[54] C. Farabet, C. Couprie, L. Najman, and Y. LeCun. Learning hierarchical features for scene labeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1915–1929, 2013.

[55] P. Felzenszwalb and D. Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59(2):167–181, 2004.

[56] P. F. Felzenszwalb, D. McAllester, and D. Ramanan. A discriminatively trained, multiscale, deformable part model. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[57] R. A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7(7):179–188, 1936.

[58] R. A. Fisher. The statistical utilization of multiple measurements. *Annals of Eugenics*, 8:376–386, 1938.

[59] T.R. Franklin, Z. Wang, Y. Li, et al. Dopamine transporter genotype modulation of neural responses to smoking cues: confirmation in a new cohort. *Addiction Biology*, 16(2):308–322, 2011.

[60] David Freedman. *Statistical Models: Theory and Practice*. Cambridge University Press, 2005.

[61] Brendan J. Frey. *Graphical Models for Machine Learning and Digital Communication*. MIT Press, Cambridge, MA, USA, 1998.

[62] J. Friedman, T. Hastie, and R. Tibshirani. A note on the group lasso and a sparse group lasso. Technical Report arXiv:1001.0736, Jan 2010.

[63] K. J. Friston, A. P. Holmes, K. J. Worsley, J. P. Poline, C. D. Frith, and R. S. J. Frackowiak. Statistical parametric maps in functional imaging: A general linear approach. *Human Brain Mapping*, 2(4):189–210, 1994.

[64] K.J. Friston, J. Ashburner, S.J. Kiebel, T.E. Nichols, and W.D. Penny, editors. *Statistical Parametric Mapping: The Analysis of Functional Brain Images*. Academic Press, 2007.

[65] S. Fujishige. *Submodular Functions and Optimization*. Elsevier, 2005.

[66] K. Fukumizu, A. Gretton, X. Sun, and B. Schölkopf. Kernel measures of conditional dependence. In *NIPS*, pages 489–496, 2008.

[67] J. Gall and V. Lempitsky. Class-specific hough forests for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

[68] Tianshi Gao and Daphne Koller. Discriminative learning of relaxed hierarchy for large-scale visual recognition. In *ICCV*, pages 2072–2079, 2011.

[69] Michael R. Garey and David S. Johnson. *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., 1979.

[70] Thomas Gärtner, Peter Flach, and Stefan Wrobel. On graph kernels: Hardness results and efficient alternatives. In Bernhard Schölkopf and Manfred K. Warmuth, editors, *Learning Theory and Kernel Machines*, volume 2777 of *Lecture Notes in Computer Science*, pages 129–143. Springer Berlin Heidelberg, 2003.

[71] C. F. Gauß. *Theoria motus corporum coelestium*. Königliche Gesellschaft der Wissenschaften, 1906.

[72] P. Gehler and S. Nowozin. On feature combination methods for multiclass object classification. In *ICCV*, 2009.

[73] Katerina Gkirtzou. *Sparsity regularization and graph-based representation in medical imaging*. PhD thesis, École Centrale Paris, 2013.

[74] Boris Goldengorin, Gerard Sierksma, Gert A. Tijssen, and Michael Tso. The data-correcting algorithm for the minimization of supermodular functions. *Management Science*, 45(11):1539–1551, 1999.

[75] R.Z. Goldstein, N. Alia-Klein, D. Tomasi, J.H. Carrillo, T. Maloney, P.A. Woicik, R. Wang, F. Telang, and N.D. Volkow. Anterior cingulate cortex hypoactivations to an emotionally salient task in cocaine addiction. *PNAS*, 106(23):9453, 2009.

[76] R.Z. Goldstein, P.A. Woicik, T. Maloney, et al. Oral methylphenidate normalizes cingulate activity in cocaine addiction during a salient cognitive task. *PNAS*, 107(38):16667–16672, 2010.

[77] K. Grauman and T. Darrell. Approximate Correspondences in High Dimensions. In *Advances in Neural Information Processing Systems 19 (NIPS)*, 2007.

[78] Kristen Grauman and Trevor Darrell. The pyramid match kernel: Efficient learning with sets of features. *Journal of Machine Learning Research*, 8:725–760, May 2007.

[79] G. Griffin and P. Perona. Learning and using taxonomies for fast visual categorization. In *CVPR*, 2008.

[80] Carlos Guestrin, Andreas Krause, and Ajit Singh. Near-optimal sensor placements in Gaussian processes. In *International Conference on Machine Learning (ICML)*, August 2005.

[81] R. W. Hamming. Error detecting and error correcting codes. *Bell System Technical Journal*, 29(2):147–160, 1950.

[82] David R. Hardoon, Sándor Szedmák, and John R. Shawe-Taylor. Canonical correlation analysis: An overview with application to learning methods. *Neural Computation*, 16(12):2639–2664, 2004.

[83] D.R. Hardoon, J. Mourão-Miranda, M. Brammer, and J. Shawe-Taylor. Unsupervised analysis of fMRI data using kernel canonical correlation. *NeuroImage*, 37(4):1250 – 1259, 2007.

[84] C. Harris and M. Stephens. A combined corner and edge detector. In *Proc. of The Fourth Alvey Vision Conference*, pages 147–151, 1988.

[85] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer, 2001.

[86] David Haussler. Convolution kernels on discrete structures. Technical Report UCSC-CRL-99-10, Department of Computer Science, University of California at Santa Cruz, 1999.

[87] Matthias Hein, Jean-Yves Audibert, and Ulrike von Luxburg. Graph Laplacians and their convergence on random neighborhood graphs. *Journal of Machine Learning Research*, 8:1325–1370, 2007.

[88] B. Hemery, H. Laurent, and C. Rosenberger. Comparative study of metrics for evaluation of object localisation by bounding boxes. In *Image and Graphics, 2007. ICIG 2007. Fourth International Conference on*, pages 459 –464, August 2007.

[89] R. Herbrich, T. Graepel, and K. Obermayer. Large margin rank boundaries for ordinal regression. In A.J. Smola, P.L. Bartlett, B. Schölkopf, and D. Schuurmans, editors, *Advances in Large Margin Classifiers*, pages 115–132. MIT Press, 2000.

[90] Geoffrey E. Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7):1527–1554, July 2006.

[91] Geoffrey Hollinger and Sanjiv Singh. Proofs and experiments in scalable, near-optimal search by multiple robots. In *Robotics: Science and Systems*, June 2008.

[92] J. Honorio, D. Tomasi, R. Goldstein, H.C. Leung, and D. Samaras. Can a single brain region predict a disorder? *IEEE Transactions on Medical Imaging*, 2012.

[93] Harold Hotelling. Relations between two sets of variates. *Biometrika*, 28:321–377, 1936.

[94] Cho-Jui Hsieh, Kai-Wei Chang, Chih-Jen Lin, S. Sathiya Keerthi, and S. Sundararajan. A dual coordinate descent method for large-scale linear SVM. In *ICML*, 2008.

[95] Junzhou Huang, Tong Zhang, and Dimitris Metaxas. Learning with structured sparsity. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 417–424, 2009.

[96] Rodolphe Jenatton, Alexandre Gramfort, Vincent Michel, Guillaume Obozinski, Evelyn Eger, Francis Bach, and Bertrand Thirion. Multiscale mining of fMRI data with hierarchical structured sparsity. *SIAM Journal on Imaging Sciences*, 5(3):835–856, 2012.

[97] Thorsten Joachims, Thomas Finley, and Chun-Nam John Yu. Cutting-plane training of structural SVMs. *Machine Learning*, 77(1):27–59, October 2009.

[98] Torsten Joachims. Training linear SVMs in linear time. In *ACM KDD*, 2006.

[99] Michael I. Jordan, editor. *Learning in Graphical Models*. MIT Press, Cambridge, MA, USA, 1999.

[100] D. Kempe, J. Kleinberg, and E. Tardos. Maximizing the spread of influence through a social network. In *Proc. KDD*, 2003.

[101] G. Kimeldorf and G. Wahba. A correspondence between Bayesian estimation of stochastic processes and smoothing by splines. *Annals of Mathematical Statistics*, 41:495–502, 1970.

[102] G. S. Kimeldorf and G. Wahba. Some results on Tchebycheffian spline functions. *Journal of Mathematical Analysis and Applications*, 33(1):82–95, 1971.

[103] Ross Kindermann and J. Laurie Snell. *Markov random fields and their applications*, volume 1 of *Contemporary Mathematics*. American Mathematical Society, Providence, R.I., 1980.

[104] Donald E. Knuth. Two notes on notation. *American Mathematical Monthly*, 99(5):403–422, May 1992.

[105] Daphne Koller and Nir Friedman. *Probabilistic Graphical Models: Principles and Techniques*. The MIT Press, 2009.

[106] Vladimir Kolmogorov and Ramin Zabih. What energy functions can be minimized via graph cuts? *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(2):147–159, 2004.

[107] Andreas Krause. SFO: A toolbox for submodular function optimization. *Journal of Machine Learning Research*, 11:1141–1144, 2010.

[108] Andreas Krause, Carlos Guestrin, Anupam Gupta, and Jon Kleinberg. Near-optimal sensor placements: Maximizing information while minimizing communication cost. In *International Symposium on Information Processing in Sensor Networks (IPSN)*, April 2006.

[109] Alex Krizhevsky, Ilya Sutskever, and Geoff Hinton. Imagenet classification with deep convolutional neural networks. In P. Bartlett, F.C.N. Pereira, C.J.C. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1106–1114. 2012.

[110] John Lafferty, Xiaojin Zhu, and Yan Liu. Kernel conditional random fields: representation and clique selection. In *Proceedings of the International Conference on Machine Learning*, 2004.

[111] P.L. Lai and C. Fyfe. Kernel and nonlinear canonical correlation analysis. *Int. J. Neural Syst.*, 10(5):365–377, 2000.

[112] Christoph H. Lampert and Matthew B. Blaschko. A multiple kernel learning approach to joint multi-class object detection. In Gerhard Rigoll, editor, *Pattern Recognition*, volume 5096 of *Lecture Notes in Computer Science*, pages 31–40. Springer, 2008.

[113] Christoph H. Lampert, Matthew B. Blaschko, and Thomas Hofmann. Beyond sliding windows: Object localization by efficient subwindow search. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8, 2008.

[114] Christoph H. Lampert, M.B. Blaschko, and T. Hofmann. Efficient subwindow search: A branch and bound framework for object localization. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(12):2129–2142, Dec 2009.

[115] I. Laptev. Improvements of object detection using boosted histograms. In *Proceedings of the European Conference on Computer Vision*, 2006.

[116] S. L. Lauritzen. *Graphical Models*. Clarendon Press, 1996.

[117] A. M. Legendre. *Nouvelles méthodes pour la détermination des orbites des comètes*. F. Didot, 1805.

[118] A. Lehmann, B. Leibe, and L. van Gool. Feature-centric efficient subwindow search. In *Proceedings of the International Conference on Computer Vision*, 2009.

[119] Alain Lehmann, Bastian Leibe, and Luc Van Gool. Fast PRISM: Branch and bound Hough transform for object class detection. *International Journal of Computer Vision*, 2010.

[120] B. Leibe, A. Leonardis, and B. Schiele. Combined object categorization and segmentation with implicit shape model. In *ECCV Workshop on Statistical Learning in Comp. Vision*, 2004.

[121] S. E. Leurgans, R. A. Moyeed, and B. W. Silverman. Canonical correlation analysis when the data are curves. *Journal of the Royal Statistical Society, Series B (Methodological)*, 55(3):725–740, 1993.

[122] Chih-Jen Lin, Ruby C. Weng, and S. Sathiya Keerthi. Trust region Newton method for logistic regression. *JMLR*, 2008.

[123] D. A. Lisin, M. A. Mattar, M. B. Blaschko, E. G. Learned-Miller, and M. C. Benfield. Combining local and global image features for object class recognition. In *Computer Vision and Pattern Recognition - Workshops*, 2005.

[124] Laszló Lovász. Submodular functions and convexity. *Mathematical programming: The state of the art*, 1983.

[125] J. R. Magnus and H. Neudecker. *Matrix Differential Calculus with Applications in Statistics and Econometrics*. Wiley, 1999.

[126] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[127] Marcin Marszałek and Cordelia Schmid. Semantic hierarchies for visual object recognition. In *CVPR*, 2007.

[128] Marcin Marszałek and Cordelia Schmid. Constructing category hierarchies for visual recognition. In *ECCV*, 2008.

[129] JulianJ. McAuley, Arnau Ramisa, and TibrioS. Caetano. Optimization of robust loss functions for weakly-labeled image taxonomies. *IJCV*, pages 1–19, 2012.

[130] Colin McDiarmid. On the method of bounded differences. In J. Siemons, editor, *Surveys in Combinatorics*, pages 148–188. Cambridge University Press, 1989.

[131] Kazuo Murota. *Discrete Convex Analysis*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2003.

[132] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher. An analysis of approximations for maximizing submodular set functions. *Mathematical Programming*, 14:265–294, 1978.

[133] B. Ng, V. Siless, G. Varoquaux, J.-B. Poline, B. Thirion, and R. Abugharbieh. Connectivity-informed sparse classifiers for fMRI brain decoding. In *Pattern Recognition in Neuroimaging*, 2012.

[134] M-E. Nilsback and A. Zisserman. Delving deeper into the whorl of flower segmentation. *Image and Vision Computing*, 2009.

[135] Joaquin Quiñonero Candela, Masashi Sugiyama, Anton Schwaighofer, and Neil D. Lawrence, editors. *Dataset Shift in Machine Learning*. The MIT Press, 2009.

[136] Rajat Raina, Anand Madhavan, and Andrew Y. Ng. Large-scale deep unsupervised learning using graphics processors. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 873–880, 2009.

[137] Bernhard Schölkopf, Ralf Herbrich, and Alex J. Smola. A generalized representer theorem. In David Helmbold and Bob Williamson, editors, *Computational Learning Theory*, volume 2111 of *Lecture Notes in Computer Science*, pages 416–426. Springer, 2001.

[138] Bernhard Schölkopf, John C. Platt, John Shawe-Taylor, Alex J. Smola, and Robert C. Williamson. Estimating the support of a high-dimensional distribution. *Neural Computation*, 13(7):1443–1471, 2001.

[139] Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299–1319, 1998.

[140] Bernhard Schölkopf and Alexander J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2001.

[141] A. Schrijver. *Combinatorial Optimization: Polyhedra and Efficiency*. Algorithms and combinatorics. Springer, 2003.

[142] Shai Shalev-Shwartz, Yoram Singer, and Nathan Srebro. Pegasos: Primal Estimated sub-GrAdient SOlver for SVM. In *ICML*, 2007.

[143] Nino Shervashidze, Pascal Schweitzer, Erik Jan van Leeuwen, Kurt Mehlhorn, and Karsten M. Borgwardt. Weisfeiler-Lehman graph kernels. *Journal of Machine Learning Research*, 12:2539–2561, November 2011.

[144] L. Song, A. Smola, A. Gretton, and K. M. Borgwardt. A dependence maximization view of clustering. In *ICML*, 2007.

[145] S. Song, Z. Zhan, Z. Long, J. Zhang, and L. Yao. Comparative study of svm methods combined with voxel selection for obj ect category classification on fmri data. *PLoS One*, 6(2):e17191, 2011.

[146] O. Sporns. *Networks of the Brain*. MIT Press, 2010.

[147] Joes Staal, Michael D Abràmoff, Meindert Niemeijer, Max A Viergever, and Bram van Ginneken. Ridge based vessel segmentation in color images of the retina. *IEEE T-MI*, 23(4):501–509, 2004.

[148] Martin Szummer, Pushmeet Kohli, and Derek Hoiem. Learning CRFs using graph cuts. In David Forsyth, Philip Torr, and Andrew Zisserman, editors, *Computer Vision – ECCV 2008*, volume 5303 of *Lecture Notes in Computer Science*, pages 582–595. Springer, 2008.

[149] Ben Taskar, Carlos Guestrin, and Daphne Koller. Max-margin Markov networks. In Sebastian Thrun, Lawrence Saul, and Bernhard Schölkopf, editors, *Advances in Neural Information Processing Systems 16*. MIT Press, Cambridge, MA, 2004.

[150] David Martinus Johannes Tax. *One-class classification: Concept-learning in the absence of counter-examples*. PhD thesis, Delft University of Technology, 2001.

[151] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society Series B*, 58:267–288, 1996.

[152] R. Tibshirani and T. Hastie. Margin trees for high-dimensional classification. *JMLR*, 8:637–652, 2007.

[153] A. N. Tikhonov. Solution of incorrectly formulated problems and the regularization method. *Soviet Math. Dokl.*, 4:1035–1038, 1963.

[154] Ioannis Tsochantaridis, Thomas Hofmann, Thorsten Joachims, and Yasemin Altun. Support vector machine learning for interdependent and structured output spaces. In *Proceedings of the Twenty-first International Conference on Machine learning*, 2004.

[155] Ioannis Tsochantaridis, Thorsten Joachims, Thomas Hofmann, and Yasemin Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6:1453–1484, 2005.

[156] V. N. Vapnik and A. Y. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications*, 16(2):264–280, 1971.

[157] Vladimir N. Vapnik. *The Nature of Statistical Learning Theory*. Springer, 1995.

[158] A. Vedaldi, V. Gulshan, M. Varma, and A. Zisserman. Multiple kernels for object detection. In *Proceedings of the International Conference on Computer Vision*, 2009.

[159] A. Vedaldi and A. Zisserman. Structured output regression for detection with partial occulsion. In *Advances in Neural Information Processing Systems*, 2009.

[160] Régis Vert and Jean-Philippe Vert. Consistency of one-class SVM and related algorithms. In *NIPS*, 2005.

[161] S. Vichy, N. Vishwanathan, Nicol N. Schraudolph, Risi Imre Kondor, and Karsten M. Borgwardt. Graph kernels. *Journal of Machine Learning Research*, 11:1201–1242, 2010.

[162] Paul Viola and Michael Jones. Robust real-time object detection. *International Journal of Computer Vision*, 57(2):137–154, 2002.

[163] A. J. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, 13(2):260–269, 1967.

[164] N. D. Volkow, J. S. Fowler, and G. J. Wang. The addicted human brain: Insights from imaging studies. *Journal of Clinical Investigation*, 111:1444–1451, 2003.

[165] N. D. Volkow, J. S. Fowler, and G. J. Wang. The addicted human brain viewed in the light of imaging studies: Brain circuits and treatment strategies. *Neuropharmacology*, 47 Suppl 1:3–13, 2004.

[166] Martin J. Wainwright and Michael I. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1-2):1–305, January 2008.

[167] K. Wang, S. Zhou, and S. C. Liew. Building hierarchical classifiers using class proximity. In *VLDB*, 1999.

[168] Joe H. Ward. Hierarchical Grouping to Optimize an Objective Function. *Journal of the American Statistical Association*, 58(301):236–244, March 1963.

[169] Chong-Yaw Wee, Pew-Thian Yap, Wenbin Li, Kevin Denny, Jeffrey N. Browndyke, Guy G. Potter, Kathleen A. Welsh-Bohmer, Lihong Wang, and Dinggang Shen. Enriched white matter connectivity networks for accurate identification of {MCI} patients. *NeuroImage*, 54(3):1812 – 1822, 2011.

[170] K.Q. Weinberger and O. Chapelle. Large margin taxonomy embedding for document categorization. In *NIPS*, pages 1737–1744. 2009.

[171] Boris Weisfeiler and A.A. Lehman. A reduction of a graph to a canonical form and an algebra arising during this reduction. *Nauchno-Technicheskaya Informatsia*, 2(9):12–16, 1968.

[172] David H. Wolpert. The lack of a priori distinctions between learning algorithms. *Neural Computation*, 8(7):1341–1390, October 1996.

[173] World Intellectual Property Organization. WIPO-alpha data set, `http://www.wipo.int/`, 2009.

[174] Ziming Zhang, Jonathan Warrell, and Philip Torr. Proposal generation for object detection using cascaded ranking SVMs. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

[175] Bin Zhao, Fei Fei F. Li, and Eric P. Xing. Large-scale category structure aware image categorization. In *NIPS*, pages 1251–1259. 2011.

[176] H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society Series B*, 67(2):301–320, 2005.

[177] A. Zweig and D. Weinshall. Exploiting object hierarchy: Combining models from different category levels. In *ICCV*, 2007.