Journal of Machine Learning Research 5 (2004) 27-72 Submitted 10/02; Revised 8/03; Published 1/04

Learning the Kernel Matrix with Semidefinite Programming

Gert R.G. Lanckriet GERT@EECS.BERKELEY.EDU
Department of Electrical Engineering and Computer Science
University of California
Berkeley, CA 94720, USA

Nello Cristianini NELLO@SUPPORT-VECTOR.NET
Department of Statistics
University of California
Davis, CA 95616, USA

Peter Bartlett BARTLETT@STAT.BERKELEY.EDU
Department of Electrical Engineering and Computer Science and Department of Statistics
Berkeley, CA 94720, USA

Laurent El Ghaoui ELGHAOUI@EECS.BERKELEY.EDU
Department of Electrical Engineering and Computer Science
University of California
Berkeley, CA 94720, USA

Michael I. Jordan JORDAN@STAT.BERKELEY.EDU
Department of Electrical Engineering and Computer Science and Department of Statistics
University of California
Berkeley, CA 94720, USA

Editor: Bernhard Schölkopf

Abstract

Kernel-based learning algorithms work by embedding the data into a Euclidean space, and then searching for linear relations among the embedded data points. The embedding is performed implicitly, by specifying the inner products between each pair of points in the embedding space. This information is contained in the so-called kernel matrix, a symmetric and positive semidefinite matrix that encodes the relative positions of all points. Specifying this matrix amounts to specifying the geometry of the embedding space and inducing a notion of similarity in the input space—classical model selection problems in machine learning. In this paper we show how the kernel matrix can be learned from data via semidefinite programming (SDP) techniques. When applied to a kernel matrix associated with both training and test data this gives a powerful transductive algorithm—using the labeled part of the data one can learn an embedding also for the unlabeled part. The similarity between test points is inferred from training points and their labels. Importantly, these learning problems are convex, so we obtain a method for learning both the model class and the function without local minima. Furthermore, this approach leads directly to a convex method for learning the 2-norm soft margin parameter in support vector machines, solving an important open problem.

Keywords: kernel methods, learning kernels, transduction, model selection, support vector machines, convex optimization, semidefinite programming

1. Introduction

Recent advances in kernel-based learning algorithms have brought the field of machine learning closer to the desirable goal of autonomy—the goal of providing learning systems that require as little intervention as possible on the part of a human user. In particular, kernel-based algorithms are generally formulated in terms of convex optimization problems, which have a single global optimum and thus do not require heuristic choices of learning rates, starting configurations or other free parameters. There are, of course, statistical model selection problems to be faced within the kernel approach; in particular, the choice of the kernel and the corresponding feature space are central choices that must generally be made by a human user. While this provides opportunities for prior knowledge to be brought to bear, it can also be difficult in practice to find prior justification for the use of one kernel instead of another. It would be desirable to explore model selection methods that allow kernels to be chosen in a more automatic way based on data.

It is important to observe that we do not necessarily need to choose a kernel *function*, specifying the inner product between the images of all possible data points when mapped from an input space $\mathcal{X}$ to an appropriate feature space $\mathcal{F}$. Since kernel-based learning methods extract all information needed from inner products of training data points in $\mathcal{F}$, the values of the kernel function at pairs which are not present are irrelevant. So, there is no need to learn a kernel function over the entire sample space to specify the embedding of a finite training data set via a kernel function mapping. Instead, it is sufficient to specify a finite-dimensional *kernel matrix* (also known as a *Gram matrix*) that contains as its entries the inner products in $\mathcal{F}$ between all pairs of data points. Note also that it is possible to show that any symmetric positive semidefinite matrix is a valid Gram matrix, based on an inner product in some Hilbert space. This suggests viewing the model selection problem in terms of Gram matrices rather than kernel functions.

In this paper our main focus is *transduction*—the problem of completing the labeling of a partially labeled dataset. In other words, we are required to make predictions only at a finite set of points, which are specified a priori. Thus, instead of learning a function, we only need to learn a set of labels. There are many practical problems in which this formulation is natural—an example is the prediction of gene function, where the genes of interest are specified a priori, but the function of many of these genes is unknown.

We will address this problem by learning a kernel matrix corresponding to the entire dataset, a matrix that optimizes a certain cost function that depends on the available labels. In other words, we use the available labels to learn a good embedding, and we apply it to both the labeled and the unlabeled data. The resulting kernel matrix can then be used in combination with any of a number of existing learning algorithms that use kernels. One example that we discuss in detail is the support vector machine (SVM), where our methods yield a new transduction method for SVMs that scales polynomially with the number of test points. Furthermore, this approach will offer us a method to optimize the 2-norm soft margin parameter for these SVM learning algorithms, solving an important open problem.

All this can be done in full generality by using techniques from semidefinite programming (SDP), a branch of convex optimization that deals with the optimization of convex functions over the convex cone of positive semidefinite matrices, or convex subsets thereof. Any convex set of kernel matrices is a set of this kind. Furthermore, it turns out that many natural cost functions, motivated by error bounds, are convex in the kernel matrix.

A second application of the ideas that we present here is to the problem of combining data from multiple sources. Specifically, assume that each source is associated with a kernel function, such that a training set yields a set of kernel matrices. The tools that we develop in this paper make it possible to optimize over the coefficients in a linear combination of such kernel matrices. These coefficients can then be used to form linear combinations of kernel functions in the overall classifier. Thus this approach allows us to combine possibly heterogeneous data sources, making use of the reduction of heterogeneous data types to the common framework of kernel matrices, and choosing coefficients that emphasize those sources most useful in the classification decision.

In Section 2, we recall the main ideas from kernel-based learning algorithms, and introduce a variety of criteria that can be used to assess the suitability of a kernel matrix: the hard margin, the 1-norm and 2-norm soft margin, and the kernel alignment. Section 3 reviews the basic concepts of semidefinite programming. In Section 4 we put these ideas together and consider the optimization of the various criteria over sets of kernel matrices. For a set of linear combinations of fixed kernel matrices, these optimization problems reduce to SDP. If the linear coefficients are constrained to be positive, they can be simplified even further, yielding a quadratically-constrained quadratic program, a special case of the SDP framework. If the linear combination contains the identity matrix, we obtain a convex method for optimizing the 2-norm soft margin parameter in support vector machines. Section 5 presents statistical error bounds that motivate one of our cost functions. Empirical results are reported in Section 6.

Notation

Vectors are represented in bold notation, e.g., $\mathbf{v} \in \mathbb{R}^n$, and their scalar components in italic script, e.g., $v_1, v_2, \ldots, v_n$. Matrices are represented in italic script, e.g., $X \in \mathbb{R}^{m \times n}$. For a square, symmetric matrix X, $X \succeq 0$ means that X is positive semidefinite, while $X \succ 0$ means that X is positive definite. For a vector $\mathbf{v}$, the notations $\mathbf{v} \geq 0$ and $\mathbf{v} > 0$ are understood componentwise.

2. Kernel Methods

Kernel-based learning algorithms (see, for example, Cristianini and Shawe-Taylor, 2000; Schölkopf and Smola, 2002; Shawe-Taylor and Cristianini, 2004) work by embedding the data into a Hilbert space, and searching for linear relations in such a space. The embedding is performed implicitly, by specifying the inner product between each pair of points rather than by giving their coordinates explicitly. This approach has several advantages, the most important deriving from the fact that the inner product in the embedding space can often be computed much more easily than the coordinates of the points themselves.

Given an input set $\mathcal{X}$, and an embedding space $\mathcal{F}$, we consider a map $\Phi : \mathcal{X} \to \mathcal{F}$. Given two points $\mathbf{x}_i \in \mathcal{X}$ and $\mathbf{x}_j \in \mathcal{X}$, the function that returns the inner product between their images in the space $\mathcal{F}$ is known as the *kernel function.*

Definition 1 *A* kernel *is a function k, such that $k(\mathbf{x}, \mathbf{z}) = \langle \Phi(\mathbf{x}), \Phi(\mathbf{z}) \rangle$ for all $\mathbf{x}, \mathbf{z} \in \mathcal{X}$, where Φ is a mapping from $\mathcal{X}$ to an (inner product) feature space $\mathcal{F}$. A* kernel matrix *is a square matrix $K \in \mathbb{R}^{n \times n}$ such that $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$ for some $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathcal{X}$ and some kernel function k.*

The kernel matrix is also known as the Gram matrix. It is a symmetric, positive semidefinite matrix, and since it specifies the inner products between all pairs of points $\{\mathbf{x}_i\}_{i=1}^n$, it completely determines the relative positions of those points in the embedding space.

Since in this paper we will consider a *finite* input set $\mathcal{X}$, we can characterize kernel functions and matrices in the following simple way.

Proposition 2 *Every positive semidefinite and symmetric matrix is a kernel matrix. Conversely, every kernel matrix is symmetric and positive semidefinite.*

Notice that, if we have a kernel matrix, we do not need to know the kernel function, nor the implicitly defined map Φ, nor the coordinates of the points $\Phi(\mathbf{x}_i)$. We do not even need $\mathcal{X}$ to be a vector space; in fact in this paper it will be a generic finite set. We are guaranteed that the data are implicitly mapped to some Hilbert space by simply checking that the kernel matrix is symmetric and positive semidefinite.

The solutions sought by kernel-based algorithms such as the support vector machine (SVM) are affine functions in the feature space:

$$f(\mathbf{x}) = \langle \mathbf{w}, \Phi(\mathbf{x}) \rangle + b,$$

for some weight vector $\mathbf{w} \in \mathcal{F}$. The kernel can be exploited whenever the weight vector can be expressed as a linear combination of the training points, $\mathbf{w} = \sum_{i=1}^{n} \alpha_i \Phi(\mathbf{x}_i)$, implying that we can express f as

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

For example, for binary classification, we can use a thresholded version of $f(\mathbf{x})$, i.e., $\operatorname{sign}(f(\mathbf{x}))$, as a decision function to classify unlabeled data. If $f(\mathbf{x})$ is positive, then we classify $\mathbf{x}$ as belonging to class $+1$; otherwise, we classify $\mathbf{x}$ as belonging to class -1. An important issue in applications is that of choosing a kernel k for a given learning task; intuitively, we wish to choose a kernel that induces the "right" metric in the input space.

2.1 Criteria Used in Kernel Methods

Kernel methods choose a function that is linear in the feature space by optimizing some criterion over the sample. This section describes several such criteria (see, for example, Cristianini and Shawe-Taylor, 2000; Schölkopf and Smola, 2002; Shawe-Taylor and Cristianini, 2004). All of these criteria can be considered as measures of separation of the labeled data. We first consider the *hard margin* optimization problem.

Definition 3 Hard Margin *Given a labeled sample $S_l = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$, the hyperplane $(\mathbf{w}_*, b_*)$ that solves the optimization problem*

$$\begin{aligned} \min_{\mathbf{w}, b} \quad & \langle \mathbf{w}, \mathbf{w} \rangle \\ \textit{subject to} \quad & y_i(\langle \mathbf{w}, \Phi(\mathbf{x}_i) \rangle + b) \geq 1, \quad i = 1, \ldots, n, \end{aligned} \tag{1}$$

realizes the maximal margin classifier *with geometric margin $\gamma = 1/\|\mathbf{w}_*\|_2$, assuming it exists.*

Geometrically, γ corresponds to the distance between the convex hulls (the smallest convex sets that contain the data in each class) of the two classes (Bennett and Bredensteiner, 2000).

By transforming (1) into its corresponding Lagrangian dual problem, the solution is given by

$$\begin{aligned} \omega(K) &= 1/\gamma^2 \\ &= \langle \mathbf{w}_*, \mathbf{w}_* \rangle \\ &= \max_{\boldsymbol{\alpha}} \; 2\boldsymbol{\alpha}^T \mathbf{e} - \boldsymbol{\alpha}^T G(K) \boldsymbol{\alpha} \; : \; \boldsymbol{\alpha} \geq 0, \;\; \boldsymbol{\alpha}^T \mathbf{y} = 0, \end{aligned} \tag{2}$$

where $\mathbf{e}$ is the n-vector of ones, $\boldsymbol{\alpha} \in \mathbb{R}^n$, $G(K)$ is defined by $G_{ij}(K) = [K]_{ij} y_i y_j = k(\mathbf{x}_i, \mathbf{x}_j) y_i y_j$, and $\boldsymbol{\alpha} \geq 0$ means $\alpha_i \geq 0, \; i = 1, \ldots, n$.

The hard margin solution exists only when the labeled sample is linearly separable in feature space. For a non-linearly-separable labeled sample S_l, we can define the *soft margin*. We consider the 1-norm and 2-norm soft margins.

Definition 4 1-norm Soft Margin *Given a labeled sample* $S_l = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$, *the hyperplane* $(\mathbf{w}_*, b_*)$ *that solves the optimization problem*

$$\begin{aligned} \min_{\mathbf{w},b,\boldsymbol{\xi}} \quad & \langle \mathbf{w}, \mathbf{w} \rangle + C \sum_{i=1}^{n} \xi_i \\ \textit{subject to} \quad & y_i(\langle \mathbf{w}, \Phi(\mathbf{x}_i) \rangle + b) \geq 1 - \xi_i, \quad i = 1, \ldots, n \\ & \xi_i \geq 0, \quad i = 1, \ldots, n \end{aligned} \tag{3}$$

realizes the 1-norm soft margin classifier *with geometric margin* $\gamma = 1/\|\mathbf{w}_*\|_2$. *This margin is also called the 1-norm soft margin.*

As for the hard margin, we can express the solution of (3) in a revealing way by considering the corresponding Lagrangian dual problem:

$$\begin{aligned} \omega_{S1}(K) &= \langle \mathbf{w}_*, \mathbf{w}_* \rangle + C \sum_{i=1}^{n} \xi_{i,*} \\ &= \max_{\boldsymbol{\alpha}} \; 2\boldsymbol{\alpha}^T \mathbf{e} - \boldsymbol{\alpha}^T G(K) \boldsymbol{\alpha} \;:\; C \geq \boldsymbol{\alpha} \geq 0, \;\; \boldsymbol{\alpha}^T \mathbf{y} = 0. \end{aligned} \tag{4}$$

Definition 5 2-norm Soft Margin *Given a labeled sample* $S_l = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$, *the hyperplane* $(\mathbf{w}_*, b_*)$ *that solves the optimization problem*

$$\begin{aligned} \min_{\mathbf{w},b,\boldsymbol{\xi}} \quad & \langle \mathbf{w}, \mathbf{w} \rangle + C \sum_{i=1}^{n} \xi_i^2 \\ \textit{subject to} \quad & y_i(\langle \mathbf{w}, \Phi(\mathbf{x}_i) \rangle + b) \geq 1 - \xi_i, \quad i = 1, \ldots, n \\ & \xi_i \geq 0, \quad i = 1, \ldots, n \end{aligned} \tag{5}$$

realizes the 2-norm soft margin classifier *with geometric margin* $\gamma = 1/\|\mathbf{w}_*\|_2$. *This margin is also called the 2-norm soft margin.*

Again, by considering the corresponding dual problem, the solution of (5) can be expressed as

$$\begin{aligned} \omega_{S2}(K) &= \langle \mathbf{w}_*, \mathbf{w}_* \rangle + C \sum_{i=1}^{n} \xi_{i,*}^2 \\ &= \max_{\boldsymbol{\alpha}} \; 2\boldsymbol{\alpha}^T \mathbf{e} - \boldsymbol{\alpha}^T \left(G(K) + \frac{1}{C} I_n \right) \boldsymbol{\alpha} \;:\; \boldsymbol{\alpha} \geq 0, \;\; \boldsymbol{\alpha}^T \mathbf{y} = 0. \end{aligned} \tag{6}$$

With a fixed kernel, all of these criteria give upper bounds on misclassification probability (see, for example, Chapter 4 of Cristianini and Shawe-Taylor, 2000). Solving these optimization problems for a single kernel matrix is therefore a way of optimizing an upper bound on error probability.

In this paper, we allow the kernel matrix to be chosen from a class of kernel matrices. Previous error bounds are not applicable in this case. However, as we will see in Section 5, the margin γ can be used to bound the performance of support vector machines for transduction, with a linearly parameterized class of kernels.

We do not discuss further the merit of these different cost functions, deferring to the current literature on classification, where these cost functions are widely used with fixed kernels. Our goal is to show that these cost functions can be optimized—with respect to the kernel matrix—in an SDP setting.

Finally, we define the *alignment* of two kernel matrices (Cristianini et al., 2001, 2002). Given an (unlabeled) sample $S = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$, we use the following (Frobenius) inner product between Gram matrices, $\langle K_1, K_2 \rangle_F = \mathrm{trace}(K_1^T K_2) = \sum_{i,j=1}^n k_1(\mathbf{x}_i, \mathbf{x}_j) k_2(\mathbf{x}_i, \mathbf{x}_j)$.

Definition 6 Alignment *The* (empirical) alignment *of a kernel k_1 with a kernel k_2 with respect to the sample S is the quantity*

$$\hat{A}(S, k_1, k_2) = \frac{\langle K_1, K_2 \rangle_F}{\sqrt{\langle K_1, K_1 \rangle_F \langle K_2, K_2 \rangle_F}},$$

where K_i is the kernel matrix for the sample S using kernel k_i.

This can also be viewed as the cosine of the angle between two bi-dimensional vectors K_1 and K_2, representing the Gram matrices. Notice that we do not need to know the labels for the sample S in order to define the alignment of two kernels with respect to S. However, when the vector $\mathbf{y}$ of $\{\pm 1\}$ labels for the sample is known, we can consider $K_2 = \mathbf{y}\mathbf{y}^T$—the optimal kernel since $k_2(\mathbf{x}_i, \mathbf{x}_j) = 1$ if $y_i = y_j$ and $k_2(\mathbf{x}_i, \mathbf{x}_j) = -1$ if $y_i \neq y_j$. The alignment of a kernel k with k_2 with respect to S can be considered as a quality measure for k:

$$\hat{A}(S, K, \mathbf{y}\mathbf{y}^T) = \frac{\langle K, \mathbf{y}\mathbf{y}^T \rangle_F}{\sqrt{\langle K, K \rangle_F \langle \mathbf{y}\mathbf{y}^T, \mathbf{y}\mathbf{y}^T \rangle_F}} = \frac{\left\langle K, \mathbf{y}\mathbf{y}^T \right\rangle_F}{n\sqrt{\langle K, K \rangle_F}}, \tag{7}$$

since $\left\langle \mathbf{y}\mathbf{y}^T, \mathbf{y}\mathbf{y}^T \right\rangle_F = n^2$.

3. Semidefinite Programming (SDP)

In this section we review the basic definition of semidefinite programming as well as some important concepts and key results. Details and proofs can be found in Boyd and Vandenberghe (2003).

Semidefinite programming (Nesterov and Nemirovsky, 1994; Vandenberghe and Boyd, 1996; Boyd and Vandenberghe, 2003) deals with the optimization of convex functions over the convex cone[1] of symmetric, positive semidefinite matrices

$$\mathcal{P} = \left\{ X \in \mathbb{R}^{p \times p} \mid X = X^T, X \succeq 0 \right\},$$

or affine subsets of this cone. Given Proposition 2, $\mathcal{P}$ can be viewed as a search space for possible kernel matrices. This consideration leads to the key problem addressed in this paper—we wish to specify a convex cost function that will enable us to learn the optimal kernel matrix within $\mathcal{P}$ using semidefinite programming.

3.1 Definition of Semidefinite Programming

A *linear matrix inequality,* abbreviated LMI, is a constraint of the form

$$F(\mathbf{u}) := F_0 + u_1 F_1 + \ldots + u_q F_q \preceq 0.$$

Here, $\mathbf{u}$ is the vector of decision variables, and $F_0, \ldots, F_q$ are given symmetric $p \times p$ matrices. The notation $F(\mathbf{u}) \preceq 0$ means that the symmetric matrix F is negative semidefinite. Note that such a constraint is in general a *nonlinear* constraint; the term "linear" in the name LMI merely

1. $S \subseteq \mathbb{R}^d$ is a convex cone if and only if $\forall \mathbf{x}, \mathbf{y} \in S$ and $\forall \lambda, \mu \geq 0$, we have $\lambda \mathbf{x} + \mu \mathbf{y} \in S$.

emphasizes that F is affine in $\mathbf{u}$. Perhaps the most important feature of an LMI constraint is its convexity: the set of $\mathbf{u}$ that satisfy the LMI is a convex set.

An LMI constraint can be seen as an *infinite* set of scalar, affine constraints. Indeed, for a given $\mathbf{u}$, $F(\mathbf{u}) \preceq 0$ if and only if $\mathbf{z}^T F(\mathbf{u})\mathbf{z} \leq 0$ for every $\mathbf{z}$; every constraint indexed by $\mathbf{z}$ is an affine inequality, in the ordinary sense, i.e., the left-hand side of the inequality is a scalar, composed of a linear term in $\mathbf{u}$ and a constant term. Alternatively, using a standard result from linear algebra, we may state the constraint as

$$\forall Z \in \mathcal{P} \ : \ \mathrm{trace}(F(\mathbf{u})Z) \leq 0. \tag{8}$$

This can be seen by writing down the spectral decomposition of Z and using the fact that $\mathbf{z}^T F(\mathbf{u})\mathbf{z} \leq 0$ for every $\mathbf{z}$.

A semidefinite program (SDP) is an optimization problem with a linear objective, and linear matrix inequality and affine equality constraints.

Definition 7 *A semidefinite program is a problem of the form*

$$\begin{aligned} \min_{\mathbf{u}} \quad & \mathbf{c}^T\mathbf{u} \\ \textit{subject to} \quad & F^j(\mathbf{u}) = F_0^j + u_1 F_1^j + \ldots + u_q F_q^j \preceq 0, \quad j = 1, \ldots, L \\ & A\mathbf{u} = \mathbf{b}, \end{aligned} \tag{9}$$

where $\mathbf{u} \in \mathbb{R}^q$ is the vector of decision variables, $\mathbf{c} \in \mathbb{R}^q$ is the objective vector, and matrices $F_i^j = (F_i^j)^T \in \mathbb{R}^{p \times p}$ are given.

Given the convexity of its LMI constraints, SDPs are convex optimization problems. The usefulness of the SDP formalism stems from two important facts. First, despite the seemingly very specialized form of SDPs, they arise in a host of applications; second, there exist interior-point algorithms to solve SDPs that have good theoretical and practical computational efficiency (Vandenberghe and Boyd, 1996).

One very useful tool to reduce a problem to an SDP is the so-called Schur complement lemma; it will be invoked repeatedly.

Lemma 8 (Schur Complement Lemma) *Consider the partitioned symmetric matrix*

$$X = X^T = \begin{pmatrix} A & B \\ B^T & C \end{pmatrix},$$

where A, C are square and symmetric. If $\det(A) \neq 0$, we define the Schur complement of A in X by the matrix $S = C - B^T A^{-1} B$. The Schur Complement Lemma states that if $A \succ 0$, then $X \succeq 0$ if and only if $S \succeq 0$.

To illustrate how this lemma can be used to cast a nonlinear convex optimization problem as an SDP, consider the following result:

Lemma 9 *The quadratically constrained quadratic program (QCQP)*

$$\begin{aligned} \min_{\mathbf{u}} \quad & f_0(\mathbf{u}) \\ \textit{subject to} \quad & f_i(\mathbf{u}) \leq 0, \quad i = 1, \ldots, M, \end{aligned} \tag{10}$$

with $f_i(\mathbf{u}) \triangleq (A_i\mathbf{u}+\mathbf{b}_i)^T(A_i\mathbf{u}+\mathbf{b}_i) - \mathbf{c}_i^T\mathbf{u} - d_i$, *is equivalent to the semidefinite programming problem*

$$\begin{array}{ll} \min\limits_{\mathbf{u},t} & t \\ \textit{subject to} & \begin{pmatrix} I & A_0\mathbf{u}+\mathbf{b_0} \\ (A_0\mathbf{u}+\mathbf{b_0})^T & \mathbf{c_0}^T\mathbf{u}+d_0+t \end{pmatrix} \succeq 0, \\ & \begin{pmatrix} I & A_i\mathbf{u}+\mathbf{b}_i \\ (A_i\mathbf{u}+\mathbf{b}_i)^T & \mathbf{c}_i^T\mathbf{u}+d_i \end{pmatrix} \succeq 0, \quad i=1,\ldots,M. \end{array} \tag{11}$$

This can be seen by rewriting the QCQP (10) as

$$\begin{array}{ll} \min\limits_{\mathbf{u},t} & t \\ \text{subject to} & t - f_0(\mathbf{u}) \geq 0, \\ & -f_i(\mathbf{u}) \geq 0, \quad i=1,\ldots,M. \end{array}$$

Note that for a fixed and feasible $\mathbf{u}$, $t = f_0(\mathbf{u})$ is the optimal solution. The convex quadratic inequality $t - f_0(\mathbf{u}) = (t + \mathbf{c_0}^T\mathbf{u} + d_0) - (A_0\mathbf{u}+\mathbf{b_0})^T I^{-1}(A_0\mathbf{u}+\mathbf{b_0}) \geq 0$ is now equivalent to the following LMI, using the Schur Complement Lemma 8:

$$\begin{pmatrix} I & A_0\mathbf{u}+\mathbf{b_0} \\ (A_0\mathbf{u}+\mathbf{b_0})^T & \mathbf{c_0}^T\mathbf{u}+d_0+t \end{pmatrix} \succeq 0.$$

Similar steps for the other quadratic inequality constraints finally yield (11), an SDP in standard form (9), equivalent to (10). This shows that a QCQP can be cast as an SDP. Of course, in practice a QCQP should not be solved using general-purpose SDP solvers, since the particular structure of the problem at hand can be efficiently exploited. The above show that QCQPs, and in particular linear programming problems, belong to the SDP family.

3.2 Duality

An important principle in optimization—perhaps even the most important principle—is that of *duality*. To illustrate duality in the case of an SDP, we will first review basic concepts in duality theory and then show how they can be extended to semidefinite programming. In particular, duality will give insights into optimality conditions for the semidefinite program.

Consider an optimization problem with n variables and m scalar constraints:

$$\begin{array}{ll} \min\limits_{\mathbf{u}} & f_0(\mathbf{u}) \\ \text{subject to} & f_i(\mathbf{u}) \leq 0, \quad i=1,\ldots,m, \end{array} \tag{12}$$

where $\mathbf{u} \in \mathbb{R}^n$. In the context of duality, problem (12) is called the *primal problem*; we denote its optimal value p^*. For now, we do not assume convexity.

Definition 10 Lagrangian *The* Lagrangian $\mathcal{L} : \mathbb{R}^{n+m} \to \mathbb{R}$ *corresponding to the minimization problem (12) is defined as*

$$\mathcal{L}(\mathbf{u},\boldsymbol{\lambda}) = f_0(\mathbf{u}) + \lambda_1 f_1(\mathbf{u}) + \ldots + \lambda_m f_m(\mathbf{u}).$$

The $\lambda_i \in \mathbb{R},\ i=1,\ldots,m$ *are called* Lagrange multipliers *or* dual variables.

One can now notice that

$$h(\mathbf{u}) = \max_{\boldsymbol{\lambda} \geq 0} \mathcal{L}(\mathbf{u}, \boldsymbol{\lambda}) = \begin{cases} f_0(\mathbf{u}) & \text{if } f_i(\mathbf{u}) \leq 0, \ i = 1, \ldots, m \\ +\infty & \text{otherwise.} \end{cases}$$

So, the function $h(\mathbf{u})$ coincides with the objective $f_0(\mathbf{u})$ in regions where the constraints $f_i(\mathbf{u}) \leq 0$, $i = 1, \ldots, m$, are satisfied and $h(\mathbf{u}) = +\infty$ in infeasible regions. In other words, h acts as a "barrier" of the feasible set of the primal problem. Thus we can as well use $h(\mathbf{u})$ as objective function and rewrite the original primal problem (12) as an *unconstrained* optimization problem:

$$p^* = \min_{\mathbf{u}} \max_{\boldsymbol{\lambda} \geq 0} \mathcal{L}(\mathbf{u}, \boldsymbol{\lambda}). \tag{13}$$

The notion of weak duality amounts to exchanging the "min" and "max" operators in the above formulation, resulting in a lower bound on the optimal value of the primal problem. Strong duality refers to the case when this exchange can be done without altering the value of the result: the lower bound is actually equal to the optimal value p^*. While weak duality always hold, even if the primal problem (13) is not convex, strong duality may not hold. However, for a large class of generic convex problems, strong duality holds.

Lemma 11 Weak duality *For all functions $f_0, f_1, \ldots, f_m$ in (12), not necessarily convex, we can exchange the* max *and the* min *and get a lower bound on p^*:*

$$d^* = \max_{\boldsymbol{\lambda} \geq 0} \min_{\mathbf{u}} \mathcal{L}(\mathbf{u}, \boldsymbol{\lambda}) \leq \min_{\mathbf{u}} \max_{\boldsymbol{\lambda} \geq 0} \mathcal{L}(\mathbf{u}, \boldsymbol{\lambda}) = p^*.$$

The objective function of the maximization problem is now called the (Lagrange) dual function.

Definition 12 (Lagrange) dual function *The* (Lagrange) dual function $g : \mathbb{R}^m \to \mathbb{R}$ *is defined as*

$$\begin{aligned} g(\boldsymbol{\lambda}) &= \min_{\mathbf{u}} \mathcal{L}(\mathbf{u}, \boldsymbol{\lambda}) \\ &= \min_{\mathbf{u}} \ f_0(\mathbf{u}) + \lambda_1 f_1(\mathbf{u}) + \ldots + \lambda_m f_m(\mathbf{u}). \end{aligned} \tag{14}$$

Furthermore $g(\boldsymbol{\lambda})$ is concave, even if the $f_i(\mathbf{u})$ are not convex.

The concavity can easily be seen by considering first that for a given $\mathbf{u}$, $\mathcal{L}(\mathbf{u}, \boldsymbol{\lambda})$ is an affine function of $\boldsymbol{\lambda}$ and hence is a concave function. Since $g(\boldsymbol{\lambda})$ is the pointwise minimum of such concave functions, it is concave.

Definition 13 Lagrange dual problem *The* Lagrange dual problem *is defined as*

$$d^* = \max_{\boldsymbol{\lambda} \geq 0} g(\boldsymbol{\lambda}).$$

Since $g(\boldsymbol{\lambda})$ is concave, this will always be a convex optimization problem, even if the primal is not. By *weak duality*, we always have $d^* \leq p^*$, even for non-convex problems. The value $p^* - d^*$ is called the duality gap. For **convex** problems, we usually (although not always) have *strong duality* at the optimum, i.e.,

$$d^* = p^*,$$

which is also referred to as a *zero duality gap*. For convex problems, a sufficient condition for zero duality gap is provided by *Slater's condition*:

Lemma 14 Slater's condition *If the primal problem (12) is convex and is* **strictly feasible**, *i.e., $\exists\ \mathbf{u}_0 \ : \ f_i(\mathbf{u}_0) < 0, \ i = 1, \ldots, m$, then*

$$p^* = d^*.$$

3.3 SDP Duality and Optimality Conditions

Consider for simplicity the case of an SDP with a single LMI constraint, and no affine equalities:

$$p^* = \min_{\mathbf{u}} \mathbf{c}^T\mathbf{u} \text{ subject to } F(\mathbf{u}) = F_0 + u_1 F_1 + \ldots u_q F_q \preceq 0. \tag{15}$$

The general case of multiple LMI constraints and affine equalities can be handled by elimination of the latter and using block-diagonal matrices to represent the former as a single LMI.

The classical Lagrange duality theory outlined in the previous section does not directly apply here, since we are not dealing with finitely many constraints in scalar form; as noted earlier, the LMI constraint involves an infinite number of such constraints, of the form (8). One way to handle such constraints is to introduce a Lagrangian of the form

$$\mathcal{L}(\mathbf{u}, Z) = \mathbf{c}^T\mathbf{u} + \operatorname{trace}(ZF(\mathbf{u})),$$

where the dual variable Z is now a symmetric matrix, of the same size as $F(\mathbf{u})$. We can check that such a Lagrange function fulfills the same role assigned to the function defined in Definition 10 for the case with scalar constraints. Indeed, if we define $h(\mathbf{u}) = \max_{Z\succeq 0} \mathcal{L}(\mathbf{u}, Z)$ then

$$h(\mathbf{u}) = \max_{Z\succeq 0} \mathcal{L}(\mathbf{u}, Z) = \begin{cases} \mathbf{c}^T\mathbf{u} & \text{if } F(\mathbf{u}) \preceq 0, \\ +\infty & \text{otherwise.} \end{cases}$$

Thus, $h(\mathbf{u})$ is a barrier for the primal SDP (15), that is, it coincides with the objective of (15) on its feasible set, and is infinite otherwise. Notice that to the LMI constraint we now associate a multiplier *matrix*, which will be constrained to the positive semidefinite cone.

In the above, we made use of the fact that, for a given symmetric matrix F,

$$\phi(F) := \sup_{Z\succeq 0} \operatorname{trace}(ZF)$$

is $+\infty$ if F has a positive eigenvalue, and zero if F is negative semidefinite. This property is obvious for diagonal matrices, since in that case the variable Z can be constrained to be diagonal without loss of generality. The general case follows from the fact that if F has the eigenvalue decomposition $F = U\Lambda U^T$, where Λ is a diagonal matrix containing the eigenvalues of F, and U is orthogonal, then $\operatorname{trace}(ZF) = \operatorname{trace}(Z'\Lambda)$, where $Z' = U^T Z U$ spans the positive semidefinite cone whenever Z does.

Using the above Lagrangian, one can cast the original problem (15) as an unconstrained optimization problem:

$$p^* = \min_{\mathbf{u}} \max_{Z\succeq 0} \mathcal{L}(\mathbf{u}, Z).$$

By weak duality, we obtain a lower bound on p^* by exchanging the min and max:

$$d^* = \max_{Z\succeq 0} \min_{\mathbf{u}} \mathcal{L}(\mathbf{u}, Z) \leq \min_{\mathbf{u}} \max_{Z\succeq 0} \mathcal{L}(\mathbf{u}, Z) = p^*.$$

The inner minimization problem is easily solved analytically, due to the special structure of the SDP. We obtain a closed form for the (Lagrange) dual function:

$$\begin{aligned} g(Z) = \min_{\mathbf{u}} \mathcal{L}(\mathbf{u}, Z) &= \min_{\mathbf{u}} \mathbf{c}^T\mathbf{u} + \operatorname{trace}(ZF_0) + \sum_{i=1}^{q} u_i \operatorname{trace}(ZF_i) \\ &= \begin{cases} \operatorname{trace}(ZF_0) & \text{if } c_i = -\operatorname{trace}(ZF_i),\ i = 1, \ldots, q \\ -\infty & \text{otherwise.} \end{cases} \end{aligned}$$

The dual problem can be explicitly stated as follows:

$$d^* = \max_{Z \succeq 0} \min_{\mathbf{u}} \mathcal{L}(\mathbf{u}, Z) = \max_{Z} \operatorname{trace}(ZF_0) \text{ subject to } Z \succeq 0,\ c_i = -\operatorname{trace}(ZF_i),\ i = 1, \ldots, q. \quad (16)$$

We observe that the above problem is an SDP, with a single LMI constraint and q affine equalities in the matrix dual variable Z.

While weak duality always holds, strong duality may not, even for SDPs. Not surprisingly, a Slater-type condition ensures strong duality. Precisely, if the primal SDP (15) is strictly feasible, that is, there exists a $\mathbf{u}_0$ such that $F(\mathbf{u}_0) \prec 0$, then $p^* = d^*$. If, in addition, the dual problem is also strictly feasible, meaning that there exists a $Z \succ 0$ such that $c_i = \operatorname{trace}(ZF_i)$, $i = 1, \ldots, q$, then both primal and dual optimal values are attained by some optimal pair $(\mathbf{u}^*, Z^*)$. In that case, we can characterize such optimal pairs as follows. In view of the equality constraints of the dual problem, the duality gap can be expressed as

$$\begin{aligned} p^* - d^* &= \mathbf{c}^T\mathbf{u}^* - \operatorname{trace}(Z^* F_0) \\ &= -\operatorname{trace}(Z^* F(\mathbf{u}^*)). \end{aligned}$$

A zero duality gap is equivalent to $\operatorname{trace}(Z^* F(\mathbf{u}^*)) = 0$, which in turn is equivalent to $Z^* F(\mathbf{u}^*) = O$, where O denotes the zero matrix, since the product of a positive semidefinite and a negative semidefinite matrix has zero trace if and only if it is zero.

To summarize, consider the SDP (15) and its Lagrange dual (16). If either problem is strictly feasible, then they share the same optimal value. If both problems are strictly feasible, then the optimal values of both problems are attained and coincide. In this case, a primal-dual pair $(\mathbf{u}^*, Z^*)$ is optimal if and only if

$$\begin{aligned} &F(\mathbf{u}^*) \preceq 0, \\ &Z^* \succeq 0, \\ &c_i = -\operatorname{trace}(Z^* F_i), \quad i = 1, \ldots, q, \\ &Z^* F(\mathbf{u}^*) = O. \end{aligned}$$

The above conditions represent the expression of the general Karush-Kuhn-Tucker (KKT) conditions in the semidefinite programming setting. The first three sets of conditions express that $\mathbf{u}^*$ and Z^* are feasible for their respective problems; the last condition expresses a complementarity condition.

For a pair of strictly feasible primal-dual SDPs, solving the primal minimization problem is equivalent to maximizing the dual problem and both can thus be considered simultaneously. Algorithms indeed make use of this relationship and use the duality gap as a stopping criterion. A general-purpose program such as SeDuMi (Sturm, 1999) handles those problems efficiently. This code uses interior-point methods for SDP (Nesterov and Nemirovsky, 1994); these methods have a worst-case complexity of $O(q^2 p^{2.5})$ for the general problem (15). In practice, problem structure can be exploited for great computational savings: e.g., when $F(\mathbf{u}) \in \mathbb{R}^{p \times p}$ consists of L diagonal blocks of size p_i, $i = 1, \ldots, L$, these methods have a worst-case complexity of $O(q^2 (\sum_{i=1}^L p_i^2) p^{0.5})$ (Vandenberghe and Boyd, 1996).

4. Algorithms for Learning Kernels

We work in a transduction setting, where some of the data (the training set $S_{n_{tr}} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_{n_{tr}}, y_{n_{tr}})\}$) are labeled, and the remainder (the test set $T_{n_t} = \{\mathbf{x}_{n_{tr}+1}, \ldots, \mathbf{x}_{n_{tr}+n_t}\}$) are unlabeled, and the aim is to predict the labels of the test data. In this setting, optimizing the kernel corresponds to choosing a kernel matrix. This matrix has the form

$$K = \begin{pmatrix} K_{tr} & K_{tr,t} \\ K_{tr,t}^T & K_t \end{pmatrix}, \tag{17}$$

where $K_{ij} = \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle$, $i, j = 1, \ldots, n_{tr}, n_{tr} + 1, \ldots, n_{tr} + n_t$. By optimizing a cost function over the "training-data block" K_{tr}, we want to learn the optimal mixed block $K_{tr,t}$ and the optimal "test-data block" K_t.

This implies that training and test-data blocks must somehow be entangled: tuning training-data entries in K (to optimize their embedding) should imply that test-data entries are automatically tuned in some way as well. This can be achieved by constraining the search space of possible kernel matrices: we control the capacity of the search space of possible kernel matrices in order to prevent overfitting and achieve good generalization on test data.

We first consider a general optimization problem in which the kernel matrix K is restricted to a convex subset $\mathcal{K}$ of $\mathcal{P}$, the positive semidefinite cone. We then consider two specific examples. The first is the set of positive semidefinite matrices with bounded trace that can be expressed as a linear combination of kernel matrices from the set $\{K_1, \ldots, K_m\}$. That is, $\mathcal{K}$ is the set of matrices K satisfying

$$\begin{aligned} K &= \sum_{i=1}^{m} \mu_i K_i, \\ K &\succeq 0, \\ \mathrm{trace}(K) &\leq c. \end{aligned} \tag{18}$$

In this case, the set $\mathcal{K}$ lies in the intersection of a low-dimensional linear subspace with the positive semidefinite cone $\mathcal{P}$. Geometrically this can be viewed as computing all embeddings (for every K_i), in disjoint feature spaces, and then weighting these. The set $\{K_1, \ldots, K_m\}$ could be a set of initial "guesses" of the kernel matrix, e.g., linear, Gaussian or polynomial kernels with different kernel parameter values. Instead of fine-tuning the kernel parameter for a given kernel using cross-validation, one can now evaluate the given kernel for a range of kernel parameters and then optimize the weights in the linear combination of the obtained kernel matrices. Alternatively, the K_i could be chosen as the rank-one matrices $K_i = \mathbf{v}_i \mathbf{v}_i^T$, with $\mathbf{v}_i$ a subset of the eigenvectors of K_0, an initial kernel matrix, or with $\mathbf{v}_i$ some other set of orthogonal vectors. A practically important form is the case in which a diverse set of possibly good Gram matrices K_i (similarity measures/representations) has been constructed, e.g., using heterogeneous data sources. The challenge is to combine these measures into one optimal similarity measure (embedding), to be used for learning.

The second example of a restricted set $\mathcal{K}$ of kernels is the set of positive semidefinite matrices with bounded trace that can be expressed as a linear combination of kernel matrices from the set $\{K_1, \ldots, K_m\}$, but with the parameters μ_i constrained to be non-negative. That is, $\mathcal{K}$ is the set of matrices K satisfying

$$\begin{aligned} K &= \sum_{i=1}^{m} \mu_i K_i, \\ \mu_i &\geq 0 \qquad i \in \{1, \ldots, m\} \\ K &\succeq 0, \\ \mathrm{trace}(K) &\leq c. \end{aligned}$$

This further constrains the class of functions that can be represented. It has two advantages: we shall see that the corresponding optimization problem has significantly reduced computational complexity, and it is more convenient for studying the statistical properties of a class of kernel matrices.

As we will see in Section 5, we can estimate the performance of support vector machines for transduction using properties of the class $\mathcal{K}$. As explained in Section 2, we can use a thresholded version of $f(\mathbf{x})$, i.e., $\text{sign}\,(f(\mathbf{x}))$, as a binary classification decision. Using this decision function, we will prove that the proportion of errors on the test data T_n (where, for convenience, we suppose that training and test data have the same size $n_{tr} = n_t = n$) is, with probability $1 - \delta$ (over the random draw of the training set S_n and test set T_n), bounded by

$$\frac{1}{n}\sum_{i=1}^{n} \max\{1 - y_i f(\mathbf{x}_i), 0\} + \frac{1}{\sqrt{n}}\left(4 + \sqrt{2\log(1/\delta)} + \sqrt{\frac{\mathcal{C}(\mathcal{K})}{n\gamma^2}}\right), \tag{19}$$

where γ is the 1-norm soft margin on the data and $\mathcal{C}(\mathcal{K})$ is a certain measure of the complexity of the kernel class $\mathcal{K}$. For instance, for the class $\mathcal{K}$ of positive linear combinations defined above, $\mathcal{C}(\mathcal{K}) \leq mc$, where m is the number of kernel matrices in the combination and c is the bound on the trace. So, the proportion of errors on the test data is bounded by the average error on the training set and a complexity term, determined by the richness of the class $\mathcal{K}$ and the margin γ. Good generalization can thus be expected if the error on the training set is small, while having a large margin and a class $\mathcal{K}$ that is not too rich.

The next section presents the main optimization result of the paper: minimizing a generalized performance measure $\omega_{C,\tau}(K)$ with respect to the kernel matrix K can be realized in a semidefinite programming framework. Afterwards, we prove a second general result showing that minimizing $\omega_{C,\tau}(K)$ with respect to a kernel matrix K, constrained to the linear subspace $K = \sum_{i=1}^{m} \mu_i K_i$ with $\boldsymbol{\mu} \geq 0$, leads to a quadratically constrained quadratic programming (QCQP) problem. Maximizing the margin of a hard margin SVM with respect to K, as well as both soft margin cases can then be treated as specific instances of this general result and will be discussed in later sections.

4.1 General Optimization Result

In this section, we first of all show that minimizing the generalized performance measure

$$\omega_{C,\tau}(K) = \max_{\boldsymbol{\alpha}} \; 2\boldsymbol{\alpha}^T\mathbf{e} - \boldsymbol{\alpha}^T(G(K) + \tau I)\boldsymbol{\alpha} \; : \; C \geq \boldsymbol{\alpha} \geq 0, \;\; \boldsymbol{\alpha}^T\mathbf{y} = 0, \tag{20}$$

with $\tau \geq 0$, on the training data with respect to the kernel matrix K, in some convex subset $\mathcal{K}$ of positive semidefinite matrices with trace equal to c,

$$\min_{K \in \mathcal{K}} \omega_{C,\tau}(K_{tr}) \qquad \text{s.t. trace}(K) = c, \tag{21}$$

can be realized in a semidefinite programming framework.

We first note a fundamental property of the generalized performance measure, a property that is crucial for the remainder of the paper.

Proposition 15 *The quantity*

$$\omega_{C,\tau}(K) = \max_{\boldsymbol{\alpha}} \; 2\boldsymbol{\alpha}^T\mathbf{e} - \boldsymbol{\alpha}^T(G(K) + \tau I)\boldsymbol{\alpha} \; : \; C \geq \boldsymbol{\alpha} \geq 0, \;\; \boldsymbol{\alpha}^T\mathbf{y} = 0,$$

is convex in K.

This is easily seen by considering first that $2\boldsymbol{\alpha}^T\mathbf{e} - \boldsymbol{\alpha}^T(G(K)+\tau I)\boldsymbol{\alpha}$ is an affine function of K, and hence is a convex function as well. Secondly, we notice that $\omega_{C,\tau}(K)$ is the pointwise maximum of such convex functions and is thus convex. The constraints $C \geq \boldsymbol{\alpha} \geq 0, \;\; \boldsymbol{\alpha}^T\mathbf{y} = 0$ are obviously convex.

Problem (21) is now a convex optimization problem. The following theorem shows that, for a suitable choice of the set $\mathcal{K}$, this problem can be cast as an SDP.

Theorem 16 *Given a labeled sample $S_{n_{tr}} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_{n_{tr}}, y_{n_{tr}})\}$ with the set of labels denoted $\mathbf{y} \in \mathbb{R}^{n_{tr}}$, the kernel matrix $K \in \mathcal{K}$ that optimizes (21), with $\tau \geq 0$, can be found by solving the following convex optimization problem:*

$$\begin{aligned} \min_{K,t,\lambda,\boldsymbol{\nu},\boldsymbol{\delta}} \quad & t \\ \textit{subject to} \quad & \mathit{trace}(K) = c, \\ & K \in \mathcal{K}, \\ & \begin{pmatrix} G(K_{tr}) + \tau I_{n_{tr}} & \mathbf{e}+\boldsymbol{\nu}-\boldsymbol{\delta}+\lambda\mathbf{y} \\ (\mathbf{e}+\boldsymbol{\nu}-\boldsymbol{\delta}+\lambda\mathbf{y})^T & t - 2C\boldsymbol{\delta}^T\mathbf{e} \end{pmatrix} \succeq 0, \\ & \boldsymbol{\nu} \geq 0, \\ & \boldsymbol{\delta} \geq 0. \end{aligned} \tag{22}$$

Proof We begin by substituting $\omega_{C,\tau}(K_{tr})$, as defined in (20), into (21), which yields

$$\min_{K\in\mathcal{K}} \max_{\boldsymbol{\alpha}} \; 2\boldsymbol{\alpha}^T\mathbf{e} - \boldsymbol{\alpha}^T(G(K_{tr}) + \tau I_{n_{tr}})\boldsymbol{\alpha} \;:\; C \geq \boldsymbol{\alpha} \geq 0, \;\; \boldsymbol{\alpha}^T\mathbf{y} = 0, \;\; \mathrm{trace}(K) = c, \tag{23}$$

with c a constant. Assume that $K_{tr} \succ 0$, hence $G(K_{tr}) \succ 0$ and $G(K_{tr}) + \tau I_{n_{tr}} \succ 0$ since $\tau \geq 0$ (the following can be extended to the general semidefinite case). From Proposition 15, we know that $\omega_{C,\tau}(K_{tr})$ is convex in K_{tr} and thus in K. Given the convex constraints in (23), the optimization problem is thus certainly convex in K. We write this as

$$\begin{aligned} \min_{K\in\mathcal{K},t} t \;\; : \quad & t \geq \max_{\boldsymbol{\alpha}} \; 2\boldsymbol{\alpha}^T\mathbf{e} - \boldsymbol{\alpha}^T(G(K_{tr}) + \tau I_{n_{tr}})\boldsymbol{\alpha}, \\ & C \geq \boldsymbol{\alpha} \geq 0, \;\; \boldsymbol{\alpha}^T\mathbf{y} = 0, \;\; \mathrm{trace}(K) = c. \end{aligned} \tag{24}$$

We now express the constraint $t \geq \max_{\boldsymbol{\alpha}} \; 2\boldsymbol{\alpha}^T\mathbf{e} - \boldsymbol{\alpha}^T(G(K_{tr}) + \tau I_{n_{tr}})\boldsymbol{\alpha}$ as an LMI using duality. In particular, duality will allow us to drop the minimization and the Schur complement lemma then yields an LMI.

Define the Lagrangian of the maximization problem (20) by

$$\mathcal{L}(\boldsymbol{\alpha}, \boldsymbol{\nu}, \lambda, \boldsymbol{\delta}) = 2\boldsymbol{\alpha}^T\mathbf{e} - \boldsymbol{\alpha}^T(G(K_{tr}) + \tau I_{n_{tr}})\boldsymbol{\alpha} + 2\boldsymbol{\nu}^T\boldsymbol{\alpha} + 2\lambda\mathbf{y}^T\boldsymbol{\alpha} + 2\boldsymbol{\delta}^T(C\mathbf{e} - \boldsymbol{\alpha}),$$

where $\lambda \in \mathbb{R}$ and $\boldsymbol{\nu}, \boldsymbol{\delta} \in \mathbb{R}^{n_{tr}}$. By duality, we have

$$\omega_{C,\tau}(K_{tr}) = \max_{\boldsymbol{\alpha}} \min_{\boldsymbol{\nu}\geq 0, \boldsymbol{\delta}\geq 0, \lambda} \mathcal{L}(\boldsymbol{\alpha}, \boldsymbol{\nu}, \lambda, \boldsymbol{\delta}) = \min_{\boldsymbol{\nu}\geq 0, \boldsymbol{\delta}\geq 0, \lambda} \max_{\boldsymbol{\alpha}} \mathcal{L}(\boldsymbol{\alpha}, \boldsymbol{\nu}, \lambda, \boldsymbol{\delta}).$$

Since $G(K_{tr}) + \tau I_{n_{tr}} \succ 0$, at the optimum we have

$$\boldsymbol{\alpha} = (G(K_{tr}) + \tau I_{n_{tr}})^{-1}(\mathbf{e} + \boldsymbol{\nu} - \boldsymbol{\delta} + \lambda\mathbf{y}),$$

and we can form the dual problem

$$\omega_{C,\tau}(K_{tr}) = \min_{\boldsymbol{\nu},\boldsymbol{\delta},\lambda} \; (\mathbf{e}+\boldsymbol{\nu}-\boldsymbol{\delta}+\lambda\mathbf{y})^T(G(K_{tr}) + \tau I_{n_{tr}})^{-1}(\mathbf{e}+\boldsymbol{\nu}-\boldsymbol{\delta}+\lambda\mathbf{y}) + 2C\boldsymbol{\delta}^T\mathbf{e} \;:\; \boldsymbol{\nu} \geq 0, \boldsymbol{\delta} \geq 0.$$

This implies that for any $t > 0$, the constraint $\omega_{C,\tau}(K_{tr}) \leq t$ holds if and only if there exist $\boldsymbol{\nu} \geq 0, \boldsymbol{\delta} \geq 0$ and λ such that

$$(\mathbf{e}+\boldsymbol{\nu}-\boldsymbol{\delta}+\lambda\mathbf{y})^T(G(K_{tr})+\tau I_{n_{tr}})^{-1}(\mathbf{e}+\boldsymbol{\nu}-\boldsymbol{\delta}+\lambda\mathbf{y})+2C\boldsymbol{\delta}^T\mathbf{e} \leq t,$$

or, equivalently (using the Schur complement lemma), such that

$$\begin{pmatrix} G(K_{tr})+\tau I_{n_{tr}} & \mathbf{e}+\boldsymbol{\nu}-\boldsymbol{\delta}+\lambda\mathbf{y} \\ (\mathbf{e}+\boldsymbol{\nu}-\boldsymbol{\delta}+\lambda\mathbf{y})^T & t-2C\boldsymbol{\delta}^T\mathbf{e} \end{pmatrix} \succeq 0$$

holds. Taking this into account, (24) can be expressed as

$$\begin{aligned} \min_{K,t,\lambda,\boldsymbol{\nu},\boldsymbol{\delta}} \quad & t \\ \text{subject to} \quad & \operatorname{trace}(K)=c, \\ & K \in \mathcal{K}, \\ & \begin{pmatrix} G(K_{tr})+\tau I_{n_{tr}} & \mathbf{e}+\boldsymbol{\nu}-\boldsymbol{\delta}+\lambda\mathbf{y} \\ (\mathbf{e}+\boldsymbol{\nu}-\boldsymbol{\delta}+\lambda\mathbf{y})^T & t-2C\boldsymbol{\delta}^T\mathbf{e} \end{pmatrix} \succeq 0, \\ & \boldsymbol{\nu} \geq 0, \\ & \boldsymbol{\delta} \geq 0, \end{aligned}$$

which yields (22). Notice that $\boldsymbol{\nu} \geq 0 \Leftrightarrow \operatorname{diag}(\boldsymbol{\nu}) \succeq 0$, and is thus an LMI; similarly for $\boldsymbol{\delta} \geq 0$. ■

Notice that if $\mathcal{K} = \{K \succeq 0\}$, this optimization problem is an SDP in the standard form (9). Of course, in that case there is no constraint to ensure entanglement of training and test-data blocks. Indeed, it is easy to see that the criterion would be optimized with a test matrix $K_t = O$.

Consider the constraint $\mathcal{K} = \operatorname{span}\{K_1, \ldots, K_m\} \cap \{K \succeq 0\}$. We obtain the following convex optimization problem:

$$\begin{aligned} \min_{K} \quad & \omega_{C,\tau}(K_{tr}) \\ \text{subject to} \quad & \operatorname{trace}(K)=c, \\ & K \succeq 0, \\ & K = \sum_{i=1}^m \mu_i K_i, \end{aligned} \tag{25}$$

which can be written in the standard form of a semidefinite program, in a manner analogous to (22):

$$\begin{aligned} \min_{\boldsymbol{\mu},t,\lambda,\boldsymbol{\nu},\boldsymbol{\delta}} \quad & t \\ \text{subject to} \quad & \operatorname{trace}\left(\sum_{i=1}^m \mu_i K_i\right)=c, \\ & \sum_{i=1}^m \mu_i K_i \succeq 0, \\ & \begin{pmatrix} G(\sum_{i=1}^m \mu_i K_{i,tr})+\tau I_{n_{tr}} & \mathbf{e}+\boldsymbol{\nu}-\boldsymbol{\delta}+\lambda\mathbf{y} \\ (\mathbf{e}+\boldsymbol{\nu}-\boldsymbol{\delta}+\lambda\mathbf{y})^T & t-2C\boldsymbol{\delta}^T\mathbf{e} \end{pmatrix} \succeq 0, \\ & \boldsymbol{\nu} \geq 0, \\ & \boldsymbol{\delta} \geq 0. \end{aligned} \tag{26}$$

To solve this general optimization problem, one has to solve a semidefinite programming problem. General-purpose programs such as SeDuMi (Sturm, 1999) use interior-point methods to solve SDP problems (Nesterov and Nemirovsky, 1994). These methods are polynomial time. However, applying the complexity results mentioned in Section 3.3 leads to a worst-case complexity $O\big((m+n_{tr})^2(n^2+n_{tr}^2)(n+n_{tr})^{0.5}\big)$, or roughly $O\left((m+n_{tr})^2 n^{2.5}\right)$, in this particular case.

Consider a further restriction on the set of kernel matrices, where the matrices are restricted to positive linear combinations of kernel matrices $\{K_1,\ldots,K_m\}\cap\{K\succeq 0\}$:

$$K=\sum_{i=1}^m \mu_i K_i, \qquad \boldsymbol{\mu}\geq 0.$$

For this restricted linear subspace of the positive semidefinite cone $\mathcal{P}$, we can prove the following theorem:

Theorem 17 *Given a labeled sample $S_{n_{tr}}=\{(\mathbf{x}_1,y_1),\ldots,(\mathbf{x}_{n_{tr}},y_{n_{tr}})\}$ with the set of labels denoted $\mathbf{y}\in\mathbb{R}^{n_{tr}}$, the kernel matrix $K=\sum_{i=1}^m\mu_iK_i$ that optimizes (21), with $\tau\geq 0$, under the additional constraint $\boldsymbol{\mu}\geq 0$ can be found by solving the following convex optimization problem, and considering its dual solution:*

$$\begin{aligned} \max_{\boldsymbol{\alpha},t} \quad & 2\boldsymbol{\alpha}^T\mathbf{e}-\tau\boldsymbol{\alpha}^T\boldsymbol{\alpha}-ct \\ \textit{subject to} \quad & t\geq \frac{1}{r_i}\boldsymbol{\alpha}^T G(K_{i,tr})\boldsymbol{\alpha}, \quad i=1,\ldots,m \\ & \boldsymbol{\alpha}^T\mathbf{y}=0, \\ & C\geq\boldsymbol{\alpha}\geq 0, \end{aligned} \tag{27}$$

where $\mathbf{r}\in\mathbb{R}^m$ with trace$(K_i)=r_i$.

Proof Solving problem (21) subject to $K=\sum_{i=1}^m\mu_iK_i$, with $K_i\succeq 0$, and the extra constraint $\boldsymbol{\mu}\geq 0$ yields

$$\begin{aligned} \min_K \max_{\boldsymbol{\alpha}\,:\,C\geq\boldsymbol{\alpha}\geq 0,\boldsymbol{\alpha}^T\mathbf{y}=0} \quad & 2\boldsymbol{\alpha}^T\mathbf{e}-\boldsymbol{\alpha}^T(G(K_{tr})+\tau I_{n_{tr}})\boldsymbol{\alpha} \\ \text{subject to} \quad & \text{trace}(K)=c, \\ & K\succeq 0, \\ & K=\sum_{i=1}^m\mu_iK_i, \\ & \boldsymbol{\mu}\geq 0, \end{aligned}$$

when $\omega_{C,\tau}(K_{tr})$ is expressed using (20). We can omit the second constraint, because this is implied by the last two constraints, if $K_i\succeq 0$. The problem then reduces to

$$\begin{aligned} \min_{\boldsymbol{\mu}} \max_{\boldsymbol{\alpha}\,:\,C\geq\boldsymbol{\alpha}\geq 0,\boldsymbol{\alpha}^T\mathbf{y}=0} \quad & 2\boldsymbol{\alpha}^T\mathbf{e}-\boldsymbol{\alpha}^T(G(\sum_{i=1}^m\mu_iK_{i,tr})+\tau I_{n_{tr}})\boldsymbol{\alpha} \\ \text{subject to} \quad & \boldsymbol{\mu}^T\mathbf{r}=c, \\ & \boldsymbol{\mu}\geq 0, \end{aligned}$$

where $K_{i,tr} = K_i(1:n_{tr}, 1:n_{tr})$. We can write this as

$$\begin{aligned}
&\min_{\boldsymbol{\mu}\,:\,\boldsymbol{\mu}\geq 0,\boldsymbol{\mu}^T\mathbf{r}=c} \quad \max_{\boldsymbol{\alpha}\,:\,C\geq\boldsymbol{\alpha}\geq 0,\boldsymbol{\alpha}^T\mathbf{y}=0} 2\boldsymbol{\alpha}^T\mathbf{e} - \boldsymbol{\alpha}^T\left(\operatorname{diag}(\mathbf{y})(\sum_{i=1}^m \mu_i K_{i,tr})\operatorname{diag}(\mathbf{y}) + \tau I_{n_{tr}}\right)\boldsymbol{\alpha} \\
&= \min_{\boldsymbol{\mu}\,:\,\boldsymbol{\mu}\geq 0,\boldsymbol{\mu}^T\mathbf{r}=c} \quad \max_{\boldsymbol{\alpha}\,:\,C\geq\boldsymbol{\alpha}\geq 0,\boldsymbol{\alpha}^T\mathbf{y}=0} 2\boldsymbol{\alpha}^T\mathbf{e} - \sum_{i=1}^m \mu_i \boldsymbol{\alpha}^T\operatorname{diag}(\mathbf{y})K_{i,tr}\operatorname{diag}(\mathbf{y})\boldsymbol{\alpha} - \tau\boldsymbol{\alpha}^T\boldsymbol{\alpha} \\
&= \min_{\boldsymbol{\mu}\,:\,\boldsymbol{\mu}\geq 0,\boldsymbol{\mu}^T\mathbf{r}=c} \quad \max_{\boldsymbol{\alpha}\,:\,C\geq\boldsymbol{\alpha}\geq 0,\boldsymbol{\alpha}^T\mathbf{y}=0} 2\boldsymbol{\alpha}^T\mathbf{e} - \sum_{i=1}^m \mu_i \boldsymbol{\alpha}^T G(K_{i,tr})\boldsymbol{\alpha} - \tau\boldsymbol{\alpha}^T\boldsymbol{\alpha} \\
&= \max_{\boldsymbol{\alpha}\,:\,C\geq\boldsymbol{\alpha}\geq 0,\boldsymbol{\alpha}^T\mathbf{y}=0} \quad \min_{\boldsymbol{\mu}\,:\,\boldsymbol{\mu}\geq 0,\boldsymbol{\mu}^T\mathbf{r}=c} 2\boldsymbol{\alpha}^T\mathbf{e} - \sum_{i=1}^m \mu_i \boldsymbol{\alpha}^T G(K_{i,tr})\boldsymbol{\alpha} - \tau\boldsymbol{\alpha}^T\boldsymbol{\alpha},
\end{aligned}$$

with $G(K_{i,tr}) = \operatorname{diag}(\mathbf{y})K_{i,tr}\operatorname{diag}(\mathbf{y})$. The interchange of the order of the minimization and the maximization is justified (see, e.g., Boyd and Vandenberghe, 2003) because the objective is convex in $\boldsymbol{\mu}$ (it is linear in $\boldsymbol{\mu}$) and concave in $\boldsymbol{\alpha}$, because the minimization problem is strictly feasible in $\boldsymbol{\mu}$, and the maximization problem is strictly feasible in $\boldsymbol{\alpha}$ (we can skip the case for all elements of $\mathbf{y}$ having the same sign, because we cannot even define a margin in such a case). We thus obtain

$$\begin{aligned}
&\max_{\boldsymbol{\alpha}\,:\,C\geq\boldsymbol{\alpha}\geq 0,\boldsymbol{\alpha}^T\mathbf{y}=0} \quad \min_{\boldsymbol{\mu}\,:\,\boldsymbol{\mu}\geq 0,\boldsymbol{\mu}^T\mathbf{r}=c} 2\boldsymbol{\alpha}^T\mathbf{e} - \sum_{i=1}^m \mu_i \boldsymbol{\alpha}^T G(K_{i,tr})\boldsymbol{\alpha} - \tau\boldsymbol{\alpha}^T\boldsymbol{\alpha} \\
&= \max_{\boldsymbol{\alpha}\,:\,C\geq\boldsymbol{\alpha}\geq 0,\boldsymbol{\alpha}^T\mathbf{y}=0} \left[2\boldsymbol{\alpha}^T\mathbf{e} - \tau\boldsymbol{\alpha}^T\boldsymbol{\alpha} - \max_{\boldsymbol{\mu}\,:\,\boldsymbol{\mu}\geq 0,\boldsymbol{\mu}^T\mathbf{r}=c}\left(\sum_{i=1}^m \mu_i \boldsymbol{\alpha}^T G(K_{i,tr})\boldsymbol{\alpha}\right)\right] \\
&= \max_{\boldsymbol{\alpha}\,:\,C\geq\boldsymbol{\alpha}\geq 0,\boldsymbol{\alpha}^T\mathbf{y}=0} \left[2\boldsymbol{\alpha}^T\mathbf{e} - \tau\boldsymbol{\alpha}^T\boldsymbol{\alpha} - \max_i\left(\frac{c}{r_i}\boldsymbol{\alpha}^T G(K_{i,tr})\boldsymbol{\alpha}\right)\right].
\end{aligned}$$

Finally, this can be reformulated as

$$\begin{aligned}
\max_{\boldsymbol{\alpha},t} \quad & 2\boldsymbol{\alpha}^T\mathbf{e} - \tau\boldsymbol{\alpha}^T\boldsymbol{\alpha} - ct \\
\text{subject to} \quad & t \geq \frac{1}{r_i}\boldsymbol{\alpha}^T G(K_{i,tr})\boldsymbol{\alpha}, \quad i = 1,\ldots,m \\
& \boldsymbol{\alpha}^T\mathbf{y} = 0, \\
& C \geq \boldsymbol{\alpha} \geq 0,
\end{aligned}$$

which proves the theorem. ■

This convex optimization problem, a QCQP more precisely, is a special instance of an SOCP (second-order cone programming problem), which is in turn a special form of SDP (Boyd and Vandenberghe, 2003). SOCPs can be solved efficiently with programs such as SeDuMi (Sturm, 1999) or Mosek (Andersen and Andersen, 2000). These codes use interior-point methods (Nesterov and Nemirovsky, 1994) which yield a worst-case complexity of $O(mn_{tr}^3)$. This implies a major improvement compared to the worst-case complexity of a general SDP. Furthermore, the codes simultaneously solve the above problem and its dual form. They thus return optimal values for the dual variables as well—this allows us to obtain the optimal weights μ_i, for $i = 1,\ldots,m$.

4.2 Hard Margin

In this section, we show how maximizing the margin of a hard margin SVM with respect to the kernel matrix can be realized in the semidefinite programming framework derived in Theorem 16.

Inspired by (19), let us try to find the kernel matrix K in some convex subset $\mathcal{K}$ of positive semidefinite matrices for which the corresponding embedding shows maximal margin on the training data, keeping the trace of K constant:

$$\min_{K \in \mathcal{K}} \omega(K_{tr}) \qquad \text{s.t. } \operatorname{trace}(K) = c. \tag{28}$$

Note that $\omega(K_{tr}) = \omega_{\infty,0}(K_{tr})$. From Proposition 15, we then obtain the following important result:

Corollary 18 *The quantity*

$$\omega(K) = \max_{\boldsymbol{\alpha}} \; 2\boldsymbol{\alpha}^T \mathbf{e} - \boldsymbol{\alpha}^T G(K) \boldsymbol{\alpha} \;:\; \boldsymbol{\alpha} \geq 0, \;\; \boldsymbol{\alpha}^T \mathbf{y} = 0,$$

is convex in K.

So, a fundamental property of the inverse margin is that it is convex in K. This is essential, since it allows us to optimize this quantity in a convex framework. The following theorem shows that, for a suitable choice of the set $\mathcal{K}$, this convex optimization problem can be cast as an SDP.

Theorem 19 *Given a linearly separable labeled sample* $S_{n_{tr}} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_{n_{tr}}, y_{n_{tr}})\}$ *with the set of labels denoted* $\mathbf{y} \in \mathbb{R}^{n_{tr}}$*, the kernel matrix* $K \in \mathcal{K}$ *that optimizes (28) can be found by solving the following problem:*

$$\begin{array}{rl} \displaystyle\min_{K,t,\lambda,\boldsymbol{\nu}} & t \\ \textit{subject to} & \operatorname{trace}(K) = c, \\ & K \in \mathcal{K}, \\ & \begin{pmatrix} G(K_{tr}) & \mathbf{e} + \boldsymbol{\nu} + \lambda \mathbf{y} \\ (\mathbf{e} + \boldsymbol{\nu} + \lambda \mathbf{y})^T & t \end{pmatrix} \succeq 0, \\ & \boldsymbol{\nu} \geq 0. \end{array} \tag{29}$$

Proof Observe $\omega(K_{tr}) = \omega_{\infty,0}(K_{tr})$. Apply Theorem 16 for $C = \infty$ and $\tau = 0$. ■

If $\mathcal{K} = \{K \succeq 0\}$, there is no constraint to ensure that a large margin on the training data will give a large margin on the test data: a test matrix $K_t = O$ would optimize the criterion.

If we restrict the kernel matrix to a linear subspace $\mathcal{K} = \operatorname{span}\{K_1, \ldots, K_m\} \cap \{K \succeq 0\}$, we obtain

$$\begin{array}{rl} \displaystyle\min_{K} & \omega(K_{tr}) \\ \text{subject to} & \operatorname{trace}(K) = c, \\ & K \succeq 0, \\ & K = \displaystyle\sum_{i=1}^{m} \mu_i K_i, \end{array} \tag{30}$$

which can be written in the standard form of a semidefinite program, in a manner analogous to (29):

$$\begin{aligned} \min_{\mu_i, t, \lambda, \boldsymbol{\nu}} \quad & t \\ \text{subject to} \quad & \operatorname{trace}\left(\sum_{i=1}^{m} \mu_i K_i\right) = c, \\ & \sum_{i=1}^{m} \mu_i K_i \succeq 0, \\ & \begin{pmatrix} G(\sum_{i=1}^{m} \mu_i K_{i,tr}) & \mathbf{e} + \boldsymbol{\nu} + \lambda \mathbf{y} \\ (\mathbf{e} + \boldsymbol{\nu} + \lambda \mathbf{y})^T & t \end{pmatrix} \succeq 0, \\ & \boldsymbol{\nu} \geq 0. \end{aligned} \tag{31}$$

Notice that the SDP approach is consistent with the bound in (19). The margin is optimized over the labeled data (via the use of $K_{i,tr}$), while the positive semidefiniteness and the trace constraint are imposed for the entire kernel matrix K (via the use of K_i). This leads to a general method for learning the kernel matrix with semidefinite programming, when using a margin criterion for hard margin SVMs. Applying the complexity results mentioned in Section 3.3 leads to a worst-case complexity $O\left((m + n_{tr})^2 n^{2.5}\right)$ when using general-purpose interior-point methods to solve this particular SDP.

Furthermore, this gives a new transduction method for hard margin SVMs. Whereas Vapnik's original method for transduction scales exponentially in the number of test samples, the new SDP method has polynomial time complexity.

Remark. For the specific case in which the K_i are rank-one matrices $K_i = \mathbf{v}_i \mathbf{v}_i^T$, with $\mathbf{v}_i$ orthonormal (e.g., the normalized eigenvectors of an initial kernel matrix K_0), the semidefinite program reduces to a QCQP:

$$\begin{aligned} \max_{\boldsymbol{\alpha}, t} \quad & 2\boldsymbol{\alpha}^T \mathbf{e} - ct \\ \text{subject to} \quad & t \geq (\check{\mathbf{v}}_i^T \boldsymbol{\alpha})^2, i = 1, \ldots, m \\ & \boldsymbol{\alpha}^T \mathbf{y} = 0, \\ & \boldsymbol{\alpha} \geq 0, \end{aligned} \tag{32}$$

with $\check{\mathbf{v}}_i = \operatorname{diag}(\mathbf{y})\ \mathbf{v}_i(1 : n_{tr})$.

This can be seen by observing that, for $K_i = \mathbf{v}_i \mathbf{v}_i^T$, with $\mathbf{v}_i^T \mathbf{v}_j = \delta_{ij}$, we have that $\sum_{i=1}^m \mu_i K_i \succeq 0$ is equivalent to $\boldsymbol{\mu} \geq 0$. So, we can apply Theorem 17, with $\tau = 0$ and $C = \infty$, where $\frac{1}{r_i}\ \boldsymbol{\alpha}^T\ G(K_{i,tr})\ \boldsymbol{\alpha} = \boldsymbol{\alpha}^T \operatorname{diag}(\mathbf{y})\ \mathbf{v}_i(1 : n_{tr})\ \mathbf{v}_i(1 : n_{tr})^T\ \operatorname{diag}(\mathbf{y})\ \boldsymbol{\alpha} = (\check{\mathbf{v}}_i^T \boldsymbol{\alpha})^2$.

4.3 Hard Margin with Kernel Matrices that are Positive Linear Combinations

To learn a kernel matrix from this linear class $\mathcal{K}$, one has to solve a semidefinite programming problem: interior-point methods (Nesterov and Nemirovsky, 1994) are polynomial time, but have a worst-case complexity $O\left((m + n_{tr})^2 n^{2.5}\right)$ in this particular case.

We now restrict $\mathcal{K}$ to the positive linear combinations of kernel matrices:

$$K = \sum_{i=1}^{m} \mu_i K_i, \qquad \boldsymbol{\mu} \geq 0.$$

Assuming positive weights yields a smaller set of kernel matrices, because the weights need not be positive for K to be positive semidefinite, even if the components K_i are positive semidefinite. Moreover, the restriction has beneficial computational effects: (1) the general SDP reduces to a QCQP, which can be solved with significantly lower complexity $O(mn_{tr}^3)$; (2) the constraint can result in improved numerical stability—it prevents the algorithm from using large weights with opposite sign that cancel. Finally, we shall see in Section 5 that the constraint also yields better estimates of the generalization performance of these algorithms.

Theorem 20 *Given a labeled sample $S_{n_{tr}} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_{n_{tr}}, y_{n_{tr}})\}$ with the set of labels denoted $\mathbf{y} \in \mathbb{R}^{n_{tr}}$, the kernel matrix $K = \sum_{i=1}^m \mu_i K_i$ that optimizes (21), with $\tau \geq 0$, under the additional constraint $\boldsymbol{\mu} \geq 0$ can be found by solving the following convex optimization problem, and considering its dual solution:*

$$\begin{aligned} \max_{\boldsymbol{\alpha}, t} \quad & 2\boldsymbol{\alpha}^T \mathbf{e} - ct \\ \textit{subject to} \quad & t \geq \frac{1}{r_i} \boldsymbol{\alpha}^T G(K_{i,tr}) \boldsymbol{\alpha}, \quad i = 1, \ldots, m \\ & \boldsymbol{\alpha}^T \mathbf{y} = 0, \\ & \boldsymbol{\alpha} \geq 0. \end{aligned} \tag{33}$$

where $\mathbf{r} \in \mathbb{R}^m$ with trace$(K_i) = r_i$.

Proof Apply Theorem 17 for $C = \infty$ and $\tau = 0$. ■

Note once again that the optimal weights $\mu_i, \quad i = 1, \ldots, m$, can be recovered from the primal-dual solution found by standard software such as SeDuMi (Sturm, 1999) or Mosek (Andersen and Andersen, 2000).

4.4 1-Norm Soft Margin

For the case of non-linearly separable data, we can consider the 1-norm soft margin cost function in (3). Training the SVM for a given kernel involves minimizing this quantity with respect to $\mathbf{w}, b$, and $\boldsymbol{\xi}$, which yields the optimal value (4): obviously this minimum is a function of the particular choice of K, which is expressed explicitly in (4) as a dual problem. Let us now optimize this quantity with respect to the kernel matrix K, i.e., let us try to find the kernel matrix $K \in \mathcal{K}$ for which the corresponding embedding yields minimal $\omega_{S1}(K_{tr})$, keeping the trace of K constant:

$$\min_{K \in \mathcal{K}} \omega_{S1}(K_{tr}) \qquad \text{s.t. trace}(K) = c. \tag{34}$$

This is again a convex optimization problem.

Theorem 21 *Given a labeled sample $S_{n_{tr}} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_{n_{tr}}, y_{n_{tr}})\}$ with the set of labels denoted $\mathbf{y} \in \mathbb{R}^{n_{tr}}$, the kernel matrix $K \in \mathcal{K}$ that optimizes (34), can be found by solving the following convex* *optimization problem:*

$$\begin{aligned}
\min_{K,t,\lambda,\boldsymbol{\nu},\boldsymbol{\delta}} \quad & t \\
\textit{subject to} \quad & \textit{trace}(K) = c, \\
& K \in \mathcal{K}, \\
& \begin{pmatrix} G(K_{tr}) & \mathbf{e}+\boldsymbol{\nu}-\boldsymbol{\delta}+\lambda\mathbf{y} \\ (\mathbf{e}+\boldsymbol{\nu}-\boldsymbol{\delta}+\lambda\mathbf{y})^T & t-2C\boldsymbol{\delta}^T\mathbf{e} \end{pmatrix} \succeq 0, \\
& \boldsymbol{\nu} \geq 0, \\
& \boldsymbol{\delta} \geq 0.
\end{aligned} \tag{35}$$

Proof Observe $\omega_{S1}(K_{tr}) = \omega_{C,0}(K_{tr})$. Apply Theorem 16 for $\tau = 0$. ■

Again, if $\mathcal{K} = \{K \succeq 0\}$, this is an SDP. Adding the additional constraint (18) that K is a linear combination of fixed kernel matrices leads to the following SDP:

$$\begin{aligned}
\min_{\mu_i,t,\lambda,\boldsymbol{\nu},\boldsymbol{\delta}} \quad & t \\
\text{subject to} \quad & \text{trace}\left(\sum_{i=1}^m \mu_i K_i\right) = c, \\
& \sum_{i=1}^m \mu_i K_i \succeq 0, \\
& \begin{pmatrix} G(\sum_{i=1}^m \mu_i K_{i,tr}) & \mathbf{e}+\boldsymbol{\nu}-\boldsymbol{\delta}+\lambda\mathbf{y} \\ (\mathbf{e}+\boldsymbol{\nu}-\boldsymbol{\delta}+\lambda\mathbf{y})^T & t-2C\boldsymbol{\delta}^T\mathbf{e} \end{pmatrix} \succeq 0, \\
& \boldsymbol{\nu}, \boldsymbol{\delta} \geq 0.
\end{aligned} \tag{36}$$

Remark. For the specific case in which the K_i are rank-one matrices $K_i = \mathbf{v}_i\mathbf{v}_i^T$, with $\mathbf{v}_i$ orthonormal (e.g., the normalized eigenvectors of an initial kernel matrix K_0), the SDP reduces to a QCQP using Theorem 17, with $\tau = 0$, in a manner analogous to the hard margin case:

$$\begin{aligned}
\max_{\boldsymbol{\alpha},t} \quad & 2\boldsymbol{\alpha}^T\mathbf{e} - ct \\
\text{subject to} \quad & t \geq (\check{\mathbf{v}}_i^T\boldsymbol{\alpha})^2, i = 1,\ldots,m \\
& \boldsymbol{\alpha}^T\mathbf{y} = 0, \\
& C \geq \boldsymbol{\alpha} \geq 0,
\end{aligned} \tag{37}$$

with $\check{\mathbf{v}}_i = \text{diag}(\mathbf{y})\ \mathbf{v}_i(1:n_{tr})$.

Solving the original learning problem subject to the extra constraint $\boldsymbol{\mu} \geq 0$ yields, after applying Theorem 17, with $\tau = 0$:

$$\begin{aligned}
\max_{\boldsymbol{\alpha},t} \quad & 2\boldsymbol{\alpha}^T\mathbf{e} - ct \\
\text{subject to} \quad & t \geq \frac{1}{r_i}\boldsymbol{\alpha}^T G(K_{i,tr})\boldsymbol{\alpha}, \quad i = 1,\ldots,m \\
& \boldsymbol{\alpha}^T\mathbf{y} = 0, \\
& C \geq \boldsymbol{\alpha} \geq 0.
\end{aligned} \tag{38}$$

4.5 2-Norm Soft Margin

For the case of non-linearly separable data, we can also consider the 2-norm soft margin cost function (5). Again, training for a given kernel will minimize this quantity with respect to $\mathbf{w}, b$, and $\boldsymbol{\xi}$ and the minimum is a function of the particular choice of K, as expressed in (6) in dual form. Let us now optimize this quantity with respect to the kernel matrix K:

$$\min_{K \in \mathcal{K}} \omega_{S2}(K_{tr}) \qquad \text{s.t. trace}(K) = c. \tag{39}$$

This is again a convex optimization problem, and can be restated as follows.

Theorem 22 *Given a labeled sample $S_{n_{tr}} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_{n_{tr}}, y_{n_{tr}})\}$ with the set of labels denoted $\mathbf{y} \in \mathbb{R}^{n_{tr}}$, the kernel matrix $K \in \mathcal{K}$ that optimizes (39) can be found by solving the following optimization problem:*

$$\begin{aligned}
\min_{K,t,\lambda,\boldsymbol{\nu}} \quad & t \qquad\qquad (40)\\
\textit{subject to} \quad & \mathit{trace}(K) = c,\\
& K \in \mathcal{K},\\
& \begin{pmatrix} G(K_{tr}) + \frac{1}{C} I_{n_{tr}} & \mathbf{e} + \boldsymbol{\nu} + \lambda \mathbf{y} \\ (\mathbf{e} + \boldsymbol{\nu} + \lambda \mathbf{y})^T & t \end{pmatrix} \succeq 0,\\
& \boldsymbol{\nu} \geq 0.
\end{aligned}$$

Proof Observe $\omega_{S2}(K_{tr}) = \omega_{\infty,\tau}(K_{tr})$. Apply Theorem 16 for $C = \infty$. ■

Again, if $\mathcal{K} = \{K \succeq 0\}$, this is an SDP. Moreover, constraining K to be a linear combination of fixed kernel matrices, we obtain

$$\begin{aligned}
\min_{\mu_i,t,\lambda,\boldsymbol{\nu}} \quad & t \qquad\qquad (41)\\
\text{subject to} \quad & \text{trace}\left(\sum_{i=1}^m \mu_i K_i\right) = c,\\
& \sum_{i=1}^m \mu_i K_i \succeq 0,\\
& \begin{pmatrix} G(\sum_{i=1}^m \mu_i K_{i,tr}) + \frac{1}{C} I_{n_{tr}} & \mathbf{e} + \boldsymbol{\nu} + \lambda \mathbf{y} \\ (\mathbf{e} + \boldsymbol{\nu} + \lambda \mathbf{y})^T & t \end{pmatrix} \succeq 0,\\
& \boldsymbol{\nu} \geq 0.
\end{aligned}$$

Also, when the K_i are rank-one matrices, $K_i = \mathbf{v}_i \mathbf{v}_i^T$, with $\mathbf{v}_i$ orthonormal, we obtain a QCQP:

$$\begin{aligned}
\max_{\boldsymbol{\alpha},t} \quad & 2\boldsymbol{\alpha}^T \mathbf{e} - \frac{1}{C} \boldsymbol{\alpha}^T \boldsymbol{\alpha} - ct \qquad\qquad (42)\\
\text{subject to} \quad & t \geq (\check{\mathbf{v}}_i^T \boldsymbol{\alpha})^2, i = 1, \ldots, m\\
& \boldsymbol{\alpha}^T \mathbf{y} = 0,\\
& \boldsymbol{\alpha} \geq 0,
\end{aligned}$$

and, finally, imposing the constraint $\boldsymbol{\mu} \geq 0$ yields

$$\begin{array}{rl} \max\limits_{\boldsymbol{\alpha},t} & 2\boldsymbol{\alpha}^T\mathbf{e} - \frac{1}{C}\boldsymbol{\alpha}^T\boldsymbol{\alpha} - ct \\ \text{subject to} & t \geq \frac{1}{r_i}\boldsymbol{\alpha}^T G(K_{i,tr})\boldsymbol{\alpha}, \quad i = 1,\ldots,m \\ & \boldsymbol{\alpha}^T\mathbf{y} = 0, \\ & \boldsymbol{\alpha} \geq 0, \end{array} \tag{43}$$

following a similar derivation as before: apply Theorem 17 with $C = \infty$, and, for (42), observe that $\boldsymbol{\mu} \geq 0$ is equivalent to $\sum_{i=1}^m \mu_i K_i \succeq 0$ if $K_i = \mathbf{v}_i\mathbf{v}_i^T$ and $\mathbf{v}_i^T\mathbf{v}_j = \delta_{ij}$.

4.6 Learning the 2-Norm Soft Margin Parameter $\tau = 1/C$

This section shows how the 2-norm soft margin parameter of SVMs can be learned using SDP or QCQP. More details can be found in De Bie et al. (2003).

In the previous section, we tried to find the kernel matrix $K \in \mathcal{K}$ for which the corresponding embedding yields minimal $\omega_{S2}(K_{tr})$, keeping the trace of K constant. Since in the dual formulation (6) the identity matrix induced by the 2-norm formulation appears in exactly the same way as the other matrices K_i, we can treat it on the same basis and optimize its weight to obtain the optimal dual formulation, i.e., to minimize $\omega_{S2}(K_{tr})$. Since this weight now happens to correspond to the parameter $\tau = 1/C$, optimizing it corresponds to learning the 2-norm soft margin parameter and thus has a significant meaning.

Since the parameter $\tau = 1/C$ can be treated in the same way as the weights μ_i, tuning it such that the quantity $\omega_{S2}(K_{tr}, \tau)$ is minimized can be viewed as a method for choosing τ. First of all, consider the dual formulation (6) and notice that $\omega_{S2}(K_{tr}, \tau)$ is convex in $\tau = 1/C$ (being the pointwise maximum of affine and thus convex functions in τ). Secondly, since $\tau \to \infty$ leads to $\omega_{S2}(K_{tr}, \tau) \to 0$, we impose the constraint $\operatorname{trace}(K + \tau I_n) = c$. This results in the following convex optimization problem:

$$\min_{K \in \mathcal{K}, \tau \geq 0} \omega_{S2}(K_{tr}, \tau) \qquad \text{s.t. } \operatorname{trace}(K + \tau I_n) = c.$$

According to Theorem 22, this can be restated as follows:

$$\begin{array}{rl} \min\limits_{K,t,\lambda,\boldsymbol{\nu},\tau} & t \\ \text{subject to} & \operatorname{trace}(K + \tau I_n) = c, \\ & K \in \mathcal{K}, \\ & \begin{pmatrix} G(K_{tr}) + \tau I_{n_{tr}} & \mathbf{e} + \boldsymbol{\nu} + \lambda\mathbf{y} \\ (\mathbf{e} + \boldsymbol{\nu} + \lambda\mathbf{y})^T & t \end{pmatrix} \succeq 0, \\ & \boldsymbol{\nu}, \tau \geq 0. \end{array} \tag{44}$$

Again, if $\mathcal{K} = \{K \succeq 0\}$, this is an SDP. Imposing the additional constraint that K is a linear function of fixed kernel matrices, we obtain the SDP

$$\begin{aligned} \min_{\mu_i, t, \lambda, \boldsymbol{\nu}, \tau} \quad & t \\ \text{subject to} \quad & \operatorname{trace}\left(\sum_{i=1}^m \mu_i K_i + \tau I_n\right) = c, \\ & \sum_{i=1}^m \mu_i K_i \succeq 0, \\ & \begin{pmatrix} G(\sum_{i=1}^m \mu_i K_{i,tr}) + \tau I_{n_{tr}} & \mathbf{e} + \boldsymbol{\nu} + \lambda \mathbf{y} \\ (\mathbf{e} + \boldsymbol{\nu} + \lambda \mathbf{y})^T & t \end{pmatrix} \succeq 0, \\ & \boldsymbol{\nu}, \tau \geq 0, \end{aligned} \tag{45}$$

and imposing the additional constraint that the K_i are rank-one matrices, we obtain a QCQP:

$$\begin{aligned} \max_{\boldsymbol{\alpha}, t} \quad & 2\boldsymbol{\alpha}^T \mathbf{e} - ct \\ \text{subject to} \quad & t \geq (\check{\mathbf{v}}_i^T \boldsymbol{\alpha})^2, i = 1, \ldots, m \\ & t \geq \frac{1}{n} \boldsymbol{\alpha}^T \boldsymbol{\alpha} \\ & \boldsymbol{\alpha}^T \mathbf{y} = 0, \\ & \boldsymbol{\alpha} \geq 0, \end{aligned} \tag{46}$$

with $\check{\mathbf{v}}_i = \operatorname{diag}(\mathbf{y})\ \bar{\mathbf{v}}_i = \operatorname{diag}(\mathbf{y})\ \mathbf{v}_i(1 : n_{tr})$. Finally, imposing the constraint that $\boldsymbol{\mu} \geq 0$ yields the following:

$$\begin{aligned} \max_{\boldsymbol{\alpha}, t} \quad & 2\boldsymbol{\alpha}^T \mathbf{e} - ct && (47) \\ \text{subject to} \quad & t \geq \frac{1}{r_i} \boldsymbol{\alpha}^T G(K_{i,tr}) \boldsymbol{\alpha}, \quad i = 1, \ldots, m \\ & t \geq \frac{1}{n} \boldsymbol{\alpha}^T \boldsymbol{\alpha} && (48) \\ & \boldsymbol{\alpha}^T \mathbf{y} = 0, \\ & \boldsymbol{\alpha} \geq 0, \end{aligned}$$

which, as before, is a QCQP.

Solving (47) corresponds to learning the kernel matrix as a positive linear combination of kernel matrices according to a 2-norm soft margin criterion and simultaneously learning the 2-norm soft margin parameter $\tau = 1/C$. Comparing (47) with (33), we can see that this reduces to learning an augmented kernel matrix K' as a positive linear combination of kernel matrices and the identity matrix, $K' = K + \tau I_n = \sum_{i=1}^m \mu_i K_i + \tau I_n$, using a hard margin criterion. However, there is an important difference: when evaluating the resulting classifier, the actual kernel matrix K is used, instead of the augmented K' (see, for example, Shawe-Taylor and Cristianini, 1999).

For $m = 1$, we notice that (45) directly reduces to (47) if $K_1 \succeq 0$. This corresponds to automatically tuning the parameter $\tau = 1/C$ for a 2-norm soft margin SVM with kernel matrix K_1. So, even when not learning the kernel matrix, this approach can be used to tune the 2-norm soft margin parameter $\tau = 1/C$ automatically.

4.7 Alignment

In this section, we consider the problem of optimizing the alignment between a set of labels and a kernel matrix from some class $\mathcal{K}$ of positive semidefinite kernel matrices. We show that, if $\mathcal{K}$ is a class of linear combinations of fixed kernel matrices, this problem can be cast as an SDP. This result generalizes the approach presented in Cristianini et al. (2001, 2002).

Theorem 23 *The kernel matrix $K \in \mathcal{K}$ which is maximally aligned with the set of labels $\mathbf{y} \in \mathbb{R}^{n_{tr}}$ can be found by solving the following optimization problem:*

$$\begin{array}{cl} \max_{A,K} & \langle K_{tr}, \mathbf{y}\mathbf{y}^T \rangle_F \\ \textit{subject to} & \textit{trace}(A) \leq 1 \\ & \begin{pmatrix} A & K^T \\ K & I_n \end{pmatrix} \succeq 0 \\ & K \in \mathcal{K}, \end{array} \tag{49}$$

where I_n is the identity matrix of dimension n.

Proof We want to find the kernel matrix K which is maximally aligned with the set of labels $\mathbf{y}$:

$$\begin{array}{cl} \max_{K} & \hat{A}(S, K_{tr}, \mathbf{y}\mathbf{y}^T) \\ \text{subject to} & K \in \mathcal{K},\ \text{trace}(K) = 1. \end{array}$$

This is equivalent to the following optimization problem:

$$\begin{array}{cl} \max_{K} & \langle K_{tr}, \mathbf{y}\mathbf{y}^T \rangle_F \\ \text{subject to} & \langle K, K \rangle_F = 1 \\ & K \in \mathcal{K},\ \text{trace}(K) = 1. \end{array} \tag{50}$$

To express this in the standard form (9) of a semidefinite program, we need to express the quadratic equality constraint $\langle K, K \rangle_F = 1$ as an LMI. First, notice that (50) is equivalent to

$$\begin{array}{cl} \max_{K} & \langle K_{tr}, \mathbf{y}\mathbf{y}^T \rangle_F \\ \text{subject to} & \langle K, K \rangle_F \leq 1 \\ & K \in \mathcal{K}. \end{array} \tag{51}$$

Indeed, we are maximizing an objective which is linear in the entries of K, so at the optimum $K = K^*$, the constraint $\langle K, K \rangle_F = \text{trace}(K^T K) \leq 1$ is achieved: $\langle K^*, K^* \rangle_F = 1$. The quadratic inequality constraint in (51) is now equivalent to

$$\exists A \ :\ K^T K \preceq A \quad \text{and} \quad \text{trace}(A) \leq 1.$$

Indeed, $A - K^T K \succeq 0$ implies $\text{trace}(A - K^T K) = \text{trace}(A) - \text{trace}(K^T K) \geq 0$ because of linearity of the trace. Using the Schur complement lemma, we can express $A - K^T K \succeq 0$ as an LMI:

$$A - K^T K \succeq 0 \Leftrightarrow \begin{pmatrix} A & K^T \\ K & I_n \end{pmatrix} \succeq 0.$$

We can thus rewrite the optimization problem (50) as

$$\begin{array}{ll} \max_{A,K} & \langle K_{tr}, \mathbf{y}\mathbf{y}^T \rangle_F \\ \text{subject to} & \operatorname{trace}(A) \leq 1 \\ & \begin{pmatrix} A & K^T \\ K & I_n \end{pmatrix} \succeq 0 \\ & K \in \mathcal{K}, \end{array}$$

which corresponds to (49). ■

Notice that, when $\mathcal{K}$ is the set of all positive semidefinite matrices, this is an SDP (an inequality constraint corresponds to a one-dimensional LMI; consider the entries of the matrices A and K as the unknowns, corresponding to the u_i in (9)). In that case, one solution of (49) is found by simply selecting $K_{tr} = \frac{c}{n}\mathbf{y}\mathbf{y}^T$, for which the alignment (7) is equal to one and thus maximized.

Adding the additional constraint (18) that K is a linear combination of fixed kernel matrices leads to

$$\begin{array}{ll} \max_{K} & \left\langle K_{tr}, \mathbf{y}\mathbf{y}^T \right\rangle_F \\ \text{subject to} & \langle K, K \rangle_F \leq 1, \\ & K \succeq 0, \\ & K = \sum_{i=1}^{m} \mu_i K_i, \end{array} \tag{52}$$

which can be written in the standard form of a semidefinite program, in a similar way as for (49):

$$\begin{array}{ll} \max_{A,\mu_i} & \left\langle \sum_{i=1}^{m} \mu_i K_{i,tr}, \mathbf{y}\mathbf{y}^T \right\rangle_F \\ \text{subject to} & \operatorname{trace}(A) \leq 1, \\ & \begin{pmatrix} A & \sum_{i=1}^m \mu_i K_i^T \\ \sum_{i=1}^m \mu_i K_i & I_n \end{pmatrix} \succeq 0, \\ & \sum_{i=1}^{m} \mu_i K_i \succeq 0. \end{array} \tag{53}$$

Remark. For the specific case where the K_i are rank-one matrices $K_i = \mathbf{v}_i \mathbf{v}_i^T$, with $\mathbf{v}_i$ orthonormal (e.g., the normalized eigenvectors of an initial kernel matrix K_0), the semidefinite program reduces to a QCQP (see Appendix A):

$$\begin{array}{ll} \max_{\mu_i} & \sum_{i=1}^{m} \mu_i (\bar{\mathbf{v}}_i^T \mathbf{y})^2 \\ \text{subject to} & \sum_{i=1}^{m} \mu_i^2 \leq 1 \\ & \mu_i \geq 0, i = 1, \dots, m \end{array} \tag{54}$$

with $\bar{\mathbf{v}}_i = \mathbf{v}_i(1 : n_{tr})$. This corresponds exactly to the QCQP obtained as an illustration in Cristianini et al. (2002), which is thus entirely captured by the general SDP result obtained in this section.

Solving the original learning problem (52) subject to the extra constraint $\boldsymbol{\mu} \geq 0$ yields

$$\begin{array}{ll} \max\limits_{K} & \left\langle K_{tr}, \mathbf{y}\mathbf{y}^T \right\rangle_F \\ \text{subject to} & \langle K, K \rangle_F \leq 1, \\ & K \succeq 0, \\ & K = \sum\limits_{i=1}^{m} \mu_i K_i, \\ & \boldsymbol{\mu} \geq 0. \end{array}$$

We can omit the second constraint, because this is implied by the last two constraints, if $K_i \succeq 0$. This reduces to

$$\begin{array}{ll} \max\limits_{\boldsymbol{\mu}} & \left\langle \sum\limits_{i=1}^{m} \mu_i K_{i,tr}, \mathbf{y}\mathbf{y}^T \right\rangle_F \\ \text{subject to} & \left\langle \sum\limits_{i=1}^{m} \mu_i K_i, \sum\limits_{j=1}^{m} \mu_j K_j \right\rangle_F \leq 1, \\ & \boldsymbol{\mu} \geq 0, \end{array}$$

where $K_{i,tr} = K_i(1 : n_{tr}, 1 : n_{tr})$. Expanding this further yields

$$\begin{aligned} \left\langle \sum_{i=1}^{m} \mu_i K_{i,tr}, \mathbf{y}\mathbf{y}^T \right\rangle_F &= \sum_{i=1}^{m} \mu_i \left\langle K_{i,tr}, \mathbf{y}\mathbf{y}^T \right\rangle_F \\ &= \boldsymbol{\mu}^T \mathbf{q}, \end{aligned} \tag{55}$$

$$\begin{aligned} \left\langle \sum_{i=1}^{m} \mu_i K_i, \sum_{j=1}^{m} \mu_j K_j \right\rangle_F &= \sum_{i,j=1}^{m} \mu_i \mu_j \langle K_i, K_j \rangle_F \\ &= \boldsymbol{\mu}^T S \boldsymbol{\mu} \end{aligned} \tag{56}$$

with $q_i = \left\langle K_{i,tr}, \mathbf{y}\mathbf{y}^T \right\rangle_F = \mathrm{trace}(K_{i,tr}\mathbf{y}\mathbf{y}^T) = \mathrm{trace}(\mathbf{y}^T K_{i,tr}\mathbf{y}) = \mathbf{y}^T K_{i,tr}\mathbf{y}$ and $S_{ij} = \langle K_i, K_j \rangle_F$, where $\mathbf{q} \in \mathbb{R}^m, S \in \mathbb{R}^{m \times m}$. We used the fact that $\mathrm{trace}(ABC) = \mathrm{trace}(BCA)$ (if the products are well-defined). We obtain the following learning problem:

$$\begin{array}{ll} \max\limits_{\boldsymbol{\mu}} & \boldsymbol{\mu}^T \mathbf{q} \\ \text{subject to} & \boldsymbol{\mu}^T S \boldsymbol{\mu} \leq 1, \\ & \boldsymbol{\mu} \geq 0, \end{array}$$

which is a QCQP.

4.8 Induction

In previous sections we have considered the transduction setting, where it is assumed that the covariate vectors for both training (labeled) and test (unlabeled) data are known beforehand. While this setting captures many realistic learning problems, it is also of interest to consider possible extensions of our approach to the more general setting of induction, in which the covariates are known beforehand only for the training data.

Consider the following situation. We learn the kernel matrix as a positive linear combination of normalized kernel matrices K_i. Those K_i are obtained through the evaluation of a kernel function or through a known procedure (e.g., a string-matching kernel), yielding $K_i \succeq 0$. So, $K = \sum_{i=1}^m \mu_i K_i \succeq 0$. Normalization is done by replacing $K_i(k,l)$ by $K_i(k,l)/\sqrt{K_i(k,k) \cdot K_i(l,l)}$. In this case, the extension to an induction setting is elegant and simple.

Let n_{tr} be the number of training data points (all labeled). Consider the transduction problem for those n_{tr} data points and one unknown test point, e.g., for a hard margin SVM. The optimal weights $\mu_i^*, \quad i = 1, \ldots, m$ are learned by solving (33):

$$\begin{aligned} \max_{\boldsymbol{\alpha},t} \quad & 2\boldsymbol{\alpha}^T\mathbf{e} - ct \\ \text{subject to} \quad & t \geq \frac{1}{n_{tr}+1}\boldsymbol{\alpha}^T G(K_{i,tr})\boldsymbol{\alpha}, \quad i = 1, \ldots, m \\ & \boldsymbol{\alpha}^T\mathbf{y} = 0, \\ & \boldsymbol{\alpha} \geq 0. \end{aligned} \tag{57}$$

Even without knowing the test point and the entries of the K_i's related to it (column and row $n_{tr}+1$), we know that $K(n_{tr}+1, n_{tr}+1) = 1$ because of the normalization. So, $\text{trace}(K_i) = n_{tr}+1$. This allows solving for the optimal weights $\mu_i^*, \quad i = 1, \ldots, m$ and the optimal SVM parameters $\alpha_j^*, \quad j = 1, \ldots, n_{tr}$ and b^*, without knowing the test point. When a test point becomes available, we complete the K_i's by computing their $(n_{tr}+1)$-th column and row (evaluate the kernel function or follow the procedure and normalize). Combining those K_i with weights μ_i^* yields the final kernel matrix K, which can then be used to label the test point:

$$y = \text{sign}(\sum_{i=1}^m \sum_{j=1}^{n_{tr}} \mu_i^* \alpha_j K_i(x_j, x)).$$

Remark: The optimal weights are independent of the number of unknown test points that are considered in this setting. Consider the transduction problem (57) for l unknown test points instead of one unknown test point:

$$\begin{aligned} \max_{\tilde{\boldsymbol{\alpha}},\tilde{t}} \quad & 2\tilde{\boldsymbol{\alpha}}^T\mathbf{e} - c\tilde{t} \\ \text{subject to} \quad & \tilde{t} \geq \frac{1}{n_{tr}+l}\tilde{\boldsymbol{\alpha}}^T G(K_{i,tr})\tilde{\boldsymbol{\alpha}}, \quad i = 1, \ldots, m \\ & \tilde{\boldsymbol{\alpha}}^T\mathbf{y} = 0, \\ & \tilde{\boldsymbol{\alpha}} \geq 0. \end{aligned} \tag{58}$$

One can see that solving (58) is equivalent to solving (57) where the optimal values relate as $\tilde{\boldsymbol{\alpha}}^* = \frac{n_{tr}+l}{n_{tr}+1}\boldsymbol{\alpha}^*$ and $\tilde{t}^* = \frac{n_{tr}+l}{n_{tr}+1}t^*$ and where the optimal weights $\mu_i^*, \quad i = 1, \ldots, m$ are the same.

Tackling the induction problem in full generality remains a challenge for future work. Obviously, one could consider the transduction case with zero test points, yielding the induction case. If the weights μ_i are constrained to be nonnegative and furthermore the matrices K_i are guaranteed to be positive semidefinite, the weights can be reused at new test points. To deal with induction in a general SDP setting, one could solve a transduction problem for each new test point. For every test point, this leads to solving an SDP of dimension $n_{tr}+1$, which is computationally expensive. Clearly there is a need to explore recursive solutions to the SDP problem that allow the solution of the SDP of dimension n_{tr} to be used in the solution of an SDP of dimension $n_{tr}+1$. Such solutions would of course also have immediate applications to on-line learning problems.

5. Error Bounds for Transduction

In the problem of transduction, we have access to the unlabeled test data, as well as the labeled training data, and the aim is to optimize accuracy in predicting the test data. We assume that the data are fixed, and that the order is chosen randomly, yielding a random partition into a labeled training set and an unlabeled test set. For convenience, we suppose here that the training and test sets have the same size. Of course, if we can show a performance guarantee that holds with high probability over uniformly chosen training/test partitions of this kind, it also holds with high probability over an i.i.d. choice of the training and test data, since permuting an i.i.d. sample leaves the distribution unchanged.

The following theorem gives an upper bound on the error of a kernel classifier on the test data in terms of the average over the training data of a certain margin cost function, together with properties of the kernel matrix. We focus on the 1-norm soft margin classifier, although our results extend in a straightforward way to other cases, including the 2-norm soft margin classifier. The 1-norm soft margin classifier chooses a kernel classifier f to minimize a weighted combination of a regularization term, $\|\mathbf{w}\|^2$, and the average over the training sample of the slack variables,

$$\xi_i = \max\left(1 - y_i f(x_i), 0\right).$$

We can view this regularized empirical criterion as the Lagrangian for the constrained minimization of

$$\frac{1}{n}\sum_{i=1}^{n}\xi_i = \frac{1}{n}\sum_{i=1}^{n}\max(1 - y_i f(x_i), 0)$$

subject to the upper bound $\|\mathbf{w}\|^2 \leq 1/\gamma^2$.

Fix a sequence of $2n$ pairs $(\mathbf{X}_1, Y_1), \ldots, (\mathbf{X}_{2n}, Y_{2n})$ from $\mathcal{X}\times\mathcal{Y}$. Let $\pi : \{1, \ldots, 2n\} \to \{1, \ldots, 2n\}$ be a random permutation, chosen uniformly, and let $(\mathbf{x}_i, y_i) = (\mathbf{X}_{\pi(i)}, Y_{\pi(i)})$. The first half of this randomly ordered sequence is the training data $T_n = ((\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n))$, and the second half is the test data $S_n = ((\mathbf{x}_{n+1}, y_{n+1}), \ldots, (\mathbf{x}_{2n}, y_{2n}))$. For a function $f : \mathcal{X} \to \mathbb{R}$, the proportion of errors on the test data of a thresholded version of f can be written as

$$\mathrm{er}(f) = \frac{1}{n}|\{n+1 \leq i \leq 2n : y_i f(\mathbf{x}_i) \leq 0\}|.$$

We consider kernel classifiers obtained by thresholding kernel expansions of the form

$$f(\mathbf{x}) = \langle \mathbf{w}, \Phi(\mathbf{x})\rangle = \sum_{i=1}^{2n}\alpha_i k(\mathbf{x}_i, \mathbf{x}), \tag{59}$$

where $\mathbf{w} = \sum_{i=1}^{2n}\alpha_i\Phi(\mathbf{x}_i)$ is chosen with bounded norm,

$$\|\mathbf{w}\|^2 = \sum_{i,j=1}^{2n}\alpha_i\alpha_j k(\mathbf{x}_i, \mathbf{x}_j) = \boldsymbol{\alpha}^T K \boldsymbol{\alpha} \leq \frac{1}{\gamma^2}, \tag{60}$$

and K is the $2n \times 2n$ kernel matrix with $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$.

Let $F_{\mathcal{K}}$ denote the class of functions on S of the form (59) satisfying (60), for some $K \in \mathcal{K}$,

$$F_{\mathcal{K}} = \left\{ \mathbf{x}_j \mapsto \sum_{i=1}^{2n} \alpha_i K_{ij} : K \in \mathcal{K},\, \boldsymbol{\alpha}^T K \boldsymbol{\alpha} \leq \frac{1}{\gamma^2} \right\},$$

where $\mathcal{K}$ is a set of positive semidefinite $2n \times 2n$ matrices. We also consider the class of kernel expansions obtained from certain linear combinations of a fixed set $\{K_1, \ldots, K_m\}$ of kernel matrices: Define the class $F_{\mathcal{K}_c}$ as

$$\mathcal{K}_c = \left\{ K = \sum_{j=1}^{m} \mu_j K_j : K \succeq 0,\, \mu_j \in \mathbb{R},\, \mathrm{trace}(K) \leq c \right\},$$

and the class $F_{\mathcal{K}_c^+}$ as

$$\mathcal{K}_c^+ = \left\{ \sum_{j=1}^{m} \mu_j K_j : K \succeq 0,\, \mu_j \geq 0,\, \mathrm{trace}(K) \leq c \right\}.$$

Theorem 24 *For every $\gamma > 0$, with probability at least $1-\delta$ over the data $(\mathbf{x}_i, y_i)$ chosen as above, every function $f \in F_{\mathcal{K}}$ has $\mathrm{er}(f)$ no more than*

$$\frac{1}{n}\sum_{i=1}^{n} \max\{1 - y_i f(\mathbf{x}_i), 0\} + \frac{1}{\sqrt{n}}\left(4 + \sqrt{2\log(1/\delta)} + \sqrt{\frac{\mathcal{C}(\mathcal{K})}{n\gamma^2}}\right),$$

where

$$\mathcal{C}(\mathcal{K}) = \mathbf{E} \max_{K \in \mathcal{K}} \boldsymbol{\sigma}^T K \boldsymbol{\sigma},$$

with the expectation over $\boldsymbol{\sigma}$ chosen uniformly from $\{\pm 1\}^{2n}$.

Furthermore,

$$\mathcal{C}(\mathcal{K}_c) = c\mathbf{E} \max_{K \in \mathcal{K}} \boldsymbol{\sigma}^T \frac{K}{\mathit{trace}(K)} \boldsymbol{\sigma},$$

and this is always no more than cn, and

$$\mathcal{C}(\mathcal{K}_c^+) \leq c \min\left(m, n \max_j \frac{\lambda_j}{\mathit{trace}(K_j)}\right),$$

where λ_j is the largest eigenvalue of K_j.

Notice that the test error is bounded by a sum of the average over the training data of a margin cost function plus a complexity penalty term that depends on the ratio between the trace of the kernel matrix and the squared margin parameter, γ^2. The kernel matrix here is the full matrix, combining both test and training data.

The proof of the theorem is in Appendix B. The proof technique for the first part of the theorem was introduced by Koltchinskii and Panchenko (2002), who used it to give error bounds for boosting algorithms.

Although the theorem requires the margin parameter γ to be specified in advance, it is straightforward to extend the result to give an error bound that holds with high probability over all values of γ. In this case, the $\log(1/\delta)$ in the bound would be replaced by $\log(1/\delta) + |\log(1/\gamma)|$ and the constants would increase slightly. See, for example, Proposition 8 and its applications in the work of Bartlett (1998).

The result is presented for the 1-norm soft margin classifier, but the proof uses only two properties of the cost function $a \mapsto \max\{1-a, 0\}$: that it is an upper bound on the indicator function for $a \leq 0$, and that it satisfies a Lipschitz constraint on $[0, \infty)$. These conditions are also satisfied by the cost function associated with the 2-norm soft margin classifier, $a \mapsto (\max\{1-a, 0\})^2$, for example.

The bound on the complexity $\mathcal{C}(\mathcal{K}_B^+)$ of the kernel class $\mathcal{K}_B^+$ is easier to check than the bound on $\mathcal{C}(\mathcal{K}_B)$. The first term in the minimum shows that the set of positive linear combinations of a small set of kernel matrices is not very complex. The second term shows that if, for each matrix in the set, the largest eigenvalue does not dominate the sum of the eigenvalues (the trace), then the set of positive linear combinations is not too complex, even if the set is large. In either case, the upper bound is linear in c, the upper bound on the trace of the combined kernel matrix.

6. Empirical Results

We first present results on benchmark data sets, using kernels K_i that are derived from the same input vector. The goal here is to explore different possible representations of the same data source, and to choose a representation or combinations of representations that yield the best performance. We compare to the soft margin SVM with an RBF kernel, in which the hyperparameter is tuned via cross-validation. Note that in our framework there is no need for cross-validation to tune the corresponding kernel hyperparameters. Moreover, when using the 2-norm soft margin SVM, the methods are directly comparable, because the hyperparameter C is present in both cases.

In the second section we explore the use of our framework to combine kernels that are built using data from heterogeneous sources. Here our main interest is in comparing the combined classifier to the best *individual* classifier. To the extent that the heterogeneous data sources provide complementary information, we might expect that the performance of the combined classifier can dominate that of the best individual classifier.

6.1 Benchmark Data Sets

We present results for hard margin and soft margin support vector machines. We use a kernel matrix $K = \sum_{i=1}^{3} \mu_i K_i$, where the K_i's are initial "guesses" of the kernel matrix. We use a polynomial kernel function $k_1(\mathbf{x}_1, \mathbf{x}_2) = (1+\mathbf{x}_1^T\mathbf{x}_2)^d$ for K_1, a Gaussian kernel function $k_2(\mathbf{x}_1, \mathbf{x}_2) = \exp(-0.5(\mathbf{x}_1 - \mathbf{x}_2)^T(\mathbf{x}_1 - \mathbf{x}_2)/\sigma)$ for K_2 and a linear kernel function $k_3(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{x}_1^T\mathbf{x}_2$ for K_3. Afterwards, all K_i are normalized. After evaluating the initial kernel matrices $\{K_i\}_{i=1}^3$, the weights $\{\mu_i\}_{i=1}^3$ are optimized in a transduction setting according to a hard margin, a 1-norm soft margin and a 2-norm soft margin criterion, respectively; the semidefinite programs (31), (36) and (41) are solved using the general-purpose optimization software SeDuMi (Sturm, 1999), leading to optimal weights $\{\mu_i^*\}_{i=1}^3$. Next, the weights $\{\mu_i\}_{i=1}^3$ are constrained to be non-negative and optimized according to the same criteria, again in a transduction setting: the second order cone programs (33), (38) and (43) are solved using the general-purpose optimization software Mosek (Andersen and Andersen, 2000), leading to optimal weights $\{\mu_{i,+}^*\}_{i=1}^3$. For positive weights, we also report results in which the 2-norm soft margin hyperparameter C is tuned according to (47).

Empirical results on standard benchmark datasets are summarized in Tables 1, 2 and 3.[2] The Wisconsin breast cancer dataset contained 16 incomplete examples which were not used. The breast

2. It is worth noting that the first three columns of these columns are based on an inductive algorithm whereas the last two columns are based on a transductive algorithm. This may favor the kernel combinations in the last

cancer, ionosphere and sonar data were obtained from the UCI repository. The heart data were obtained from STATLOG and normalized. Data for the 2-norm problem data were generated as specified by Breiman (1998). Each dataset was randomly partitioned into 80% training and 20% test sets. The reported results are the averages over 30 random partitions. The kernel parameters for K_1 and K_2 are given in Tables 1, 2 and 3 by d and σ respectively. For each of the kernel matrices, an SVM is trained using the training block K_{tr} and tested using the mixed block $K_{tr,t}$ as defined in (17). The margin γ (for a hard margin criterion) and the optimal soft margin cost functions ω^*_{S1} and ω^*_{S2} (for soft margin criteria) are reported for the initial kernel matrices K_i, as well as for the optimal $\sum_i \mu^*_i K_i$ and $\sum_i \mu^*_{i,+} K_i$. Furthermore, the average test set accuracy (TSA), the average value for C and the average weights over the 30 partitions are listed. For comparison, the performance of the best soft margin SVM with an RBF (Gaussian) kernel is reported—the soft margin hyperparameter C and the kernel parameter σ for the Gaussian kernel were tuned using cross-validation over 30 random partitions of the training set.

Note that not every K_i gives rise to a linearly separable embedding of the training data, in which case no hard margin classifier can be found (indicated with a dash). The matrices $\sum_i \mu^*_i K_i$ and $\sum_i \mu^*_{i,+} K_i$ however, always allow the training of a hard margin SVM and its margin is indeed larger than the margin for each of the different components K_i—this is consistent with the SDP/QCQP optimization. For the soft margin criteria, the optimal value of the cost function for $\sum_i \mu^*_i K_i$ and $\sum_i \mu^*_{i,+} K_i$ is smaller than its value for the individual K_i—again consistent with the SDP/QCQP optimizations. Notice that constraining the weights μ_i to be positive results in slightly smaller margins and larger cost functions, as expected.

Furthermore, the number of test set errors for $\sum_i \mu^*_i K_i$ and $\sum_i \mu^*_{i,+} K_i$ is in general comparable in magnitude to the best value achieved among the different components K_i. Also notice that $\sum_i \mu^*_{i,+} K_i$ does often almost as well as $\sum_i \mu^*_i K_i$, and sometimes even better: we can thus achieve a substantial reduction in computational complexity without a significant loss of performance. Moreover, the performance of $\sum_i \mu^*_i K_i$ and $\sum_i \mu^*_{i,+} K_i$ is comparable with the best soft margin SVM with an RBF kernel. In making this comparison note that the RBF SVM requires tuning of the kernel parameter using cross-validation, while the kernel learning approach achieves a similar effect without cross-validation.[3] Moreover, when using the 2-norm soft margin SVM with tuned hyperparameter C, we no longer need to do cross-validation for C. This leads to a smaller value of the optimal cost function ω^*_{S2} (compared to the case SM2, with $C = 1$) and performs well on the test set, while offering the advantage of automatically adjusting C.

One might wonder why there is a difference between the SDP and the QCQP approach for the 2-norm data, since both seem to find positive weights μ_i. However, it must be recalled that

two columns and thus the results should be interpreted with caution. However, it is also worth noting that the transduction is a weak form of transduction that is based only on the norm of the test data point.

3. The experiments were run on a 2GHz Windows XP machine. We used the programs SeDuMi to solve the SDP for kernel learning and Mosek to solve multiple QP's for cross-validated SVM and the QCQP for kernel learning with positive weights. The run time for the SDP is on the order of minutes (approximately 10 minutes for 300 data points and 5 kernels), while the run time for the QP and QCQP is on the order of seconds (approximately 1 second for 300 data points and 1 kernel, and approximately 3 seconds for 300 data points and 5 kernels). Thus, we see that kernel learning with positive weights, which requires only a QCQP solution, achieves an accuracy which is comparable to the full SDP approach at a fraction of the computational cost, and our tentative recommendation is that the QCQP approach is to be preferred. It is worth noting, however, that special-purpose implementations of SDPs that take advantage of the structure of the kernel learning problem may well yield significant speed-ups, and the recommendation should be taken with caution. Finally, the QCQP approach also compares favorably in terms of run time to the multiple runs of a QP that are required for cross-validation, and should be considered a viable alternative to cross-validation, particularly given the high variance associated with cross-validation in small data sets.

		K_1	K_2	K_3	$\sum_i \mu_i^* K_i$	$\sum_i \mu_{i,+}^* K_i$	best c/v RBF
Heart		$d=2$	$\sigma=0.5$				
HM	γ	0.0369	0.1221	-	0.1531	0.1528	
	TSA	**72.9 %**	**59.5 %**	-	**84.8 %**	**84.6 %**	**77.7 %**
	$\mu_1/\mu_2/\mu_3$	3/0/0	0/3/0	0/0/3	**-0.09/2.68/0.41**	**0.01/2.60/0.39**	
SM1	ω_{S1}^*	58.169	33.536	74.302	21.361	21.446	
	TSA	**79.3 %**	**59.5 %**	**84.3 %**	**84.8 %**	**84.6 %**	**83.9 %**
	C	1	1	1	1	1	
	$\mu_1/\mu_2/\mu_3$	3/0/0	0/3/0	0/0/3	**-0.09/2.68/0.41**	**0.01/2.60/0.39**	
SM2	ω_{S2}^*	32.726	25.386	45.891	15.988	16.034	
	TSA	**78.1 %**	**59.0 %**	**84.3 %**	**84.8 %**	**84.6 %**	**83.2 %**
	C	1	1	1	1	1	
	$\mu_1/\mu_2/\mu_3$	3/0/0	0/3/0	0/0/3	**-0.08/2.54/0.54**	**0.01/2.47/0.53**	
SM2,C	ω_{S2}^*	19.643	25.153	16.004		15.985	
	TSA	**81.3 %**	**59.6 %**	**84.7 %**		**84.6 %**	**83.2 %**
	C	0.3378	1.18e+7	0.2880		0.4365	
	$\mu_1/\mu_2/\mu_3$	1.04/0/0	0/3.99/0	0/0/0.53		**0.01/0.80/0.53**	
Sonar		$d=2$	$\sigma=0.1$				
HM	γ	0.0246	0.1460	0.0021	0.1517	0.1459	
	TSA	**80.9 %**	**85.8 %**	**74.2 %**	**84.6 %**	**85.8 %**	**84.2 %**
	$\mu_1/\mu_2/\mu_3$	3/0/0	0/3/0	0/0/3	**-2.23/3.52/1.71**	**0/3/0**	
SM1	ω_{S1}^*	87.657	23.288	102.68	21.637	23.289	
	TSA	**78.1 %**	**85.6 %**	**73.3 %**	**84.6 %**	**85.6 %**	**84.2 %**
	C	1	1	1	1	1	
	$\mu_1/\mu_2/\mu_3$	3/0/0	0/3/0	0/0/3	**-2.20/3.52/1.69**	**0/3/0**	
SM2	ω_{S2}^*	45.048	15.893	53.292	15.219	15.893	
	TSA	**79.1 %**	**85.2 %**	**76.7 %**	**84.5 %**	**85.2 %**	**84.2 %**
	C	1	1	1	1	1	
	$\mu_1/\mu_2/\mu_3$	3/0/0	0/3/0	0/0/3	**-1.78/3.46/1.32**	**0/3/0**	
SM2,C	ω_{S2}^*	20.520	15.640	20.620		15.640	
	TSA	**60.9 %**	**84.6 %**	**51.0 %**		**84.6 %**	**84.2 %**
	C	0.2591	0.6087	0.2510		0.6087	
	$\mu_1/\mu_2/\mu_3$	0.14/0/0	0/2.36/0	0/0/0.02		**0/2.34/0**	

Table 1: SVMs trained and tested with the initial kernel matrices K_1, K_2, K_3 and with the optimal kernel matrices $\sum_i \mu_i^* K_i$ and $\sum_i \mu_{i,+}^* K_i$. For hard margin SVMs (HM), the resulting margin γ is given—a dash meaning that no hard margin classifier could be found; for soft margin SVMs (SM1 = 1-norm soft margin with $C = 1$, SM2 = 2-norm soft margin with $C = 1$ and SM2,C = 2-norm soft margin with auto tuning of C) the optimal value of the cost function ω_{S1}^* or ω_{S2}^* is given. Furthermore, the test-set accuracy (TSA), the average weights and the average C-values are given. For c we used $c = \sum_i \mathrm{trace}(K_i)$ for HM, SM1 and SM2. The initial kernel matrices are evaluated after being multiplied by 3. This assures we can compare the different γ for HM, ω_{S1}^* for SM1 and ω_{S2}^* for SM2, since the resulting kernel matrix has a constant trace (thus, everything is on the same scale). For SM2,C we use $c = \sum_i \mathrm{trace}(K_i) + \mathrm{trace}(I_n)$. This not only allows comparing the different ω_{S2}^* for SM2,C but also it allows comparing ω_{S2}^* between SM2 and SM2,C (since we choose $C = 1$ for SM2, we have that trace $\left(\sum_{i=1}^m \mu_i K_i + \frac{1}{C} I_n\right)$ is constant in both cases, so again, we are on the same scale). Finally, the column 'best c/v RBF' reports the performance of the best soft margin SVM with RBF kernel, tuned using cross-validation.

the values in Table 3 are averages over 30 randomizations—for some randomizations the SDP has actually found negative weights, although the averages are positive.

As a further example illustrating the flexibility of the SDP framework, consider the following setup. Let $\{K_i\}_{i=1}^5$ be Gaussian kernels with $\sigma = 0.01, 0.1, 1, 10, 100$ respectively. Combining those optimally with $\mu_i \geq 0$ for a 2-norm soft margin SVM, with tuning of C, yields the results

		K_1	K_2	K_3	$\sum_i \mu_i^* K_i$	$\sum_i \mu_{i,+}^* K_i$	best c/v RBF
Breast cancer		$d = 2$	$\sigma = 0.5$				
HM	γ	0.0036	0.1055	-	0.1369	0.1219	
	TSA	**92.9 %**	**89.0 %**	-	**95.5 %**	**94.4 %**	**96.1 %**
	$\mu_1/\mu_2/\mu_3$	3/0/0	0/3/0	0/0/3	**1.90/2.35/-1.25**	**0.65/2.35/0**	
SM1	ω^*_{S1}	77.012	44.913	170.26	26.694	33.689	
	TSA	**96.4 %**	**89.0 %**	**87.7 %**	**95.5 %**	**94.4 %**	**96.7 %**
	C	1	1	1	1	1	
	$\mu_1/\mu_2/\mu_3$	3/0/0	0/3/0	0/0/3	**1.90/2.35/-1.25**	**0.65/2.35/0**	
SM2	ω^*_{S2}	43.138	35.245	102.51	20.696	21.811	
	TSA	**96.4 %**	**88.5 %**	**87.4 %**	**95.4 %**	**94.3 %**	**96.8 %**
	C	1	1	1	1	1	
	$\mu_1/\mu_2/\mu_3$	3/0/0	0/3/0	0/0/3	**2.32/2.13/-1.46**	**0.89/2.11/0**	
SM2,C	ω^*_{S2}	27.682	33.685	41.023		25.267	
	TSA	**94.5 %**	**89.0 %**	**87.3 %**		**94.4 %**	**96.8 %**
	C	0.3504	1.48e+8	0.3051		6.77e+7	
	$\mu_1/\mu_2/\mu_3$	1.15/0/0	0/3.99/0	0/0/0.72		**0.87/3.13/0**	
Ionosphere		$d = 2$	$\sigma = 0.5$				
HM	γ	0.0613	0.1452	-	0.1623	0.1616	
	TSA	**91.2 %**	**92.0 %**	-	**94.4 %**	**94.4 %**	**93.9 %**
	$\mu_1/\mu_2/\mu_3$	3/0/0	0/3/0	0/0/3	**1.08/2.18/-0.26**	**0.79/2.21/0**	
SM1	ω^*_{S1}	30.786	23.233	52.312	18.117	18.303	
	TSA	**94.5 %**	**92.1 %**	**83.1 %**	**94.8 %**	**94.5 %**	**94.0 %**
	C	1	1	1	1	1	
	$\mu_1/\mu_2/\mu_3$	3/0/0	0/3/0	0/0/3	**1.23/2.07/-0.30**	**0.90/2.10/0**	
SM2	ω^*_{S2}	18.533	17.907	31.662	13.382	13.542	
	TSA	**94.7 %**	**92.0 %**	**91.6 %**	**94.5 %**	**94.4 %**	**94.2 %**
	C	1	1	1	1	1	
	$\mu_1/\mu_2/\mu_3$	3/0/0	0/3/0	0/0/3	**1.68/1.73/-0.41**	**1.23/1.78/0**	
SM2,C	ω^*_{S2}	14.558	17.623	18.975		13.5015	
	TSA	**93.5 %**	**92.1 %**	**90.0 %**		**94.6 %**	**94.2 %**
	C	0.4144	5.8285	0.3442		0.8839	
	$\mu_1/\mu_2/\mu_3$	1.59/0/0	0/3.83/0	0/0/1.09		**1.24/1.61/0**	

Table 2: See the caption to Table 1 for explanation.

		K_1	K_2	K_3	$\sum_i \mu_i^* K_i$	$\sum_i \mu_{i,+}^* K_i$	best c/v RBF
2-norm		$d = 2$	$\sigma = 0.1$				
HM	γ	0.1436	0.1072	0.0509	0.2170	0.2169	
	TSA	**94.6 %**	**55.4 %**	**94.3 %**	**96.6 %**	**96.6 %**	**96.3 %**
	$\mu_1/\mu_2/\mu_3$	3/0/0	0/3/0	0/0/3	**0.03/1.91/1.06**	**0.06/1.88/1.06**	
SM1	ω^*_{S1}	23.835	43.509	22.262	10.636	10.641	
	TSA	**95.0 %**	**55.4 %**	**95.7 %**	**96.6 %**	**96.6 %**	**97.5 %**
	C	1	1	1	1	1	
	$\mu_1/\mu_2/\mu_3$	3/0/0	0/3/0	0/0/3	**0.03/1.91/1.06**	**0.06/1.88/1.06**	
SM2	ω^*_{S2}	16.134	32.631	11.991	7.9780	7.9808	
	TSA	**95.9 %**	**55.4 %**	**95.6 %**	**96.6 %**	**96.6 %**	**97.2 %**
	C	1	1	1	1	1	
	$\mu_1/\mu_2/\mu_3$	3/0/0	0/3/0	0/0/3	**0.05/1.54/1.41**	**0.08/1.51/1.41**	
SM2,C	ω^*_{S2}	16.057	32.633	7.9880		7.9808	
	TSA	**96.2 %**	**55.4 %**	**96.6 %**		**96.6 %**	**97.2 %**
	C	0.8213	0.5000	0.3869		0.8015	
	$\mu_1/\mu_2/\mu_3$	2.78/0/0	0/2/0	0/0/1.42		**0.08/1.25/1.41**	

Table 3: See the caption to Table 1 for explanation.

in Table 4—averages over 30 randomizations in 80% training and 20% test sets. The test set accuracies obtained for $\sum_i \mu_{i,+}^* K_i$ are competitive with those for the best soft margin SVM with an RBF kernel, tuned using cross-validation. The average weights show that some kernels are selected and others are not. Effectively we obtain a data-based choice of smoothing parameter without recourse to cross-validation.

	$\mu_{1,+}$	$\mu_{2,+}$	$\mu_{3,+}$	$\mu_{4,+}$	$\mu_{5,+}$	C	TSA SM2,C	TSA best c/v RBF
Breast Cancer	0	0	3.24	0.94	0.82	3.6e+08	97.1 %	96.8 %
Ionosphere	0.85	0.85	2.63	0.68	0	4.0e+06	94.5 %	94.2 %
Heart	0	3.89	0.06	1.05	0	2.5e+05	84.1 %	83.2 %
Sonar	0	3.93	1.07	0	0	3.2e+07	84.8 %	84.2 %
2-norm	0.49	0.49	0	3.51	0	2.0386	96.5 %	97.2 %

Table 4: The initial kernel matrices $\{K_i\}_{i=1}^5$ are Gaussian kernels with $\sigma = 0.01, 0.1, 1, 10, 100$ respectively. For c we used $c = \sum_i \mathrm{trace}(K_i) + \mathrm{trace}(I_n)$. $\{\mu_{i,+}\}_{i=1}^5$ are the average weights of the optimal kernel matrix $\sum_i \mu_{i,+}^* K_i$ for a 2-norm soft margin SVM with $\mu_i \geq 0$ and tuning of C. The average C-value is given as well. The test set accuracies (TSA) of the optimal 2-norm soft margin SVM with tuning of C (SM2,C) and the best crossvalidation soft margin SVM with RBF kernel (best c/v RBF) are reported.

In Cristianini et al. (2002) empirical results are given for optimization of the alignment using a kernel matrix $K = \sum_{i=1}^N \mu_i \mathbf{v}_i \mathbf{v}_i^T$. The results show that optimizing the alignment indeed improves the generalization power of Parzen window classifiers. As explained in Section 4.7, it turns out that in this particular case, the SDP in (53) reduces to exactly the quadratic program that is obtained in Cristianini et al. (2002) and thus those results also provide support for the general framework presented in the current paper.

6.2 Combining Heterogeneous Data

6.2.1 Reuters-21578 Data Set

To explore the value of this approach for combining data from heterogeneous sources, we run experiments on the Reuters-21578 data set, using two different kernels. The first kernel K_1 is derived as a linear kernel from the "bag-of-words" representation of the different documents, capturing information about the frequency of terms in the different documents (Salton and McGill, 1983). K_1 is centered and normalized. The second kernel K_2 is constructed by extracting 500 concepts from documents via probabilistic latent semantic analysis (Cai and Hofmann, 2003). This kernel can be viewed as arising from a document-concept-term graphical model, with the concepts as hidden nodes. After inferring the conditional probabilities of the concepts, given a document, a linear kernel is applied to the vector of these probabilistic "concept memberships," representing each document. Also K_2 is then centered and normalized. The concept-based document information contained in K_2 is likely to be partly overlapping and partly complementary to the term-frequency information in K_1. Although the "bag-of-words" and graphical model representation are clearly heterogeneous, they can both be cast into a homogeneous framework of kernel matrices, allowing the information that they convey to be combined according to $K = \mu_1 K_1 + \mu_2 K_2$.

The Reuters-21578 dataset consists of Reuters newswire stories from 1987 (`www.davidlewis.com/resources/testcollections/reuters21578/`). After a preprocessing stage that includes tokenization and stop word removal, 37926 word types remained. We used the modified Apte ("ModApte") split to split the collection into 12902 used and 8676 unused documents. The 12902 used documents consist of 9603 training documents and 3299 test documents. From the 9603 training documents, we randomly select a 1000-document subset as training set for a soft margin support vector machine with $C = 1$. We train the SVM for the binary classification tasks of distinguishing documents about a certain topic versus those not about that topic. We restrict our attention to the topics that appear in the most documents (cf. Cai and Hofmann (2003); Huang (2003); Eyheramendy et al. (2003)); in particular, we focused on the top five Reuters-21578 topics.

After training the SVM on the randomly selected documents using either K_1 or K_2, the accuracy is tested on the 3299 test documents from the ModApte split. This is done 20 times, i.e., for 20 randomly chosen 1000-document training sets. The average accuracies and standard errors are reported in Figure 1. After evaluating the performance of K_1 and K_2, the weights μ_1 and μ_2 are constrained to be non-zero and optimized (using only the training data) according to (38). The test set performance of the optimal combination is then evaluated and the average accuracy reported in Figure 1. The optimal weights, μ_1^* and μ_2^*, do not vary greatly over the different topics, with averages of 1.37 for μ_1^* and 0.63 for μ_2^*.

We see that in four cases out of five the optimal combination of kernels performs better than either of the individual kernels. This suggests that these kernels indeed provide complementary information for the classification decision, and that the SDP approach is able to find a combination that exploits this complementarity.

6.2.2 Protein Function Prediction

Here we illustrate the SDP approach for fusing heterogeneous genomic data in order to predict protein function in yeast; see Lanckriet et al. (2004) for more details. The task is to predict functional classifications associated with yeast proteins. We use as a gold standard the functional catalogue provided by the MIPS Comprehensive Yeast Genome Database (CYGD—`mips.gsf.de/proj/yeast`). The top-level categories in the functional hierarchy produce 13 classes, which contain 3588 proteins; the remaining yeast proteins have uncertain function and are therefore not used in evaluating the classifier. Because a given protein can belong to several functional classes, we cast the prediction problem as 13 binary classification tasks, one for each functional class. Using this setup, we follow the experimental paradigm of Deng et al. (2003).

The primary input to the classification algorithm is a collection of kernel matrices representing different types of data:

1. Amino acid sequences: this kernel incorporates information about the domain structure of each protein, by looking at the presence or absence in the protein of Pfam domains (`pfam.wustl.edu`). The corresponding kernel is simply the inner product between binary vectors describing the presence or absence of one Pfam domain. Afterwards, we also construct a richer kernel by replacing the binary scoring with log E-values using the HMMER software toolkit (`hmmer.wustl.edu`). Moreover, an additional kernel matrix is constructed by applying the Smith-Waterman (SW) pairwise sequence comparison algorithm (Smith and Waterman, 1981) to the yeast protein sequences and applying the empirical kernel map (Tsuda, 1999).

2. Protein-protein interactions: this type of data can be represented as a graph, with proteins as nodes and interactions as edges. Such interaction graph allows to establish similarities among proteins through the construction of a corresponding diffusion kernel (Kondor and Lafferty, 2002).

3. Genetic interactions: in a similar way, these interactions give rise to a diffusion kernel.

4. Protein complex data: co-participation in a protein complex can be seen as a weak sort of interaction, giving rise to a third diffusion kernel.

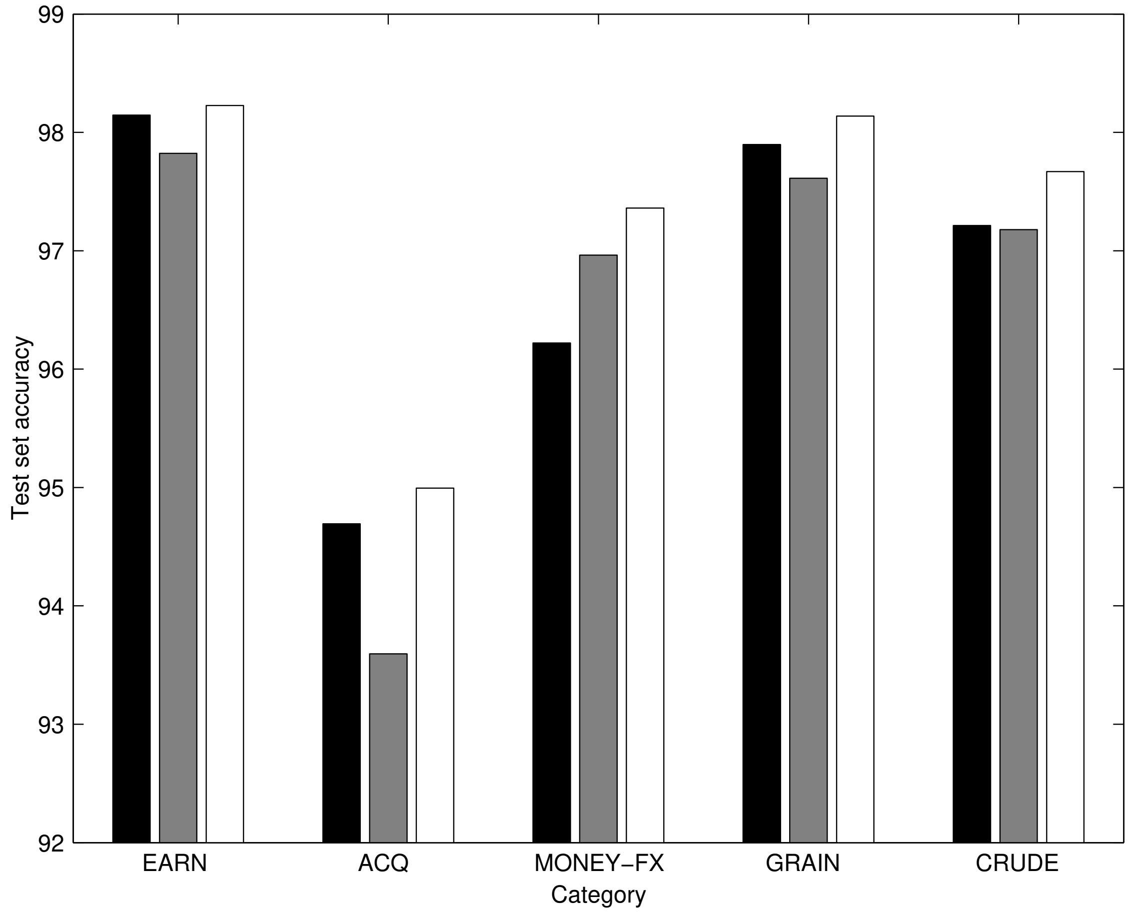

Figure 1: **Classification performance for the top five Reuters-21578 topics.** The height of each bar is proportional to the average test set accuracy for a 1-norm soft margin SVM with $C = 1$. Black bars correspond to using only kernel matrix K_1; grey bars correspond to using only kernel matrix K_2, and white bars correspond to the optimal combination $\mu_1^* K_1 + \mu_2^* K_2$. The kernel matrices K_1 and K_2 are derived from different types of data, i.e., from the "bag-of-words" representation of documents and the concept-based graphical model representation (with 500 concepts) of documents respectively. For c we used $c = \text{trace}(K_1) + \text{trace}(K_2) = 4000$. The standard errors across the 20 experiments are approximately 0.1 or smaller; indeed, all of the depicted differences between the optimal combination and the individual kernels are statistically significant except for EARN.

5. Expression data: two genes with similar expression profiles are likely to have similar functions; accordingly, Deng et al. (2003) convert the expression matrix to a square binary interaction matrix in which a 1 indicates that the corresponding pair of expression profiles exhibits a Pearson correlation greater than 0.8. This can be used to define a diffusion kernel. Also, a richer Gaussian kernel is defined directly on the expression profiles.

In order to compare the SDP/SVM approach to the Markov random field (MRF) method of Deng et al. (2003), Lanckriet et al. (2004) perform two variants of the experiment: one in which the five kernels are restricted to contain precisely the same binary information as used by the MRF method, and a second experiment in which the richer Pfam and expression kernels are used and the SW kernel is added. They show that a combined SVM classifier trained with the SDP approach performs better than an SVM trained on any single type of data. Moreover it outperforms the MRF method designed for this data set. To illustrate the latter, Figure 2 presents the average ROC scores on the test set when performing five-fold cross-validation three times.

The figure shows that, for each of the 13 classifications, the ROC score of the SDP/SVM method is better than that of the MRF method. Overall, the mean ROC improves from 0.715 to 0.854. The improvement of the SDP/SVM method over the MRF method is consistent and statistically significant across all 13 classes. An additional improvement, though not as large and only statistically significant for nine of the 13 classes, is gained by using richer kernels and adding the SW kernel.

7. Discussion

In this paper we have presented a new method for learning a kernel matrix from data. Our approach makes use of semidefinite programming (SDP) ideas. It is motivated by the fact that every symmetric, positive semidefinite matrix can be viewed as a kernel matrix (corresponding to a certain embedding of a finite set of data), and the fact that SDP deals with the optimization of convex cost functions over the convex cone of positive semidefinite matrices (or convex subsets of this cone). Thus convex optimization and machine learning concerns merge to provide a powerful methodology for learning the kernel matrix with SDP.

We have focused on the transductive setting, where the labeled data are used to learn an embedding, which is then applied to the unlabeled part of the data. Based on a new generalization bound for transduction, we have shown how to impose convex constraints that effectively control the capacity of the search space of possible kernels and yield an efficient learning procedure that can be implemented by SDP. Furthermore, this approach leads to a convex method to learn the 2-norm soft margin parameter in support vector machines, solving an important open problem. Promising empirical results are reported on standard benchmark datasets; these results show that the new approach provides a principled way to combine multiple kernels to yield a classifier that is comparable with the best individual classifier, and can perform better than any individual kernel. Performance is also comparable with a classifier in which the kernel hyperparameter is tuned with cross-validation; our approach achieves the effect of this tuning without cross-validation.

We have also shown how optimizing a linear combination of kernel matrices provides a novel method for fusing heterogeneous data sources. In this case, the empirical results show a significant improvement of the classification performance for the optimal combination of kernels when compared to individual kernels.

There are several challenges that need to be met in future research on SDP-based learning algorithms. First, it is clearly of interest to explore other convex quality measures for a kernel matrix, which may be appropriate for other learning algorithms. For example, in the setting of Gaussian

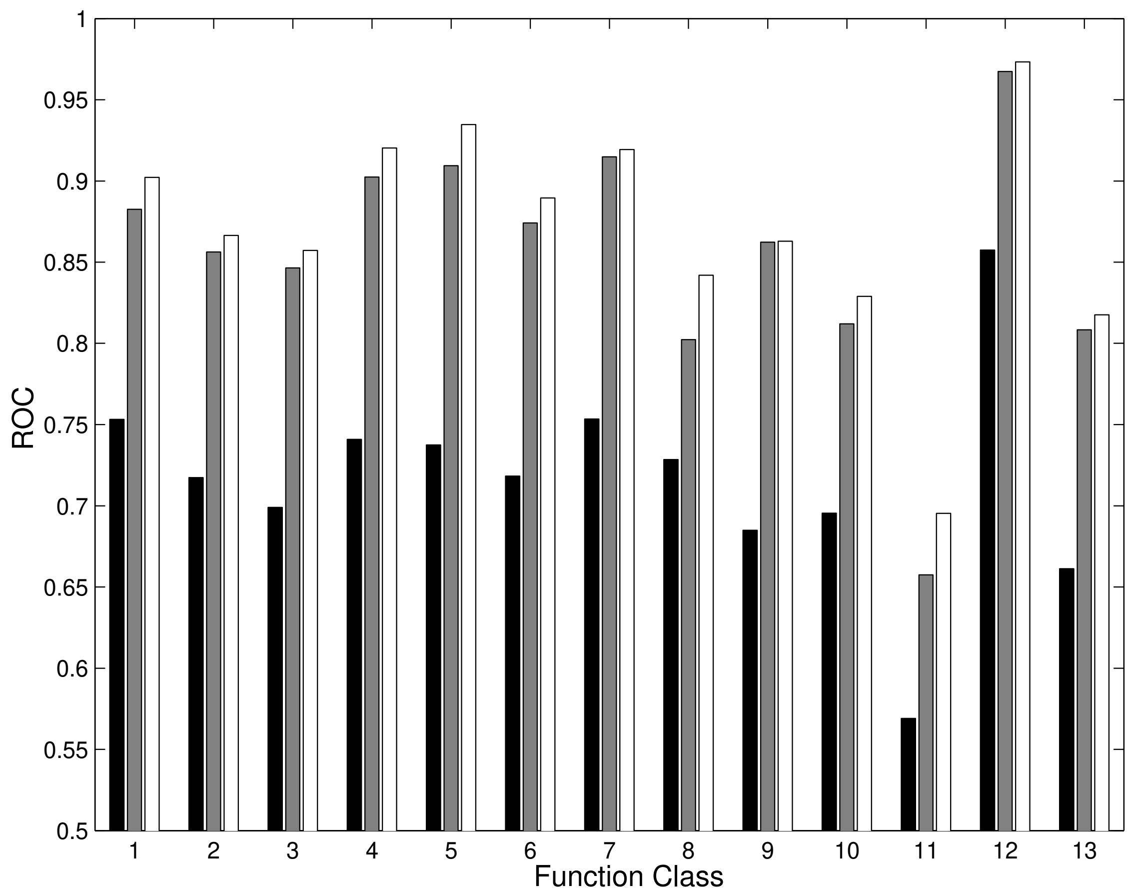

Figure 2: **Classification performance for the 13 functional protein classes.** The height of each bar is proportional to the ROC score. The standard error across the 13 experiments is usually 0.01 or smaller, so most of the depicted differences are statistically significant: between black and grey bars, all depicted differences are statistically significant, while nine of the 13 differences between grey and white bars are statistically significant. Black bars correspond to the MRF method of Deng *et al.*; grey bars correspond to the SDP/SVM method using five kernels computed on binary data, and white bars correspond to the SDP/SVM using the enriched Pfam kernel and replacing the expression kernel with the SW kernel. See Lanckriet et al. (2004) for more details.

processes, the relative entropy between the zero-mean Gaussian process prior P with covariance kernel K and the corresponding Gaussian process approximation Q to the true intractable posterior process depends on K as

$$D[P||Q] = \frac{1}{2}\log\det K + \frac{1}{2}\mathrm{trace}\left(\mathbf{y}^T K\mathbf{y}\right) + d,$$

where the constant d is independent of K. One can verify that $D[P||Q]$ is convex with respect to $R = K^{-1}$ (see, e.g., Vandenberghe et al., 1998). Minimizing this measure with respect to R, and thus K, is motivated from PAC-Bayesian generalization error bounds for Gaussian processes (see, e.g., Seeger, 2002) and can be achieved by solving a so-called *maximum-determinant problem* (Vandenberghe et al., 1998)—an even more general framework that contains semidefinite programming as a special case.

Second, the investigation of other parameterizations of the kernel matrix is an important topic for further study. While the linear combination of kernels that we have studied here is likely to be useful in many practical problems—capturing a notion of combining Gram matrix "experts"—it is also worth considering other parameterizations as well. Any such parameterizations have to respect the constraint that the quality measure for the kernel matrix is convex with respect to the parameters of the proposed parameterization. One class of examples arises via the positive definite matrix completion problem (Vandenberghe et al., 1998). Here we are given a symmetric kernel matrix K that has some entries which are fixed. The remaining entries—the parameters in this case—are to be chosen such that the resulting matrix is positive definite, while simultaneously a certain cost function is optimized, e.g., $\mathrm{trace}(SK) + \log\det K^{-1}$, where S is a given matrix. This specific case reduces to solving a maximum-determinant problem which is convex in the unknown entries of K, the parameters of the proposed parameterization.

A third important area for future research consists in finding faster implementations of semidefinite programming. As in the case of quadratic programming (Platt, 1999), it seems likely that special purpose methods can be developed to exploit the exchangeable nature of the learning problem in classification and result in more efficient algorithms.

Finally, by providing a general approach for combining heterogeneous data sources in the setting of kernel-based statistical learning algorithms, this line of research suggests an important role for kernel matrices as general building blocks of statistical models. Much as in the case of finite-dimensional sufficient statistics, kernel matrices generally involve a significant reduction of the data and represent the only aspects of the data that are used by subsequent algorithms. Moreover, given the panoply of methods that are available to accommodate not only the vectorial and matrix data that are familiar in classical statistical analysis, but also more exotic data types such as strings, trees and graphs, kernel matrices have an appealing universality. It is natural to envision libraries of kernel matrices in fields such as bioinformatics, computational vision, and information retrieval, in which multiple data sources abound. Such libraries would summarize the statistically-relevant features of primary data, and encapsulate domain specific knowledge. Tools such as the semidefinite programming methods that we have presented here can be used to bring these multiple data sources together in novel ways to make predictions and decisions.

Acknowledgements

We acknowledge support from ONR MURI N00014-00-1-0637 and NSF grant IIS-9988642. Sincere thanks to Tijl De Bie for helpful conversations and suggestions, as well as to Lijuan Cai and Thomas Hofmann for providing the data for the Reuters-21578 experiments.

Appendix A. Proof of Result (54)

For the case $K_i = \mathbf{v}_i\mathbf{v}_i^T$, with $\mathbf{v}_i$ orthonormal, the original learning problem (52) becomes

$$\begin{aligned} \max_{K} \quad & \left\langle K_{tr}, \mathbf{y}\mathbf{y}^T \right\rangle_F \\ \text{subject to} \quad & \langle K, K\rangle_F \leq 1, \\ & K \succeq 0, \\ & K = \sum_{i=1}^{m} \mu_i \mathbf{v}_i \mathbf{v}_i^T. \end{aligned} \tag{61}$$

Expanding this further gives

$$\begin{aligned} \left\langle K_{tr}, \mathbf{y}\mathbf{y}^T \right\rangle_F &= \operatorname{trace}(K(1:n_{tr}, 1:n_{tr})\mathbf{y}\mathbf{y}^T) \\ &= \operatorname{trace}((\sum_{i=1}^{m} \mu_i \mathbf{v}_i(1:n_{tr})\mathbf{v}_i(1:n_{tr})^T)\mathbf{y}\mathbf{y}^T) \\ &= \sum_{i=1}^{m} \mu_i \operatorname{trace}(\bar{\mathbf{v}}_i\bar{\mathbf{v}}_i^T\mathbf{y}\mathbf{y}^T) \\ &= \sum_{i=1}^{m} \mu_i (\bar{\mathbf{v}}_i^T\mathbf{y})^2, \end{aligned} \tag{62}$$

$$\begin{aligned} \langle K, K\rangle_F &= \operatorname{trace}(K^T K) \\ &= \operatorname{trace}(KK) \\ &= \operatorname{trace}((\sum_{i=1}^{m} \mu_i \mathbf{v}_i\mathbf{v}_i^T)(\sum_{j=1}^{m} \mu_j \mathbf{v}_j\mathbf{v}_j^T)) \\ &= \operatorname{trace}(\sum_{i,j=1}^{m} \mu_i\mu_j \mathbf{v}_i\mathbf{v}_i^T\mathbf{v}_j\mathbf{v}_j^T) \\ &= \operatorname{trace}(\sum_{i=1}^{m} \mu_i^2 \mathbf{v}_i\mathbf{v}_i^T) \\ &= \sum_{i=1}^{m} \mu_i^2 \operatorname{trace}(\mathbf{v}_i\mathbf{v}_i^T) \\ &= \sum_{i=1}^{m} \mu_i^2 \operatorname{trace}(\mathbf{v}_i^T\mathbf{v}_i) \\ &= \sum_{i=1}^{m} \mu_i^2 \end{aligned} \tag{63}$$

with $\bar{\mathbf{v}}_i = \mathbf{v}_i(1:n_{tr})$. We used the fact that $\operatorname{trace}(ABC) = \operatorname{trace}(BCA)$ (if the products are well-defined) and that the vectors $\mathbf{v}_i, i = 1, \ldots, n$ are orthonormal: $\mathbf{v}_i^T\mathbf{v}_j = \delta_{ij}$. Furthermore, because the $\mathbf{v}_i$ are orthogonal, the μ_i in $K = \sum_{i=1}^{m} \mu_i \mathbf{v}_i\mathbf{v}_i^T$ are the eigenvalues of K. This implies

$$K \succeq 0 \Leftrightarrow \boldsymbol{\mu} \geq 0 \Leftrightarrow \mu_i \geq 0, \quad i = 1, \ldots, m. \tag{64}$$

Using (62), (63) and (64) in (61), we obtain the following optimization problem:

$$\begin{array}{ll} \max_{\mu_i} & \sum_{i=1}^{m} \mu_i (\bar{\mathbf{v}}_i^T \mathbf{y})^2 \\ \text{subject to} & \sum_{i=1}^{m} \mu_i^2 \leq 1 \\ & \mu_i \geq 0, \quad i = 1, \ldots, m, \end{array}$$

which yields the result (54).

Appendix B. Proof of Theorem 24

For a function $g : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$, define

$$\begin{aligned} \hat{\mathbf{E}}_1 g(\mathbf{X}, \mathbf{Y}) &= \frac{1}{n} \sum_{i=1}^{n} g(\mathbf{x}_i, y_i), \\ \hat{\mathbf{E}}_2 g(\mathbf{X}, \mathbf{Y}) &= \frac{1}{n} \sum_{i=1}^{n} g(\mathbf{x}_{n+i}, y_{n+i}). \end{aligned}$$

Define a margin cost function $\phi : \mathbb{R} \to \mathbb{R}^+$ as

$$\phi(a) = \begin{cases} 1 & \text{if } a \leq 0, \\ 1 - a & 0 < a \leq 1, \\ 0 & a > 1. \end{cases}$$

Notice that in the 1-norm soft margin cost function, the slack variable ξ_i is a convex upper bound on $\phi(y_i f(x_i))$ for the kernel classifier f, that is,

$$\max\{1 - a, 0\} \geq \phi(a) \geq \mathbf{1}[a \leq 0],$$

where the last expression is the indicator function of $a \leq 0$.

The proof of the first part is due to Koltchinskii and Panchenko Koltchinskii and Panchenko (2002), and involves the following five steps:

Step 1. For any class F of real functions defined on $\mathcal{X}$,

$$\sup_{f \in F} \operatorname{er}(f) - \hat{\mathbf{E}}_1 \phi(Y f(\mathbf{X})) \leq \sup_{f \in F} \hat{\mathbf{E}}_2 \phi(Y f(\mathbf{X})) - \hat{\mathbf{E}}_1 \phi(Y f(\mathbf{X})).$$

To see this, notice that $\operatorname{er}(f)$ is the average over the test set of the indicator function of $Y f(\mathbf{X}) \leq 0$, and that $\phi(Y f(\mathbf{X}))$ bounds this function.

Step 2. For any class G of $[0, 1]$-valued functions,

$$\Pr\left(\sup_{g \in G} \hat{\mathbf{E}}_2 g - \hat{\mathbf{E}}_1 g \geq \mathbf{E}\left(\sup_{g \in G} \hat{\mathbf{E}}_2 g - \hat{\mathbf{E}}_1 g \right) + \epsilon \right) \leq \exp\left(\frac{-\epsilon^2 n}{4} \right),$$

where the expectation is over the random permutation. This follows from McDiarmid's inequality. To see this, we need to define the random permutation π using a set of $2n$ independent random variables. To this end, choose $\pi_1, \ldots, \pi_{2n}$ uniformly at random from the interval $[0, 1]$. These are almost surely distinct. For $j = 1, \ldots, 2n$, define $\pi(j) = |\{i : \pi_i \leq \pi_j\}|$, that is, $\pi(j)$ is the position of π_j when the random variables are ordered by size. It is easy to see that, for any g, $\hat{\mathbf{E}}_2 g - \hat{\mathbf{E}}_1 g$ changes by no more than $2/n$ when one of the π_i changes. McDiarmid's bounded difference inequality (McDiarmid, 1989) implies the result.

Step 3. For any class G of $[0,1]$-valued functions,

$$\mathbf{E}\left(\sup_{g\in G} \hat{\mathbf{E}}_2 g - \hat{\mathbf{E}}_1 g\right) \leq \hat{R}_{2n}(G) + \frac{4}{\sqrt{n}},$$

where $\hat{R}_{2n}(G) = \mathbf{E}\sup_{g\in G} \frac{1}{n}\sum_{i=1}^{2n} \sigma_i g(\mathbf{X}_i, Y_i)$, and the expectation is over the independent, uniform, $\{\pm 1\}$-valued random variables $\sigma_1, \ldots, \sigma_{2n}$. This result is essentially Lemma 3 of (Bartlett and Mendelson, 2002); that lemma contained a similar bound for i.i.d. data, but the same argument holds for fixed data, randomly permuted.

Step 4. If the class F of real-valued functions defined on $\mathcal{X}$ is closed under negations, $\hat{R}_{2n}(\phi \circ F) \leq \hat{R}_{2n}(F)$, where each $f \in F$ defines a $g \in \phi \circ F$ by $g(\mathbf{x}, y) = \phi(yf(\mathbf{x}))$. This bound is the contraction lemma of Ledoux and Talagrand (1991).

Step 5. For the class $F_\mathcal{K}$ of kernel expansions, notice (as in the proof of Lemma 26 of Bartlett and Mendelson (2002)) that

$$\begin{aligned}
\hat{R}_{2n}(F_\mathcal{K}) &= \frac{1}{n}\mathbf{E}\max_{f\in F_\mathcal{K}} \sum_{i=1}^{2n} \sigma_i f(\mathbf{X}_i) \\
&= \frac{1}{n}\mathbf{E}\max_{K\in\mathcal{K}} \max_{\|\mathbf{w}\|\leq 1/\gamma} \langle \mathbf{w}, \sum_{i=1}^{2n} \sigma_i \Phi(\mathbf{X}_i)\rangle \\
&= \frac{1}{n\gamma}\mathbf{E}\max_{K\in\mathcal{K}} \left\| \sum_{i=1}^{2n} \sigma_i \Phi(\mathbf{X}_i)\right\| \\
&\leq \frac{1}{n\gamma}\sqrt{\mathbf{E}\max_{K\in\mathcal{K}} \boldsymbol{\sigma}^T K \boldsymbol{\sigma}} \\
&= \frac{1}{n\gamma}\sqrt{\mathcal{C}(\mathcal{K})},
\end{aligned}$$

where $\boldsymbol{\sigma} = (\sigma_1, \ldots, \sigma_{2n})$ is the vector of Rademacher random variables.

Combining gives the first part of the theorem. For the second part, consider

$$\mathcal{C}(\mathcal{K}_c) = \mathbf{E}\max_{K\in\mathcal{K}_c} \boldsymbol{\sigma}^T K \boldsymbol{\sigma} = \mathbf{E}\max_{\boldsymbol{\mu}} \sum_{j=1}^{m} \mu_j \boldsymbol{\sigma}^T K_j \boldsymbol{\sigma},$$

where the max is over $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_m)$ for which the matrix $K = \sum_{j=1}^m \mu_j K_j$ satisfies the conditions $K \succeq 0$ and $\operatorname{trace}(K) \leq c$. Now,

$$\operatorname{trace}(K) = \sum_{j=1}^{m} \mu_j \operatorname{trace}(K_j),$$

and each trace in the sum is positive, so the supremum must be achieved for $\operatorname{trace}(K) = c$. So we can write

$$\mathcal{C}(\mathcal{K}_c) = c\mathbf{E}\max_{K\in\mathcal{K}_c} \sum_{j=1}^{m} \boldsymbol{\sigma}^T \frac{K}{\operatorname{trace}(K)} \boldsymbol{\sigma}.$$

Notice that $\boldsymbol{\sigma}^T K \boldsymbol{\sigma}$ is no more than $\lambda \|\boldsymbol{\sigma}\|^2 = n\lambda$, where λ is the maximum eigenvalue of K. Using $\lambda \leq \mathrm{trace}(K) = c$ shows that $\mathcal{C}(\mathcal{K}_c) \leq cn$.

Finally, for $\mathcal{K}_c^+$ we have

$$\begin{aligned}\mathcal{C}(\mathcal{K}_c^+) &= \mathbf{E} \max_{\mathcal{K} \in \mathcal{K}_c^+} \boldsymbol{\sigma}^T K \boldsymbol{\sigma} \\ &= \mathbf{E} \max_{\mu_j} \sum_{j=1}^{m} \mu_j \boldsymbol{\sigma}^T K_j \boldsymbol{\sigma} \\ &= \mathbf{E} \max_j \frac{c}{\mathrm{trace}(K_j)} \boldsymbol{\sigma}^T K_j \boldsymbol{\sigma}.\end{aligned}$$

Since each term in the maximum is non-negative, we can replace it with a sum to show that

$$\begin{aligned}\mathcal{C}(\mathcal{K}_c^+) &\leq c\mathbf{E}\boldsymbol{\sigma}^T \left(\sum_j \frac{K_j}{\mathrm{trace}(K_j)} \right) \boldsymbol{\sigma} \\ &= cm.\end{aligned}$$

Alternatively, we can write $\boldsymbol{\sigma}^T K_j \boldsymbol{\sigma} \leq \lambda_j \|\boldsymbol{\sigma}\| = \lambda_j n$, where λ_j is the maximum eigenvalue of K_j. This shows that

$$\mathcal{C}(\mathcal{K}_c^+) \leq cn \max_j \frac{\lambda_j}{\mathrm{trace}(K_j)}.$$

References

Andersen, E. D. and Andersen, A. D. (2000). The MOSEK interior point optimizer for linear programming: An implementation of the homogeneous algorithm. In Frenk, H., Roos, C., Terlaky, T., and Zhang, S., editors, *High Performance Optimization*, pages 197–232. Kluwer Academic Publishers.

Bartlett, P. L. (1998). The sample complexity of pattern classification with neural networks: The size of the weights is more important than the size of the network. *IEEE Transactions on Information Theory*, 44(2):525–536.

Bartlett, P. L. and Mendelson, S. (2002). Rademacher and Gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3:463–482.

Bennett, K. P. and Bredensteiner, E. J. (2000). Duality and geometry in SVM classifiers. In *Proceedings of the 17th International Conference on Machine Learning*, pages 57–64. Morgan Kaufmann.

Boyd, S. and Vandenberghe, L. (2003). Convex optimization. Course notes for EE364, Stanford University. Available at `http://www.stanford.edu/class/ee364`.

Breiman, L. (1998). Arcing classifiers. *Annals of Statistics*, 26(3):801–849.

Cai, L. and Hofmann, T. (2003). Text categorization by boosting automatically extracted concepts. In *Proceedings of the 26th ACM-SIGIR International Conference on Research and Development in Information Retrieval*. ACM Press.

Cristianini, N., Kandola, J., Elisseeff, A., and Shawe-Taylor, J. (2001). On kernel target alignment. Technical Report NeuroColt 2001-099, Royal Holloway University London.

Cristianini, N. and Shawe-Taylor, J. (2000). *An Introduction to Support Vector Machines*. Cambridge University Press.

Cristianini, N., Shawe-Taylor, J., Elisseeff, A., and Kandola, J. (2002). On kernel-target alignment. In Dietterich, T. G., Becker, S., and Ghahramani, Z., editors, *Advances in Neural Information Processing Systems 14*. MIT Press.

De Bie, T., Lanckriet, G., and Cristianini, N. (2003). Convex tuning of the soft margin parameter. Technical Report CSD-03-1289, University of California, Berkeley.

Deng, M., Chen, T., and Sun, F. (2003). An integrated probabilistic model for functional prediction of proteins. In *RECOMB*, pages 95–103.

Eyheramendy, S., Genkin, A., Ju, W., Lewis, D. D., and Madigan, D. (2003). Sparse bayesian classifiers for text categorization. Technical report, Department of Statistics, Rutgers University.

Huang, Y. (2003). Support vector machines for text categorization based on latent semantic indexing. Technical report, Electrical and Computer Engineering Department, The Johns Hopkins University.

Koltchinskii, V. and Panchenko, D. (2002). Empirical margin distributions and bounding the generalization error of combined classifiers. *Annals of Statistics*, 30.

Kondor, R. I. and Lafferty, J. (2002). Diffusion kernels on graphs and other discrete input spaces. In Sammut, C. and Hoffmann, A., editors, *Proceedings of the International Conference on Machine Learning*. Morgan Kaufmann.

Lanckriet, G. R. G., Deng, M., Cristianini, N., Jordan, M. I., and Noble, W. S. (2004). Kernel-based data fusion and its application to protein function prediction in yeast. In *Pacific Symposium on Biocomputing*.

Ledoux, M. and Talagrand, M. (1991). *Probability in Banach Spaces: Isoperimetry and Processes*. Springer-Verlag.

McDiarmid, C. (1989). On the method of bounded differences. In *Surveys in Combinatorics 1989*, pages 148–188. Cambridge University Press.

Nesterov, Y. and Nemirovsky, A. (1994). *Interior Point Polynomial Methods in Convex Programming: Theory and Applications*. SIAM.

Platt, J. (1999). Using sparseness and analytic QP to speed training of support vector machines. In M. S. Kearns, S. A. Solla, D. A. C., editor, *Advances in Neural Information Processing Systems 11*. MIT Press.

Salton, G. and McGill, M. J. (1983). *Introduction to Modern Information Retrieval*. McGraw-Hill.

Schölkopf, B. and Smola, A. (2002). *Learning with Kernels*. MIT Press.

Seeger, M. (2002). PAC-Bayesian generalization error bounds for Gaussian process classification. Technical Report EDI-INF-RR-0094, University of Edinburgh, Division of Informatics.

Shawe-Taylor, J. and Cristianini, N. (1999). Soft margin and margin distribution. In Smola, A., Schölkopf, B., Bartlett, P., and Schuurmans, D., editors, *Advances in Large Margin Classifiers*. MIT Press.

Shawe-Taylor, J. and Cristianini, N. (2004). *Kernel Methods for Pattern Analysis.* Cambridge University Press.

Smith, T. F. and Waterman, M. S. (1981). Identification of common molecular subsequences. *Journal of Molecular Biology*, 147(1):195–197.

Sturm, J. F. (1999). Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 11–12:625–653. Special issue on Interior Point Methods (CD supplement with software).

Tsuda, K. (1999). Support vector classification with asymmetric kernel function. In Verleysen, M., editor, *Proceedings of the European Symposium on Artificial Neural Networks*, pages 183–188.

Vandenberghe, L. and Boyd, S. (1996). Semidefinite programming. *SIAM Review*, 38(1):49–95.

Vandenberghe, L., Boyd, S., and Wu, S.-P. (1998). Determinant maximization with linear matrix inequality constraints. *SIAM Journal on Matrix Analysis and Applications*, 19(2):499–533.