Multiclass Multiple Kernel Learning

Alexander Zien ALEXANDER.ZIEN@TUEBINGEN.MPG.DE
Cheng Soon Ong CHENGSOON.ONG@TUEBINGEN.MPG.DE
Max Planck Inst. for Biol. Cybernetics and Friedrich Miescher Lab., Spemannstr. 39, Tübingen, Germany.

Abstract

In many applications it is desirable to learn from several kernels. "Multiple kernel learning" (MKL) allows the practitioner to optimize over linear combinations of kernels. By enforcing sparse coefficients, it also generalizes feature selection to kernel selection. We propose MKL for joint feature maps. This provides a convenient and principled way for MKL with multiclass problems. In addition, we can exploit the joint feature map to learn kernels on output spaces. We show the equivalence of several different primal formulations including different regularizers. We present several optimization methods, and compare a convex quadratically constrained quadratic program (QCQP) and two semi-infinite linear programs (SILPs) on toy data, showing that the SILPs are faster than the QCQP. We then demonstrate the utility of our method by applying the SILP to three real world datasets.

1. Introduction

In support vector machines (SVMs), a kernel function k implicitly maps examples $\mathbf{x}$ to a feature space given by a feature map Φ via the identity $k(\mathbf{x}_i, \mathbf{x}_j) = \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle$ (e.g. [19]). It is often unclear what the most suitable kernel for the task at hand is, and hence the user may wish to combine several possible kernels. One problem with simply adding kernels is that using uniform weights is possibly not optimal. An extreme example is the case that one kernel is not correlated with the labels at all – then giving it positive weight just adds noise [13]. Multiple kernel learning (MKL) is a way of optimizing kernel weights while training the SVM. In addition to leading to good classification accuracies, MKL can also be useful for identifying relevant and meaningful features [2, 13, 20].

Appearing in *Proceedings of the 24th International Conference on Machine Learning*, Corvallis, OR, 2007. Copyright 2007 by the author(s)/owner(s).

Since in many real world applications more than two classes are to be distinguished, there has been a lot of work on decomposing multiclass problems into several standard binary classification problems, or developing genuine multiclass classifiers. The latter has so far been restricted to single kernels. In this paper we develop and investigate a multiclass extension of MKL. We provide:

- An intuitive formulation of the multiclass MKL task, with a new sparsity-promoting regularizer.
- A proof of equivalence with (the multiclass extension of) a previous MKL formulation.
- Several optimization approaches, with a computational comparison between three of them.
- Experimental results on several datasets demonstrating the method's utility.

2. Multiclass Multiple Kernel Learning

A common approach (e.g. [21]) to multiclass classification is the use of joint feature maps $\Phi(\mathbf{x}, y)$ on data $\mathcal{X}$ and labels $\mathcal{Y} = \{1, \ldots, m\}$, $m > 2$, together with a linear output function

$$f_{\mathbf{w},\mathbf{b}}(\mathbf{x}, y) = \langle \mathbf{w}, \Phi(\mathbf{x}, y) \rangle + b_y \ , \tag{1}$$

parameterized with the hyperplane normal $\mathbf{w}$ and biases $\mathbf{b}$. Here, $\mathbf{b}$ is short for the stacked variables $(b_1, \ldots, b_m)$; we will use analoguous notation throughout the paper. The predicted class y for a point $\mathbf{x}$ is chosen to maximize the output,

$$\mathbf{x} \mapsto \arg\max_{y \in \mathcal{Y}} f_{\mathbf{w},\mathbf{b}}(\mathbf{x}, y).$$

Hence for a suitable convex loss function ℓ, training can be implemented by the following convex multiclass support vector machine (m-SVM) optimization problem (OP):

$$\min_{\mathbf{w},\mathbf{b}} \frac{1}{2} \|\mathbf{w}\|^2 + \sum_{i=1}^{n} \max_{u \neq y_i} \left\{ \ell \left(f_{\mathbf{w},\mathbf{b}}(\mathbf{x}_i, y_i) - f_{\mathbf{w},\mathbf{b}}(\mathbf{x}_i, u) \right) \right\} \ . \tag{2}$$

Here n is the size of the training set, and $u \neq y_i$ is short for $u \in \mathcal{Y} - \{y_i\}$. For the hinge loss, $\ell(t) := C \max(0, 1-t)$, the dual of this is a well-kown quadratic program (QP) [21].

2.1. Multiple Kernel Learning (MKL) Primal

We can generalize the above m-SVM further to operate on $p \geqslant 1$ feature maps $\Phi_k(\mathbf{x}_i, y_i)$ (for $k = 1, \ldots, p$). For each feature map there will be a separate weight vector $\mathbf{w}_k$. Here we consider linear combinations of the corresponding output functions:

$$f_{\mathbf{w},\mathbf{b},\boldsymbol{\beta}}(\mathbf{x}, y) \;=\; \sum_{k=1}^{p} \beta_k \langle \mathbf{w}_k, \Phi_k(x, y) \rangle + b_y \;.$$

This corresponds to direct sums of feature spaces of the form $\tilde{\Phi}(\mathbf{x}_i, y_i) = (\beta_k \Phi_k(\mathbf{x}_i, y_i))_{k=1,\ldots,p}$. The mixing coefficients $\boldsymbol{\beta}$ should reflect the utility of the respective feature map for the classification task.

We aim at choosing $\mathbf{w} = (\mathbf{w}_k)_{k=1,\ldots,p}$ and $\boldsymbol{\beta}$ such that $f_{\mathbf{w},\mathbf{b},\boldsymbol{\beta}}(\mathbf{x}_i, y_i) \geqslant f_{\mathbf{w},\mathbf{b},\boldsymbol{\beta}}(\mathbf{x}_i, u)$ for all $u \in \mathcal{Y} - \{y_i\}$. The resulting OP, a generalization of (2), can be written as

$$\begin{aligned} \min_{\boldsymbol{\beta},\mathbf{w},\mathbf{b},\xi} \quad & \frac{1}{2} \sum_{k=1}^{p} \beta_k \|\mathbf{w}_k\|^2 + \sum_{i=1}^{n} \xi_i \\ \text{s.t.} \quad & \forall i : \xi_i = \max_{u \neq y_i} \ell \left(f_{\mathbf{w},\mathbf{b},\boldsymbol{\beta}}(\mathbf{x}_i, y_i) - f_{\mathbf{w},\mathbf{b},\boldsymbol{\beta}}(\mathbf{x}_i, u) \right) \end{aligned} \tag{3}$$

where we regularize the p output functions according to their weights β_k. In this paper the weights are understood to be on standard simplex, i.e.

$$\boldsymbol{\beta} \in \Delta^p := \left\{ \boldsymbol{\beta} \,\middle|\, \sum_{k=1}^{p} \beta_k = 1, \forall k : 0 \leq \beta_k \right\},$$

giving them the flavor of probabilities. This $\mathcal{L}_1$ regularizer on $\boldsymbol{\beta}$ promotes sparsity, and hence we are trying to select a subset of kernels.

While the OP (3) is interpretable and intuitive, it has the disadvantage of in general not being convex due to the products of β_k and $\mathbf{w}_k$ in the output function. We thus apply a change of variables transformation (cf. [4, Section 4.1.3]) with $\mathbf{v}_k := \beta_k \mathbf{w}_k$. For a convex loss, the resulting OP (below) is convex.

$$\begin{aligned} \inf_{\boldsymbol{\beta},\mathbf{v},\mathbf{b},\xi} \quad & \frac{1}{2} \sum_{k=1}^{p} \|\mathbf{v}_k\|^2 / \beta_k + \sum_{i=1}^{n} \xi_i \\ \text{s.t.} \quad & \forall i : \xi_i = \max_{u \neq y_i} \ell \left(\; \langle \mathbf{v}, \Psi_{iu} \rangle + b_{y_i} - b_u \; \right) \end{aligned} \tag{4}$$

where we define $\Psi_{kiu} = \Phi_k(\mathbf{x}_i, y_i) - \Phi_k(\mathbf{x}_i, u)$ and $\Psi_{iu} = (\Psi_{kiu})_{k=1,\ldots,p}$.

Beyond easing the interpretation of the weights, the constraints on $\boldsymbol{\beta}$ are essential for technical reasons. Their non-negativity guarantees that the combined regularizer is convex, and the resulting kernel is positive semidefinite. Further, without limiting the norm of $\boldsymbol{\beta}$, the regularizer on $\mathbf{w}$ would not be effective: it could be driven to zero without changing $f_{\mathbf{w},\mathbf{b},\boldsymbol{\beta}}$ by dividing $\mathbf{w}$ by any positive scalar while multiplying $\boldsymbol{\beta}$ with the same number.

2.2. MKL Dual (General and Hinge Loss)

Key to the findings and algorithms in this paper is the dual of the MKL optimization problem. Using a more general proof technique than [20] (cf. appendix), we can find the dual of (4) for any convex loss function ℓ without having to require differentiability.

Theorem 1 *Let ℓ^* be the conjugate function of the given convex loss function ℓ, and δ_{ab} be the indicator function of $a = b$. The dual of the convex MKL problem (4) is equivalent to*

$$\begin{aligned} \inf_{\boldsymbol{\eta},\tilde{\boldsymbol{\alpha}},\gamma} \quad & \gamma + \sum_i \sum_{u \neq y_i} \eta_{iu} \ell^* \left(-\frac{\tilde{\alpha}_{iu}}{\eta_{iu}} \right) \\ \text{s.t.} \quad & \forall k : \gamma \geqslant \frac{1}{2} \sum_{i,j} \sum_{u \neq y_i} \sum_{v \neq y_j} \tilde{\alpha}_{iu} \tilde{\alpha}_{jv} \langle \Psi_{kiu}, \Psi_{kjv} \rangle \,, \\ & \forall i : \forall u \neq y_i : 0 \leqslant \eta_{iu}, \;\; \textit{and} \; \forall i : 1 = \sum_{u \neq y_i} \eta_{iu}, \\ & \forall v : 0 = \sum_i (1 - \delta_{y_i v}) \tilde{\alpha}_{iv} - \sum_i \delta_{y_i v} \sum_{u \neq y_i} \tilde{\alpha}_{iu} \end{aligned} \tag{5}$$

with $\gamma \in \mathbb{R}$, $\tilde{\boldsymbol{\alpha}} \in \mathbb{R}^{n \times (m-1)}$, $\boldsymbol{\eta} \in \mathbb{R}^{n \times (m-1)}$. Observe that the weight vector of the primal is obtained by

$$\mathbf{w}_k = \mathbf{w}_k(\tilde{\boldsymbol{\alpha}}) := \sum_i \sum_{u \neq y_i} \tilde{\alpha}_{iu} \Psi_{kiu} \;. \tag{6}$$

Now we plug the standard SVM loss function, the hinge loss $\ell(t) := C \max(0, 1-t)$, into (5). We obtain:

$$\begin{aligned} \min_{\tilde{\boldsymbol{\alpha}},\gamma} \quad & \gamma - \sum_i \sum_{u \neq y_i} \tilde{\alpha}_{iu} \\ \text{s.t.} \quad & \forall k : \gamma \geqslant \frac{1}{2} \|\mathbf{w}_k(\tilde{\boldsymbol{\alpha}})\|^2 \\ & \sum_{u \neq y_i} \tilde{\alpha}_{iu} \leq C \qquad \forall u \neq y_i : 0 \leq \tilde{\alpha}_{iu} \\ & \forall v : 0 = \sum_i (1 - \delta_{y_i v}) \tilde{\alpha}_{iv} - \sum_i \delta_{y_i v} \sum_{u \neq y_i} \tilde{\alpha}_{iu} \end{aligned} \tag{7}$$

By introducing $\alpha \in \mathbb{R}^{n \times m}$ via the substitution

$$\alpha_{iu} \;=\; \begin{cases} -\tilde{\alpha}_{iu} & \text{if } u \neq y_i \\ \sum_{v \neq y_i} \tilde{\alpha}_{iv} & \text{if } u = y_i \end{cases} \tag{8}$$

we can equivalently rewrite equation (7) into

$$
\begin{aligned}
\min_{\boldsymbol{\alpha},\gamma} \quad & \gamma - \sum_i \alpha_{iy_i} \\
\text{s.t.} \quad & \forall k : \gamma \geqslant \frac{1}{2} \left\| \mathbf{w}_k(\boldsymbol{\alpha}) \right\|^2 \\
& \boldsymbol{\alpha} \in \mathcal{S} := \left\{ \boldsymbol{\alpha} \;\middle|\; \begin{array}{l} \forall i : 0 \leqslant \alpha_{iy_i} \leqslant C \\ \forall i : \forall u \neq y_i : \alpha_{iu} \leqslant 0 \\ \forall i : \sum_{u\in\mathcal{Y}} \alpha_{iu} = 0 \\ \forall u \in \mathcal{Y} : \sum_i \alpha_{iu} = 0 \end{array} \right\}
\end{aligned}
\tag{9}
$$

with the correspondingly transformed, "unfolded", expansion for the hyperplane normal,

$$
\mathbf{w}_k \quad = \quad \sum_i \sum_{y\in\mathcal{Y}} \alpha_{iu} \Phi_k(\mathbf{x}_i, u) \ . \tag{10}
$$

Both versions of the dual, the compact (7) and the unfolded (9), are quadratically constrained quadratic programs (QCQPs).

2.3. Relation to Previous Work

There have been several developments on optimizing a linear combination of kernels while training a predictor [1, 3, 6, 8, 12, 14, 15, 20] ([16, 18] considers general parameterized kernel functions). To show the relationship of our approach to two previous approaches [1, 20], we consider the unfolded dual.

For the case of a single kernel, $p = 1$, there is a single quadratic constraint in (9) in addition to those of the original m-SVM dual. We observe that at the optimum this constraint will always be at equality, so that we can substitute its right hand side back into the objective. This way, we exactly obtain the dual of the m-SVM (4), as expected.

Our initial motivation was to propose an intuitive primal problem (3) that explicitly models the weights $\boldsymbol{\beta}$ of the given kernels (or, alternatively, feature maps). This is in contrast to the primal proposed in [1] in which no such coefficients appear. To our surprise, for the special case of $m = 2$ classes we find that our dual (9) is identical to the dual derived in [1]. Further, their method can be extended for multiple classes such that it yields exactly the same dual as our OP (4).

Figure 1 provides an overview over several multiclass MKL formulations and their relationships. The left column shows our two primals (3) and (4). The right column presents multiclass generalizations of the optimization problems given in [1, 20], which originally are binary. Within each column the primals are identical up to a variable transformation, hence equivalent. The two convex problems are equivalent as they share the same dual (and strong duality holds, cf. [4]). By the chain of equivalences, all shown OPs are equivalent, despite their different regularizers.

2.4. Optimization

Recently unconstrained primal optimization is emerging as a promising machine learning technique. For MKL this approach might be most convenient with the OP from [1] (cf. Figure 1), as it does not include $\boldsymbol{\beta}$ and the associated constraints. However, here we follow the traditional MKL route along the dual.

One possibility is to solve either version of the dual directly using an off-the-shelf solver. Equations (7) and (9) are QCQPs. In general, a QCQP can be solved much more efficiently than an SDP with interior point methods due to the added structure of the problem [4]. In Section 3.1 we investigate this approach utilizing the commercial optimization package CPLEX that offers the barrier method for QCQPs.

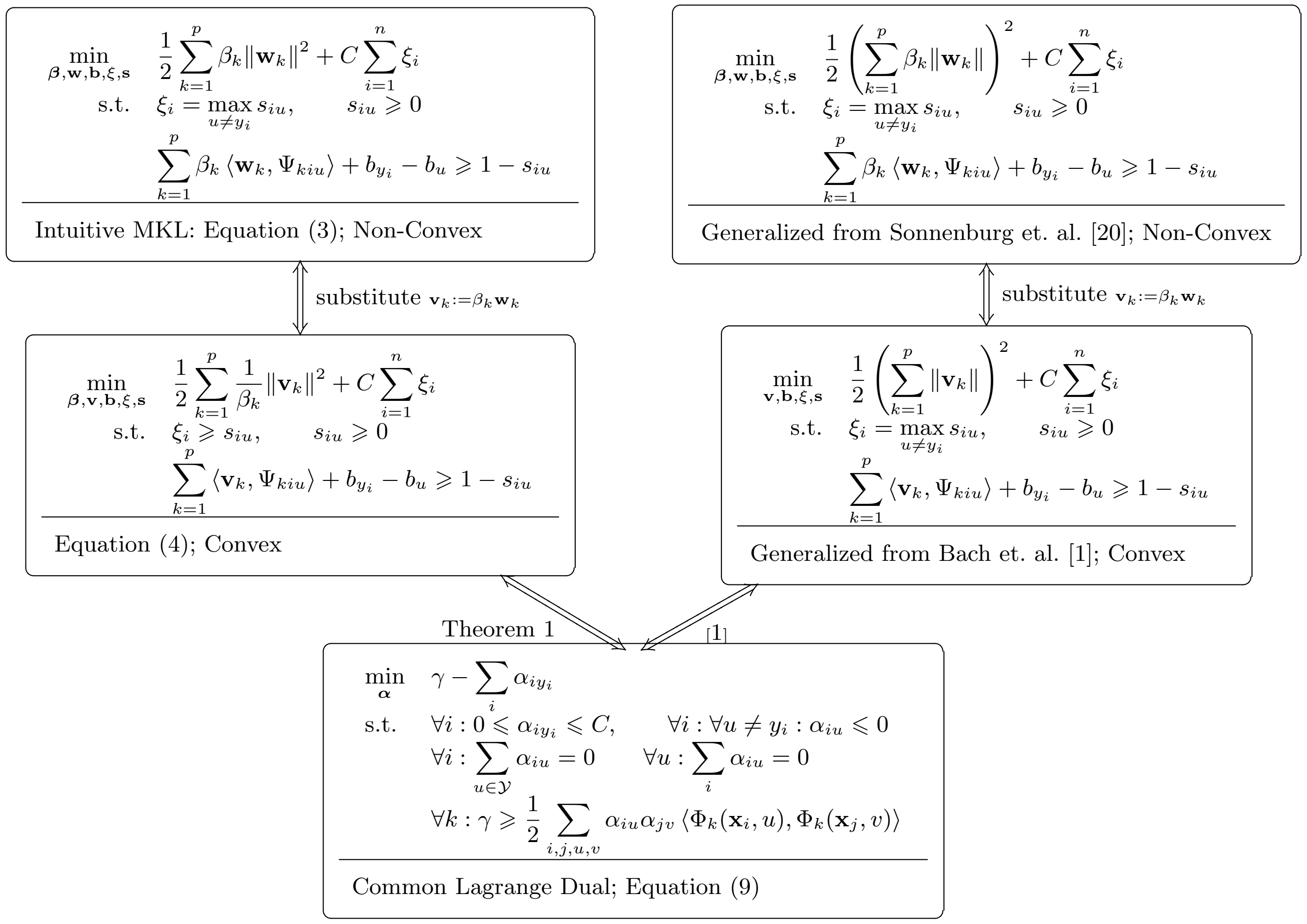
In an alternative approach, following [20], we convert both QCQPs into equivalent semi-infinite linear program (SILP) formulations by a second (partial) dualization with respect to γ while keeping all the other constraints. For the unfolded QCQP (9) we obtain:

$$
\begin{aligned}
\max_{\boldsymbol{\beta}\in\Delta^p,\theta} \quad & \theta \\
\text{s.t.} \quad & \forall \boldsymbol{\alpha} \in \mathcal{S} : \ \ \theta \leqslant \frac{1}{2} \sum_k \beta_k \left\| \mathbf{w}_k(\boldsymbol{\alpha}) \right\|^2 - \sum_i \alpha_{iy_i} \ ,
\end{aligned}
\tag{11}
$$

where $\mathcal{S}$ is defined as in Equation (9). (Analoguously for the compact QCQP (7).)

Contrary to (9), solving the problem (11) yields the values for $\boldsymbol{\beta}$, but not for $\boldsymbol{\alpha}$. However, once we know the optimal $\boldsymbol{\beta}$, the problem reduces to the m-SVM dual with a correspondingly mixed kernel, and $\boldsymbol{\alpha}$ and $\mathbf{b}$ can be determined by solving the m-SVM. As pointed out in [20], this approach is beneficial when there already exists an efficient solver for the corresponding QP. Furthermore, for fixed $\boldsymbol{\alpha}$, the optimization problem in $\boldsymbol{\beta}$ is a linear program (LP). However, the constraint on θ has to hold for every suitable $\boldsymbol{\alpha}$; hence the name (referring to the infinitely many constraints).

This suggests the use of a column generation strategy to solve (11): Solving the QP resulting from a fixed $\boldsymbol{\beta}$ yields a particular $\boldsymbol{\alpha}$, which then gives rise to a constraint on θ which is linear in $\boldsymbol{\beta}$. We alternate generating new constraints in this way and solving the LP with the constraints collected so far. This procedure is known to converge [10, 20]. Hence our seemingly complicated problem can be solved with off-the-shelf solvers. In our implementation, we used CPLEX with the dual simplex method for both QPs and LPs.



$$\begin{aligned}\min_{\boldsymbol\beta,\mathbf{w},\mathbf{b},\xi,\mathbf{s}}\quad & \frac{1}{2}\sum_{k=1}^{p}\beta_k\|\mathbf{w}_k\|^2 + C\sum_{i=1}^{n}\xi_i\\ \text{s.t.}\quad & \xi_i = \max_{u\neq y_i} s_{iu}, \qquad s_{iu}\geqslant 0\\ & \sum_{k=1}^{p}\beta_k\left\langle \mathbf{w}_k,\Psi_{kiu}\right\rangle + b_{y_i} - b_u \geqslant 1 - s_{iu}\end{aligned}$$

Intuitive MKL: Equation (3); Non-Convex

$$\begin{aligned}\min_{\boldsymbol\beta,\mathbf{w},\mathbf{b},\xi,\mathbf{s}}\quad & \frac{1}{2}\left(\sum_{k=1}^{p}\beta_k\|\mathbf{w}_k\|\right)^2 + C\sum_{i=1}^{n}\xi_i\\ \text{s.t.}\quad & \xi_i = \max_{u\neq y_i} s_{iu}, \qquad s_{iu}\geqslant 0\\ & \sum_{k=1}^{p}\beta_k\left\langle \mathbf{w}_k,\Psi_{kiu}\right\rangle + b_{y_i} - b_u \geqslant 1 - s_{iu}\end{aligned}$$

Generalized from Sonnenburg et. al. [20]; Non-Convex

substitute $\mathbf{v}_k:=\beta_k\mathbf{w}_k$

substitute $\mathbf{v}_k:=\beta_k\mathbf{w}_k$

$$\begin{aligned}\min_{\boldsymbol\beta,\mathbf{v},\mathbf{b},\xi,\mathbf{s}}\quad & \frac{1}{2}\sum_{k=1}^{p}\frac{1}{\beta_k}\|\mathbf{v}_k\|^2 + C\sum_{i=1}^{n}\xi_i\\ \text{s.t.}\quad & \xi_i \geqslant s_{iu}, \qquad s_{iu}\geqslant 0\\ & \sum_{k=1}^{p}\left\langle \mathbf{v}_k,\Psi_{kiu}\right\rangle + b_{y_i} - b_u \geqslant 1 - s_{iu}\end{aligned}$$

Equation (4); Convex

$$\begin{aligned}\min_{\mathbf{v},\mathbf{b},\xi,\mathbf{s}}\quad & \frac{1}{2}\left(\sum_{k=1}^{p}\|\mathbf{v}_k\|\right)^2 + C\sum_{i=1}^{n}\xi_i\\ \text{s.t.}\quad & \xi_i = \max_{u\neq y_i} s_{iu}, \qquad s_{iu}\geqslant 0\\ & \sum_{k=1}^{p}\left\langle \mathbf{v}_k,\Psi_{kiu}\right\rangle + b_{y_i} - b_u \geqslant 1 - s_{iu}\end{aligned}$$

Generalized from Bach et. al. [1]; Convex

Theorem 1

[1]

$$\begin{aligned}\min_{\boldsymbol\alpha}\quad & \gamma - \sum_i \alpha_{iy_i}\\ \text{s.t.}\quad & \forall i: 0\leqslant\alpha_{iy_i}\leqslant C, \qquad \forall i:\forall u\neq y_i:\alpha_{iu}\leqslant 0\\ & \forall i:\sum_{u\in\mathcal{Y}}\alpha_{iu}=0 \qquad \forall u:\sum_i\alpha_{iu}=0\\ & \forall k:\gamma\geqslant\frac{1}{2}\sum_{i,j,u,v}\alpha_{iu}\alpha_{jv}\left\langle\Phi_k(\mathbf{x}_i,u),\Phi_k(\mathbf{x}_j,v)\right\rangle\end{aligned}$$

Common Lagrange Dual; Equation (9)

Figure 1: Equivalence of multiclass MKL optimization problems. Where present, the combination weights $\boldsymbol\beta$ are understood to be on the standard simplex, i.e. $\sum_{k=1}^{p}\beta_k = 1$, $\forall k: 0\leqslant\beta_k$. (Left) Our proposed formulation. (Right) Multiclass versions of previous formulations [1, 20]. (Row 1) Non-convex primals. (Row 2) Equivalent convex primals via the variable substitution $\mathbf{v}_k := \beta_k\mathbf{w}_k$. (Row 3) The common dual optimization problem.

Finally, MKL can be implemented by semi-definite programing (SDP) [4, 14], which however is computationally more expensive.

2.5. Decomposable Kernels

One way to define a joint kernel k is to multiply a kernel $k_\mathcal{X}$ on $\mathcal{X}$, i.e. $k_\mathcal{X}:\mathcal{X}\times\mathcal{X}\to\mathbb{R}$, with a kernel $k_\mathcal{Y}$ on $\mathcal{Y}$, i.e. $k_\mathcal{Y}:\mathcal{Y}\times\mathcal{Y}\to\mathbb{R}$:

$$k\left((\mathbf{x},y),(\mathbf{x}',y')\right) = k_\mathcal{X}(\mathbf{x},\mathbf{x}')\cdot k_\mathcal{Y}(y,y') \ . \tag{12}$$

The feature space reflects this structure: it is the tensor product of the features space on $\mathcal{X}$ with that on $\mathcal{Y}$. We will use (12) in all our experiments, mostly with the matching kernel (or identity kernel) on $\mathcal{Y}$, i.e. $k_\mathcal{Y}(y,y')=\delta_{yy'}$. This kernel imposes the least structure on the classes.

With a kernel that decomposes as in (12), the quadratic terms in the OPs (8, 12, 13) simplify to a Kronecker product of kernel matrices on $\mathcal{X}$ and $\mathcal{Y}$. Further, for our choice of the matching kernel (or other diagonal kernels) on $\mathcal{Y}$, we get

$$\begin{aligned}\|\mathbf{w}\|^2 &= \sum_i\sum_j\sum_{u\in\mathcal{Y}}\sum_{v\in\mathcal{Y}}\alpha_{iu}\alpha_{jv}\left\langle\Phi(\mathbf{x}_i,u),\Phi(\mathbf{x}_j,v)\right\rangle\\ &= \sum_i\sum_j k_\mathcal{X}(\mathbf{x}_i,\mathbf{x}_j)\sum_{u\in\mathcal{Y}}\alpha_{iu}\alpha_{ju}k_\mathcal{Y}(u,u) \ .\end{aligned}$$

Thus the joint kernel matrix in the unfolded OPs is sparse (with n^2m non-zero elements) and can be arranged into a block-diagonal form. This structure allows to save memory and computation time in comparison to the compact OPs, where the matrices may in general be full.

3. Experimental Results

For each data set, ten random splits into 80% training and 20% test data are prepared. For each training set the parameter C is chosen using 3-fold cross validation on the training set. We search over a grid of values $C = \{1/27, 1/9, 1/3, 1, 3, 9, 27\}$.

In a standard SVM with linear kernel, the scaling of the features changes the resulting weights $\mathbf{w}$ (as it is equivalent to a change of the regularizer). Analoguously, in MKL the scaling of the kernels affects their resulting coefficients $\boldsymbol{\beta}$. In [5] it is argued that a reasonable order of magnitude of the SVM regularization parameter C can be estimated as the inverse of the variance of the points in feature space. Conversely, for $C = 1$ a reasonable scaling of the kernel matrix is such that the implied variance equals one. We apply this scaling to all kernels in our experiments.

3.1. Computational Complexity

We investigate the scaling behaviour of the proposed algorithm in three directions: (1) number of training examples, (2) number of classes, and (3) number of kernels. We generate some toy data using class centers on the edges of a standard simplex and Gaussian noise with standard deviation 0.3. For example, for a 4-class problem, we use the vertices of a regular tetrahedron in 3 dimensions as the class centers. We use the Gaussian RBF kernel with the width set to the $1/|\mathcal{Y}|$ quantile of the pairwise distances of the data points; for example it is set to the 1/3 quantile for a 3-class problem. The kernels are generated by considering $\{2^0, 2^{-0.5}, 2^{0.5}, 2^{-1}, 2^1, 2^{-1.5}, 2^{1.5}, \ldots, 2^{10}\}$ times the default Gaussian width.

We report results for Equation (9), Equation (11), and the m-SVM which is Equation (9) with a single kernel, and hence a QP. We optimize all methods to a relative duality gap of 10^{-2}. In Table 1 time complexities are shown that are estimated from the slopes of log-log plots. The SILP is consistently faster and has a better scaling behaviour on the data compared to the QCQP. Note that the QCQP requires that all the kernel matrices fit in active memory, but the SILP only ever deals with the matrix of weighted sums.

Since the three OPs are equivalent, one might expect that the computational time w.r.t. increasing numbers of training examples and classes would also be the same. However, CPLEX is restricted to the barrier method for solving QCQPs, but allows other solvers such as the primal and dual simplex solvers for LPs and QPs. Other experiments on using the different solvers for QPs, for which we omit the results, show that the observed difference in computation time is in large part due to different solvers.

Table 1: Estimated slopes of log-log plots, corresponding to the order of the polynomial complexity. We measure computing time while increasing a single parameter of the base case of 300 examples, 3 classes and 3 kernels. The range of values [min,max] on a log scale used were [100,5000] examples, [3,100] classes, and [2,20] kernels. The QCQP is also a constant amount slower for any particular dataset size.

	Examples	Classes	Kernels
QP, unfolded	2.5	1.8	–
QCQP, unfolded (9)	3.0	2.0	2.3
SILP, unfolded (11)	2.4	1.7	1.1
SILP, compact	2.6	2.2	1.0

The scaling behaviour of the SILP with respect to the number of kernels depends on the number of constraints that need to be generated before the SILP converges. The number of constraints depends on the set of kernels considered. For example, if there is only one meaningful kernel and the rest are noise, the SILP converges within 2 or 3 iterations since $\boldsymbol{\beta}$, and hence $\boldsymbol{\alpha}$, ceases to change between iterations.

3.2. Sanity Check

The aim of this section is to compare our method to binary MKL on a published dataset. We use previously defined kernels for the detection of membrane proteins [13]. This task consists of 2316 examples, 7 kernels and 2 classes.[1] Since the optimization method in [13] is equivalent to the method in [1], and hence equivalent to ours, we obtain identical results (up to numerical inaccuracies).

Figure 2 shows a detailed analysis of the performance of each kernel by itself, the performance of an SVM on the average of all the kernels, and the performance of MKL. Observe that while most kernels individually perform equally well in terms of accuracy, the MKL allocates very different weights (β) to them. As observed in [13], for this particular dataset the unweighted sum of kernels performs as well as MKL, unless "noise kernels" are included.

3.3. State of the Art Accuracy

Here we demonstrate the utility of our method by applying it to another real world problem, which is highly

[1]The original dataset has 2318 examples. For two of we could not find the corresponding sequences, and hence discarded them. Also, we excluded one of the original 8 kernels, as it is random noise [13].

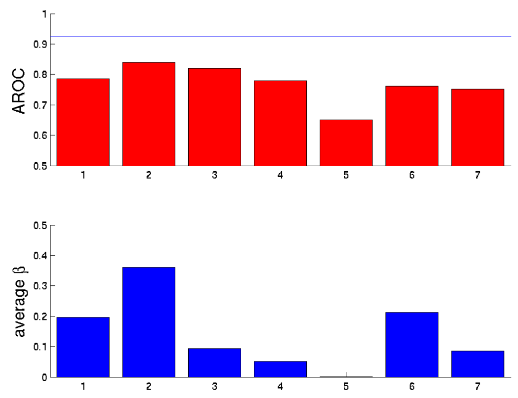


Figure 2: Accuracy and MKL weights for the binary classification task from [13], averaged over 10 random splits. The x-axis indexes the 7 kernels. (Top) Bars: test set accuracy for $C = 1$. Line: accuracy of MKL and of unweighted sum of kernels, which happen to be identical. (Bottom) Value of β as found by MKL.

relevant to cell biology: to predict the subcellular localization of proteins. We use a total of 69 kernels: 2 kernels on phylogenetic trees, 3 kernels from BLAST E-values, and 64 sequence motif kernels. When comparing our predictor to current state of the art methods, we perform substantially better. Figure 3 summarizes the results in [17] on three datasets.

The original plant dataset of TargetP [7] is classified in [11] as a 4 class problem. The method TargetLoc [11] uses three layers of SVMs, the first layer detecting certain features, and the second and third layers combining the outputs of previous SVMs. Their average performance over all classes measured in terms of Matthew's Correlation Coefficient (MCC) is 85.3%, whereas our approach achieves a MCC of 89.1%. The unweighted sum of kernels obtains 87.6%, which confirms the observation of the previous section: if all kernels are useful for the classification task, an unweighted sum often performs well. However, here multiclass MKL performs even better.

We further compare our approach to the method PSORTb on another two datasets of bacterial protein locations [9]. The psort+ dataset contains 4 classes, and PSORTb achieves an average F1 score of 90.0%, which we outperform with an F1 score of 93.8%. On psort-, a 5 class problem, PSORTb reaches an average F1 score of 87.5%, whereas we achieve 96.1%! Comparing to the F1 scores of 85.0% and 88.0% of the unweighted sum of kernels, we see that MKL again performs substantially better. However, for psort+, the probabilistic method of [9] is superior to the unweighted sum. In the ten random splits our sparsity promoting regularizer only chooses 25 different kernels.

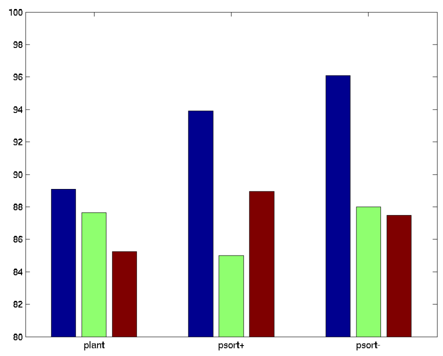


Figure 3: Protein Subcellular Localization results. The bars within each group correspond to different methods: (left, blue) MKL; (center, green) unweighted sum of kernels; (right, red) current state of the art.

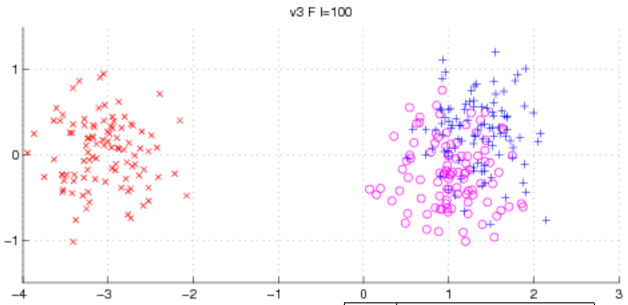
3.4. Learning the Kernel on $\mathcal{Y}$

So far we have optimized over kernels on the input data and always multiplied it with the matching kernel on the classes to obtain the joint kernel. Here we consider the reverse setting: to optimize over different kernels on $\mathcal{Y}$. Figure 4 (top) shows a toy setting with three Gaussian classes that have different pairwise relationships. For the kernels on $\mathcal{Y}$, we consider all positive semidefinite matrices that are zero except for a symetric 2×2 submatrix which is +1 along the diagonal and +1/-1 on the off-diagonal. Their convex combination allows many possible positive semidefinite matrices, although it does not span the entire space of possible output kernels. We apply MKL to the products with the linear kernel on $\mathcal{X}$. The resulting matrix $\mathbf{K}_{\mathcal{Y}}$ implies a Euclidean embedding of the classes, which in this experiment nicely reflects the arrangement of the class centers (Figure 4, bottom).

Preliminary results (not shown) indicate that in some cases learning the kernel on the classes may yield improved classification accuracy. Moreover, it seems to offer the possibility to gain understanding about the relationships between classes. However, exactly how this works does not yet seems to be clear, and further research in this direction is indicated.

4. Discussion

We extend multiclass classification to allow the use of multiple kernels. Using the conjugate of the loss function, we slightly generalize the proof technique used



pairwise distance matrix implied by the learned $\mathbf{K}_{\mathcal{Y}}$:

	x	o	+
x	0	1.8	2.2
o	1.8	0	1.0
+	2.2	1.0	0

Figure 4: Toy experiment for learning $k_{\mathcal{Y}}$.

in [20], and hence allow for all convex loss functions. Since many implementations of multiple kernel learning are based on SDP, SOCP, or QCQP approaches, we compare the direct approach to the approximation using a SILP, which results in significant computational savings. We then demonstrate that our approach is better than the state of the art in protein subcellular localization, an important problem in cell biology.

In contrast to a decomposition into binary classification algorithms, the proposed algorithm utilizes a common feature space for all the different classes. The advantages of this include easier visualization and feature extraction. Furthermore, it is possible that the number of kernels selected by our method may be lower than when running, e.g., one-vs-rest MKL. For example, there may be several related kernels any one of which can be used to differentiate between all the classes, but a different one may be selected in each binary subproblem. An interesting open question is whether there are significant gains in this direction. Disadvantages of having a common feature space arise when the user is particularly interested in which specific kernel (or even feature) identifies any class from the rest. However, further potential of our method may be in the possibility to learn a kernel on the classes; this remains to be investigated.

Apart from being a valuable tool for the vast number of real-world multiclass problems, the proposed multiclass MKL formulation can serve as a natural starting point for carrying MKL into the world of structured output learning. For example, the abundant setting of label sequence learning can be viewed as a multiclass problem with an exponential number of classes and is frequently solved by column generation methods. Extending our optimization problems in this way is an exciting direction for future research.

Acknowledgments

We thank Alex Smola for proposing to learn $\mathbf{K}_{\mathcal{Y}}$, Bill Noble for providing the yeast data, and Gunnar Rätsch for practical help with CPLEX. This work was supported in part by the IST Programme of the European Community, under the PASCAL Network of Excellence, IST-2002-506778.

Appendix: Deriving the Optimization Problem

We prove the dualization for the multiple kernel case. Recall that n denotes the number of data points, and there are $m := |\mathcal{Y}|$ classes and p kernels. The single kernel case falls out as special case with $p = 1$.

Proof [of Theorem 1] We equivalently replace the constraints on ξ_i in (4) by $t_{iu} = \sum_k \langle \mathbf{v}_k, \Psi_{kiu} \rangle + b_{y_i} - b_u$ and $\xi_i \geqslant \ell(t_{iu})$ for all i and $u \neq y_i$. The Lagrangian of the resulting OP is given by

$$\begin{aligned}\mathcal{L} = \; & \frac{1}{2}\sum_k \frac{1}{\beta_k}\|\mathbf{v}_k\|^2 + \sum_i \xi_i + \gamma\left(\sum_k \beta_k - 1\right) \\ & - \sum_k \epsilon_k \beta_k + \sum_i \sum_{u\neq y_i} \eta_{iu}\left(\ell\left(t_{iu}\right) - \xi_i\right) \\ & + \sum_i \sum_{u\neq y_i} \tilde{\alpha}_{iu}\left(t_{iu} - \sum_k \langle \mathbf{v}_k, \Psi_{kiu}\rangle - b_{y_i} + b_u\right),\end{aligned}$$

with Lagrange variables $\mathbf{0} \leqslant \boldsymbol{\epsilon} \in \mathbb{R}^p$, $\tilde{\boldsymbol{\alpha}} \in \mathbb{R}^{n\times(m-1)}$, $\mathbf{0} \leqslant \boldsymbol{\eta} \in \mathbb{R}^{n\times(m-1)}$, and $\gamma \in \mathbb{R}$. From $\frac{\partial \mathcal{L}}{\partial \mathbf{v}_k} = 0$ we prove (6), as we get $\frac{1}{\beta_k}\mathbf{v}_k = \sum_i \sum_{u\neq y_i} \tilde{\alpha}_{iu}\Psi_{kiu}$. We plug the results obtained from setting the partial derivatives wrt all primal variables to zero into the Lagrangian, thereby simplifying it considerably.

$$\begin{aligned}\max_{\boldsymbol{\eta},\tilde{\boldsymbol{\alpha}},\gamma} \min_{\mathbf{t}} \quad & \textstyle\sum_i \sum_{u\neq y_i} \left(\eta_{iu}\ell(t_{iu}) + \tilde{\alpha}_{iu} t_{iu}\right) - \gamma, \\ \text{s.t.} \quad & \forall k : \gamma \geqslant \frac{1}{2}\|\mathbf{w}_k\|^2, \text{ and } \forall i : \forall u : 0 \leqslant \eta_{iu}, \\ & \forall i : 1 = \sum_{u\neq y_i} \eta_{iu}, \\ & \forall v : 0 = \sum_i (1 - \delta_{y_i v})\tilde{\alpha}_{iv} - \sum_i \delta_{y_i v} \sum_{u\neq y_i} \tilde{\alpha}_{iu} \quad .\end{aligned}$$

Finally we transform the objective to obtain (5) by taking into account the effect of linear equality constraints on conjugate functions (cf. [4, Section 5.7]).

$$\begin{aligned} & \max_{\boldsymbol{\eta},\tilde{\boldsymbol{\alpha}},\gamma} \min_{\mathbf{t}} \sum_i \sum_{u\neq y_i} \left(\eta_{iu}\ell(t_{iu}) + \tilde{\alpha}_{iu} t_{iu}\right) - \gamma \\ & \Leftrightarrow \qquad (13) \\ & \min_{\boldsymbol{\eta},\tilde{\boldsymbol{\alpha}},\gamma} \gamma + \sum_i \sum_{u\neq y_i} \eta_{iu}\ell^*\left(-\frac{\tilde{\alpha}_{iu}}{\eta_{iu}}\right).\end{aligned}$$

■

Corollary 2 *When choosing the hinge loss, $\ell(t) := C\max(0, 1-t)$, the optimum $\mathbf{w}$ of (4) can be computed as in (10), where $\boldsymbol{\alpha} \in \mathbb{R}^{n\times\mathcal{Y}}$ is the solution of the QCQP (9).*

Proof We apply Theorem 1 with the observation that the conjugate function of the hinge loss $\ell(t) := C\max(0, 1-t)$ is given by

$$\ell^*(\nu) = \begin{cases} \nu & -C \leqslant \nu \leqslant 0 \\ \infty & \text{otherwise} \end{cases}.$$

Plugging this into (5) yields

$$\begin{aligned} \min_{\boldsymbol{\eta},\tilde{\boldsymbol{\alpha}},\gamma} \quad & \gamma - \sum_i \sum_{u\neq y_i} \tilde{\alpha}_{iu} \\ \text{s.t.} \quad & \forall k : \gamma \geqslant \frac{1}{2}\left(\sum_i \sum_{u\neq y_i} \tilde{\alpha}_{iu}\Psi_{kiu}\right)^2, \\ & \forall i : \forall u \neq y_i : 0 \leqslant \tilde{\alpha}_{iu} \leqslant \eta_{iu} C, \\ & \forall i : \forall u \neq y_i : 0 \leqslant \eta_{iu}, \quad \text{and} \quad \forall i : 1 = \sum_{u\neq y_i} \eta_{iu}, \\ & \forall v : 0 = \sum_i (1-\delta_{y_i v})\tilde{\alpha}_{iv} - \sum_i \delta_{y_i v} \sum_{u\neq y_i} \tilde{\alpha}_{iu} \end{aligned}$$

We observe that the constraints involving η_{iu} imply $\sum_{u\neq y_i} \tilde{\alpha}_{iu} \leqslant C$, and tidy up the notation by using the substitution (8). Plugging both into (6) and (4), we obtain the result. ■

References

[1] F. R. Bach, G. R. G. Lanckriet, and M. I. Jordan. Multiple Kernel Learning, Conic Duality, and the SMO Algorithm. In *Proceedings of the Twenty-first International Conference on Machine Learning*, 2004.

[2] K. M. Borgwardt, C. S. Ong, S. Schönauer, S. V. N. Vishwanathan, A. J. Smola, and H.-P. Kriegel. Protein function prediction via graph kernels. In *Proceedings of the International Conference on Intelligent Systems for Molecular Biology*, 2005.

[3] O. Bousquet and D. J. L. Herrmann. On the complexity of learning the kernel matrix. In *Neural Information Processing Systems*, 2003.

[4] S. P. Boyd and L. Vandenberghe. *Convex Optimization.* Cambridge University Press, 2004.

[5] O. Chapelle and A. Zien. Semi-Supervised Classification by Low Density Separation. In *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics*, pages 57–64, 2005.

[6] K. Crammer, J. Keshet, and Y. Singer. Kernel design using boosting. In *Neural Information Processing Systems*, 2003.

[7] O. Emanuelsson, H. Nielsen, S. Brunak, and G. von Heijne. Predicting subcellular localization of proteins based on their N-terminal amino acid sequence. *Journal of Molecular Biology*, 300:1005–1016, 2000.

[8] G. Fung, M. Dundar, J. Bi, and B. Rao. A fast iterative algorithm for fisher discriminant using heterogeneous kernels. In *Proceedings of the 21st International Conference on Machine Learning*, pages 40–47, 2004.

[9] J. L. Gardy, M. R. Laird, F. Chen, S. Rey, C. J. Walsh, M. Ester, and F. S. L. Brinkman. PSORTb v.2.0: expanded prediction of bacterial protein subcellular localization and insights gained from comparative proteome analysis. *Bioinfomatics*, 21:617–623, 2004.

[10] R. Hettich and K. O. Kortanek. Semi-Infinite Programming: Theory, Methods, and Applications. *SIAM Review*, 35(3):380–429, September 1993.

[11] A. Höglund, P. Dönnes, T. Blum, H.-W. Adolph, and O. Kohlbacher. MultiLoc: prediction of protein subcellular localization using N-terminal targeting sequences, sequence motifs, and amino acid composition. *Bioinfomatics*, 2006.

[12] S.-J. Kim, A. Magnani, and S. Boyd. Optimal kernel selection in kernel fisher discriminant analysis. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 465–472, 2006.

[13] G. Lanckriet, T. De Bie, N. Cristianini, M. I. Jordan, and W. Stafford Noble. A statistical framework for genomic data fusion. *Bioinfomatics*, 20(16):2626–2635, 2004.

[14] G. Lanckriet, N. Cristianini, P. Bartlett, L. El Ghaoui, and M. I. Jordan. Learning the kernel matrix with semi-definite programming. *Journal of Machine Learning Research*, 5:27–72, 2004.

[15] C. A. Micchelli and M. Pontil. Feature space perspectives for learning the kernel. *Machine Learning*, 66:297–319, 2007.

[16] C. S. Ong, A. J. Smola, and R. C. Williamson. Learning the kernel with hyperkernels. *Journal of Machine Learning Research*, 6:1043–1071, 2005.

[17] C. S. Ong and A. Zien. An automated combination of kernels for predicting protein subcellular localization. Manuscript in preparation, 2007.

[18] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning.* The MIT Press, 2006.

[19] B. Schölkopf and A. J. Smola. *Learning with Kernels.* MIT Press, Cambridge, MA, 2002.

[20] S. Sonnenburg, G. Rätsch, and C. Schäfer. A general and efficient multiple kernel learning algorithm. In *Neural Information Processings Systems*, 2005.

[21] I. Tsochantaridis, T. Hoffmann, T. Joachims, and Y. Altun. Support vector machine learning for interdependent and sturcutured output spaces. In *Proceedings of the 16th International Conference on Machine Learning*, 2004.