Nonstationary Kernel Combination

Darrin P. Lewis DPLEWIS@CS.COLUMBIA.EDU
Tony Jebara JEBARA@CS.COLUMBIA.EDU
Dept. of Computer Science, Columbia University, New York, NY 10027

William Stafford Noble NOBLE@GS.WASHINGTON.EDU
Dept. of Genome Sciences,
Dept. of Computer Science and Engineering, University of Washington, Seattle, WA 98195

Abstract

The power and popularity of kernel methods stem in part from their ability to handle diverse forms of structured inputs, including vectors, graphs and strings. Recently, several methods have been proposed for combining kernels from heterogeneous data sources. However, all of these methods produce stationary combinations; i.e., the relative weights of the various kernels do not vary among input examples. This article proposes a method for combining multiple kernels in a nonstationary fashion. The approach uses a large-margin latent-variable generative model within the maximum entropy discrimination (MED) framework. Latent parameter estimation is rendered tractable by variational bounds and an iterative optimization procedure. The classifier we use is a log-ratio of Gaussian mixtures, in which each component is implicitly mapped via a Mercer kernel function. We show that the support vector machine is a special case of this model. In this approach, discriminative parameter estimation is feasible via a fast sequential minimal optimization algorithm. Empirical results are presented on synthetic data, several benchmarks, and on a protein function annotation task.

1. Introduction

Although support vector machines (SVMs), kernel machines and other discriminative learning methods produce useful classifiers, they generally are not as flexible as generative models. For instance, Bayesian or generative models provide a rich framework for describing an input domain and exploring various structures and prior knowledge with respect to a given learning problem. Recently, several techniques have emerged to endow discriminative learning with useful properties found in generative models. These techniques handle probabilistic models, missing labels, feature selection, structured models, and so forth.

Consider how elegantly sequence learning is expressed using generative hidden Markov models. Only recently have large margin methods been extended to handle fully observed structured models such as trees and Markov networks (Jaakkola et al., 1999; Taskar et al., 2003; Jebara, 2004). Similarly, missing class labels, which are naturally handled via probabilistic expectation-maximization methods (Nigam et al., 2000), were difficult to migrate into an SVM setting (Joachims, 1999). Other extensions that are straightforward in a generative setting, such as feature selection, have only recently been accomplished in a large-margin setting (Weston et al., 2001).

Similarly, generative methods provide a principled means for performing inference on heterogeneous data sets, in which each example is represented by data drawn from different distributions and possibly having different forms—vectors, sets, strings, graphs, etc. Recently, several methods have been described for handling heterogeneous data sets by combining kernels in the context of SVM learning. These methods include using unweighted sums of kernels (Pavlidis et al., 2001), weighted sums using semidefinite programming (Lanckriet et al., 2002) and so-called “hyperkernels” (Ong et al., 2005; Sonnenburg et al., 2006). These kernel combination techniques have been applied primarily in computational biology (Pavlidis et al., 2001; Lanckriet et al., 2004; Ben-Hur & Noble, 2005; Borgwardt et al., 2005; Sonnenburg et al., 2006), where heterogeneous data sets are common.

This paper describes a technique for combining kernels

Appearing in *Proceedings of the 23rd International Conference on Machine Learning*, Pittsburgh, PA, 2006. Copyright 2006 by the author(s)/owner(s).

that is more general than a weighted linear summation. The technique uses a generative framework that specifically optimizes distributions to achieve maximal margins, effectively embedding latent variables into support vector machines. This permits the use of mixture modeling in a large-margin discriminative setting.

We demonstrate the proposed technique by mixing kernels in a nonstationary fashion, allowing kernel weights to be a function of the data. In order to illustrate the differences between standard SVMs, existing kernel combination methods, and our technique, consider the SVM discriminant function:

$$f(x) = \sum_t y_t \lambda_t k(x_t, x) + b.$$

Here, the class label assigned to example x by the trained SVM depends on the training examples x_t, the SVM weights λ_t, a bias term b, and a single kernel function $k(\cdot, \cdot)$. In the SDP and hyperkernel approaches, this formula is generalized to use multiple kernels, each of which has a fixed weight ν_m:

$$f(x) = \sum_t y_t \lambda_t \sum_m \nu_m k_m(x_t, x) + b$$

The approach that we propose generalizes this formulation further, allowing the kernel weight ν_m to depend upon the properties of the example being classified:

$$f(x) = \sum_t y_t \lambda_t \sum_m \nu_{m,t}(x) k_m(x_t, x) + b$$

This dependence on x is useful, for example, if the training data is structured such that some types of examples are best classified with one kernel and other types of examples are best classified using another.

In order to estimate the nonstationary kernel combination, we follow derivations similar to those in (Jebara, 2004). That work used a slightly different algorithm and only learned large margin mixtures of Gaussians. We extend this approach by allowing the Gaussians to have different kernels or mappings to Hilbert space.

In the following section, we describe the maximum entropy discrimination (MED) framework (Jaakkola et al., 1999), as well as our approach to latent variable modeling and nonstationary kernel combination (NSKC). We then describe an optimization method targeted to this particular problem, based on sequential minimal optimization (SMO) (Platt, 1999). Finally, we demonstrate the utility of our NSKC method in three sets of experiments: an illustrative synthetic example, three benchmark data sets, and a protein functional classification problem using three kernels.

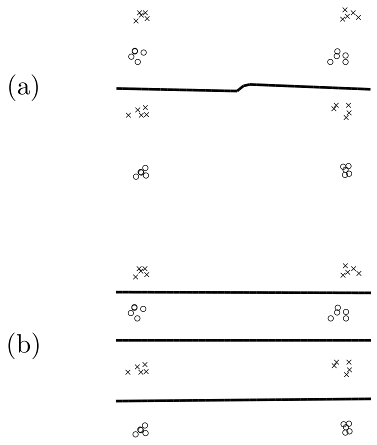

Figure 1. **Latent MED Gaussian mixture.** This figure shows the advantage of discriminative learning, even under extreme model mismatch. The classifiers are log-ratios of two Gaussian mixtures, each with two Gaussian components. Maximum likelihood (a) classifies poorly, achieving 50% accuracy due to poor discrimination. Latent MED (b) achieves perfect large-margin classification.

2. Latent Maximum Entropy Discrimination

Maximum entropy discrimination (MED) (Jaakkola et al., 1999) is a flexible generalization of support vector machines. MED produces a solution that is a distribution of parameter models $P(\Theta)$ rather than finding a single parameter setting Θ^*. We will use this approach to learn a classifier as the log-ratio of Hilbert space Gaussian mixtures. This discriminant function is simply

$$\mathcal{L}(X_t; \Theta) = \ln \frac{\sum_{m=1}^{M} \alpha_m \mathcal{N}(\phi_m^+(X_t) | \mu_m^+, I)}{\sum_{n=1}^{N} \beta_n \mathcal{N}(\phi_n^-(X_t) | \mu_n^-, I)} + b. \quad (1)$$

Given the solution distribution $P(\Theta)$, the MED classifier is $\hat{y} = \text{sign}\left(\int_\Theta P(\Theta)\mathcal{L}(X_t; \Theta) d\Theta\right)$. In Eq. (1), the model parameters Θ are the μ_m^+, μ_n^- Gaussian means, the α, β mixing proportions, and the scalar bias, b. We assume white Gaussian ($\mathcal{N}(0, I)$) prior distributions on the Gaussian means which regularize them towards zero. Non-informative priors are chosen for α, β, and b to reflect our lack of knowledge about these parameters and to simplify the resulting MED integrals. Note the use of explicit feature mappings ϕ_m^+, ϕ_n^- which permit each Gaussian to reside in a distinct feature space.

MED finds a distribution $P(\Theta)$ that minimizes divergence to a prior $P^{(0)}(\Theta)$ such that we satisfy the required classification constraints

$\int P(\Theta) y_t \mathcal{L}(X_t;\Theta) d\Theta \geq \gamma_t \ \forall t$ in the expected sense. The classical maximum entropy solution is $P(\Theta) = Z^{-1}(\lambda) P^{(0)}(\Theta) \exp(\sum_t \lambda_t [y_t \mathcal{L}(X_t;\Theta) - \gamma_t])$. Yet this solution is not fruitful here because, for mixtures, computing and minimizing the required partition function $Z(\lambda)$ (the normalizer of $P(\Theta)$) is intractable with exponentially many terms. To compensate, we use variational methods. Jensen's inequality is first applied in the classification constraints and tightens them. Then, Jensen's is applied to the partition function to yield a tractable projection. Starting with an initial solution $P^{(1)}(\Theta)$, we iterate the tractable projection step and convex hull restriction step until convergence, slowly updating the current MED solution $P^{(i)}(\Theta)$ and reducing divergence to the prior $P^{(0)}(\Theta)$, while moving within the MED admissible set. Details of the variational bounding and iterative update procedure are beyond the scope of this brief paper, but are available in (Jebara, 2004). However, here we use an improved bound over the incorrect class examples.

Invoking Jensen on the MED inequality constraints and then on the resulting partition function yields a *tractable* partition function we denote $\ddot{Z}(\lambda, Q|q)$ which uses the variational distributions Q and q required by the double Jensen bound. We can solve the optimization over λ and Q *simultaneously* as a quadratic program. We update the q distribution using an EM-like procedure to keep the MED inequality constraints tight. The iterative procedure—bounding the classification constraints with Jensen using a previous model $P^{(i-1)}(\Theta)$ and then solving the simpler MED problem—has been shown to converge to a local minimum as in the latent anomaly detection problem of (Jaakkola et al., 1999). We solve the various integrals for $\ddot{Z}(\lambda, Q|q)$ in closed form using the formulas in (Jebara, 2004). Ultimately, the negative logarithm of the partition function is our SVM-like objective function

$$
\begin{aligned}
\ddot{J}(\lambda, Q|q) \quad = \quad & \sum_{t\in\mathcal{T}} \lambda_t (H(Q_t) - H(q_t)) + \sum_{t\in\mathcal{T}} \lambda_t \gamma_t \\
& - \frac{1}{2} \sum_{t,t'\in\mathcal{T}^+} \lambda_t \lambda_{t'} \Big(\sum_m q_t(m) q_{t'}(m) k_m^+(t,t') \\
& \qquad + \sum_n Q_t(n) Q_{t'}(n) k_n^-(t,t') \Big) \\
& - \frac{1}{2} \sum_{t,t'\in\mathcal{T}^-} \lambda_t \lambda_{t'} \Big(\sum_m Q_t(m) Q_{t'}(m) k_m^+(t,t') \\
& \qquad + \sum_n q_t(n) q_{t'}(n) k_n^-(t,t') \Big) \\
& + \sum_{\substack{t\in\mathcal{T}^+ \\ t'\in\mathcal{T}^-}} \lambda_t \lambda_{t'} \Big(\sum_m q_t(m) Q_{t'}(m) k_m^+(t,t') \\
& \qquad + \sum_n Q_t(n) q_{t'}(n) k_n^-(t,t') \Big).
\end{aligned}
$$

In $\ddot{J}(\lambda, Q|q)$, we have written the objective in terms of kernel evaluations, $k_m^+(\cdot,\cdot)$ and $k_n^-(\cdot,\cdot)$, rather than inner products on our explicit feature mappings. Furthermore, a set of $M + N$ linear equality constraints emerge from barrier functions that are by products of the integration. These are enforced in addition to the positivity and upper bound constraints on λ. Note that these constraints subsume the traditional linear equality constraint of the SVM.

$$
\begin{aligned}
\sum_{t\in\mathcal{T}^-} \lambda_t Q_t(m) &= \sum_{t\in\mathcal{T}^+} \lambda_t q_t(m) \ \forall m = 1 \ldots M \\
\sum_{t\in\mathcal{T}^+} \lambda_t Q_t(n) &= \sum_{t\in\mathcal{T}^-} \lambda_t q_t(n) \ \forall n = 1 \ldots N \\
0 \leq \lambda_t &\leq c \ \forall t = 1 \ldots T
\end{aligned}
$$

We implemented the maximization of the above $\ddot{J}(\lambda, Q|q)$, subject to the above constraints, as a quadratic program, omitting the entropy terms $H(Q_t)$ due to their non-quadratic nature. This simplification is tolerable for the moment, because $H(Q_t)$ vanishes toward zero as Q becomes committed to a single mixture component, which often happens in practice.

The update for q ensures that bounds on classification constraints are tight under the previous iteration's $P^{(i-1)}(\Theta)$ solution:

$$
q_t(m) \propto \exp\left(E\{\ln \alpha_m\} + E\{\ln \mathcal{N}(\phi_m^+(X_t)|\mu_m^+)\} \right)
$$

and

$$
q_t(n) \propto \exp\left(E\{\ln \beta_n\} + E\{\ln \mathcal{N}(\phi_n^-(X_t)|\mu_n^-)\} \right),
$$

for $\forall t \in \mathcal{T}^+$ and $\forall t \in \mathcal{T}^-$, respectively.

Thus, for updating q and for making predictions with the final model, we need expectations under $P(\Theta)$ for various components of the discriminant function. For instance, the expected value of each positive-model log-Gaussian for $m = 1..M$ is

$$
\begin{aligned}
& E\{\ln \mathcal{N}(\phi_m^+(X_t)|\mu_m^+)\} = \\
& \qquad - \frac{D}{2} \ln(2\pi) - \frac{1}{2} - \frac{1}{2} k_m^+(X_t, X_t) \\
& \qquad + \sum_{\tau\in\mathcal{T}^+} \lambda_\tau q_\tau(m) k_m^+(X_\tau, X_t) \\
& \qquad - \sum_{\tau\in\mathcal{T}^-} \lambda_\tau Q_\tau(m) k_m^+(X_\tau, X_t) \\
& \quad - \frac{1}{2} \sum_{\tau,\tau'\in\mathcal{T}^+} \lambda_\tau \lambda_{\tau'} q_\tau(m) q_{\tau'}(m) k_m^+(X_\tau, X_{\tau'}) \\
& \quad - \frac{1}{2} \sum_{\tau,\tau'\in\mathcal{T}^-} \lambda_\tau \lambda_{\tau'} Q_\tau(m) Q_{\tau'}(m) k_m^+(X_\tau, X_{\tau'})
\end{aligned}
$$

$$+ \sum_{\substack{\tau\in\mathcal{T}^+ \\ \tau'\in\mathcal{T}^-}} \lambda_\tau \lambda_{\tau'} q_\tau(m) Q_{\tau'}(m) k_m^+(X_\tau, X_{\tau'})$$

We similarly compute the expected log Gaussian probability $E\{\ln \mathcal{N}(\phi_n^-(X_t)|\mu_n^-)\}$ for negative models. Finally, we obtain the expected bias and mixing proportions indirectly via the following surrogate variables:

$$\begin{aligned} a_m = E\{\ln \alpha_m\} + \frac{1}{2}E\{b\} \quad \forall m = 1..M \\ b_n = E\{\ln \beta_n\} - \frac{1}{2}E\{b\} \quad \forall n = 1..N \end{aligned}$$

This is done by using the KKT conditions, which ensure that at non-bound Lagrange multiplier settings, classification inequalities are achieved *with equality.* In other words, the stricter MED constraints implicit in $\ddot{Z}(\lambda, Q|q)$ become equalities. Thus, when $\lambda_t \in (0, c)$ we must achieve the following with equality:

$$\sum_m q_t(m)(a_m + E\{\ln \mathcal{N}(\phi_m^+(X_t)|\mu_m^+)\}) + H(q_t) =$$
$$\sum_n Q_t(n)(b_n + E\{\ln \mathcal{N}(\phi_n^-(X_t)|\mu_n^-)\}) + H(Q_t) + \gamma_t$$

for $t \in \mathcal{T}^+$ and the following equalities

$$\sum_n q_t(n)(b_n + E\{\ln \mathcal{N}(\phi_n^-(X_t)|\mu_n^-)\}) + H(q_t) =$$
$$\sum_m Q_t(m)(a_m + E\{\ln \mathcal{N}(\phi_m^+(X_t)|\mu_m^+)\}) + H(Q_t) + \gamma_t$$

for $t \in \mathcal{T}^-$. We solve for a_m for $m = 1..M$ and b_n for $n = 1..N$ in this (over-constrained) linear system, obtaining the expected bias and mixing proportions.

To make predictions with the latent MED's learned $P(\Theta)$, we cannot directly use the expectation prediction rule due to its intractable log-sums. Instead, we employ the following approximate prediction using the expectation formulas we just derived:

$$\hat{y} = \ln \frac{\sum_m \exp\left(E\{\ln \mathcal{N}(\phi_m^+(X)|\mu_m^+)\} + a_m\right)}{\sum_n \exp\left(E\{\ln \mathcal{N}(\phi_n^-(X)|\mu_n^-)\} + b_n\right)}. \qquad (2)$$

2.1. Nonstationary Kernel Prediction

We now show how the decision rule in Eq. (2) can be viewed as a nonstationary kernel combination. We rewrite the equations using the definition of the expectations for the log-Gaussians. First, recall Jensen's inequality, which holds for any non-negative q_m and ν_m scalars such that $\sum_m \nu_m = 1$:

$$\ln \sum_m q_m = \ln \sum_m \nu_m \frac{q_m}{\nu_m} \geq \sum_m \nu_m \ln \frac{q_m}{\nu_m}$$

If we set $\nu_m = \frac{q_m}{\sum_m q_m}$ and substitute into the right hand side above, the inequality is actually an equality:

$$\ln \sum_m q_m \geq \sum_m \frac{q_m}{\sum_m q_m} \ln \frac{q_m}{\frac{q_m}{\sum_m q_m}} = \ln \sum_m q_m$$

Given the above, we choose:

$$\begin{aligned} \nu_m^+(X) &= \frac{\exp(E\{\ln \mathcal{N}(\phi_m^+(X)|\mu_m^+)\} + a_m)}{\sum_m \exp(E\{\ln \mathcal{N}(\phi_m^+(X)|\mu_m^+)\} + a_m)} \\ \nu_n^-(X) &= \frac{\exp(E\{\ln \mathcal{N}(\phi_n^-(X)|\mu_n^-)\} + b_n)}{\sum_n \exp(E\{\ln \mathcal{N}(\phi_n^-(X)|\mu_n^-)\} + b_n)} \end{aligned}$$

The ratios above can be integrated into Eq. (2) as weights as follows:

$$\begin{aligned} \hat{y} = \sum_m \nu_m^+(X) E\{\ln \mathcal{N}(\phi_m^+(X)|\mu_m^+)\} \\ - \sum_n \nu_n^-(X) E\{\ln \mathcal{N}(\phi_n^-(X)|\mu_n^-)\} \end{aligned}$$

Inserting our formula for $E\{\ln \mathcal{N}(\phi_m^+(X)|\mu_m^+)\}$ and $E\{\ln \mathcal{N}(\phi_n^-(X)|\mu_n^-)\}$ yields the prediction rule:

$$\begin{aligned} \hat{y} = &\sum_{\tau\in\mathcal{T}^+} \sum_m \lambda_\tau Q_\tau(m) \nu_m^+(X) k_m^+(X_\tau, X) \\ &- \sum_{\tau\in\mathcal{T}^-} \sum_m \lambda_\tau Q_\tau(m) \nu_m^+(X) k_m^+(X_\tau, X) \\ &- \sum_{\tau\in\mathcal{T}^-} \sum_n \lambda_\tau Q_\tau(n) \nu_n^-(X) k_n^-(X_\tau, X) \\ &+ \sum_{\tau\in\mathcal{T}^+} \sum_n \lambda_\tau Q_\tau(n) \nu_n^-(X) k_n^-(X_\tau, X) \\ &+ \sum_m \nu_m^+(X) k_m^+(X, X) - \sum_n \nu_n^-(X) k_n^-(X, X) \\ &+ \mathit{constant}. \end{aligned}$$

3. Improved Optimization with SMO

The general form of the objective function for the MED discriminative projection step is $J(\lambda) = c^T\lambda + \frac{1}{2}\lambda^T \mathbf{H} \lambda$, where the Hessian matrix, $\mathbf{H}$, contains the quadratic coefficients from our objective $\ddot{J}(\lambda, Q|q)$ and the linear coefficients are in the vector c. The vector λ is the solution to the combined QP proposed in the previous section, i.e., $\lambda \leftarrow Q\lambda$. The constraints are of the form $\mathbf{A}\lambda = 0$. In general, we cannot simultaneously maintain the $M + N$ linear equality constraints by optimizing over only axis pairs, as in Platt's SMO for the SVM (Platt, 1999). However, some analysis reveals that we can reduce the number of axes to a maximum of $\max(M, N)$ and a minimum of two, depending on the situation.

Latent MED is more general than the SVM. We replicate variables to permit optimization over a latent distribution. This replication imparts regular structure to the constraints. Eq. (3) shows the form of the constraint matrix, $\mathbf{A}$, for a latent MED problem with four latent states, two for each binary class from which examples are drawn.

$$\begin{bmatrix} \dots & q_{u_1} & q_{u_1} & \dots & -1 & 0 & \dots & q_{w_1} & q_{w_1} & \dots \\ \dots & q_{u_2} & q_{u_2} & \dots & 0 & -1 & \dots & q_{w_2} & q_{w_2} & \dots \\ \dots & 1 & 0 & \dots & -q_{v_1} & -q_{v_1} & \dots & 1 & 0 & \dots \\ \dots & 0 & 1 & \dots & -q_{v_2} & -q_{v_2} & \dots & 0 & 1 & \dots \end{bmatrix} \tag{3}$$

The illustrated columns of $\mathbf{A}$ correspond to axes u_1, u_2, v_1, v_2, and w_1, w_2 associated with examples u, v, and w in the latent MED problem. The entries of the form q_{i_j} are given by the posterior probability distributions q_i over the M and N latent states of the generative models. (In this case, $M = N = 2$.) Note the replication of values within the constraint matrix. A corresponding replication is required in the dual variables, λ. The embedded $N \times N$ and $M \times M$ identity matrices effectively select a latent configuration of the *opposite* class and its associated dual variable. Note how the form of the constraints differs based upon the class of the chosen example. We will now explore the effect of this structure on the optimization problem.

3.1. Inter-class

Let us consider two examples, u and v from opposite classes. Without loss of generality, let u be positive and v be negative. Then we can maintain the constraints using the following equalities in vector form:

$$q_u\big(\hat{\lambda}_u^T 1\big) - \hat{\lambda}_v = q_u\big(\lambda_u^T 1\big) - \lambda_v$$
$$\hat{\lambda}_u - q_v\big(\hat{\lambda}_v^T 1\big) = \lambda_u - q_v\big(\lambda_v^T 1\big).$$

Then, we can write

$$\Delta\lambda_v = (\Delta\lambda_u^T 1)q_u$$
$$\Delta\lambda_u = (\Delta\lambda_v^T 1)q_v = (((\Delta\lambda_u^T 1)q_u)^T 1)q_v = (\Delta\lambda_u^T 1)q_v.$$

Thus, $(\Delta\lambda_u^T 1) = (\Delta\lambda_v^T 1)$, and $\Delta\lambda_u$ and $\Delta\lambda_v$ are coupled via their norms by a single scalar that we will call Δs. We now have $\Delta\lambda_v = \Delta s q_u$ and $\Delta\lambda_u = \Delta s q_v$. In other words, we can maintain the linear equality constraints by updating the solution with the scaled posterior distributions q_u and q_v. The change in the quadratic objective function for the axes u and v is

$$\begin{aligned} \Delta J_{uv}(\Delta\lambda) &= c_u^T \Delta\lambda_u + c_v^T \Delta\lambda_v \\ &+ \tfrac{1}{2}\Delta\lambda_u^T \mathbf{H}_{uu}\Delta\lambda_u + \Delta\lambda_u^T \mathbf{H}_{uv}\Delta\lambda_v + \tfrac{1}{2}\Delta\lambda_v^T \mathbf{H}_{vv}\Delta\lambda_v \\ &+ \sum\nolimits_{t \neq u,v} (\Delta\lambda_t^T \mathbf{H}_{tu}\Delta\lambda_u + \Delta\lambda_t^T \mathbf{H}_{tv}\Delta\lambda_v). \end{aligned}$$

All that remains is to express the change in the objective, $\Delta J_{uv}(\Delta\lambda)$ as a function of Δs by changing variables. The resulting one-dimensional quadratic objective function, $\Delta J_{uv}(\Delta s)$, can be analytically optimized by finding the root of the derivative under the box constraint.

3.2. Intra-class

Now, we examine the case in which axes u and w are chosen from the same class. Without loss of generality, we chose u and w members of the positive class as in Eq. (3). As in the inter-class case, we write the constraints in terms of our two variables, u and w, in vector form:

$$q_u\big(\hat{\lambda}_u^T 1\big) + q_w\big(\hat{\lambda}_w^T 1\big) = q_u\big(\lambda_u^T 1\big) + q_w\big(\lambda_w^T 1\big)$$
$$\hat{\lambda}_u + \hat{\lambda}_w = \lambda_u + \lambda_w$$

These constraints are simpler than in the previous case. In particular, when $q_u = q_w$, we obtain the further simplification:

$$\big(\hat{\lambda}_u^T 1\big) + \big(\hat{\lambda}_w^T 1\big) = \big(\lambda_u^T 1\big) + \big(\lambda_w^T 1\big) \tag{4a}$$
$$\hat{\lambda}_u + \hat{\lambda}_w = \lambda_u + \lambda_w. \tag{4b}$$

Note that (4a) is satisfied when (4b) is satisfied. Therefore, we need only enforce that the sum of the vectors is constant. It is important to observe that the intra-class case is comprised of two sub-cases distinguished by whether $q_u = q_w$ or not.

3.2.1. Intra-class Equal

Now, we continue with the derivation of the *intra-class equal* case. Referring to Eq. (4b), we see that the constraints can be satisfied, at each iteration, by choosing a single sub-axis k along which to optimize, so we restrict ourselves to u_k and w_k. In this case, the latent MED optimization decomposes directly into Platt's SMO. Working from (4b), we obtain $\hat{\lambda}_{u_k} + \hat{\lambda}_{w_k} = \lambda_{u_k} + \lambda_{w_k}$, which leads us to

$$\hat{\lambda}_{u_k} = \lambda_{u_k} + (\lambda_{w_k} - \hat{\lambda}_{w_k})$$
$$\hat{\lambda}_{w_k} = \lambda_{w_k} + (\lambda_{u_k} - \hat{\lambda}_{u_k}).$$

We also observe that $\Delta\lambda_{u_k} = -\Delta\lambda_{w_k}$. We again call this scalar quantity Δs. The derivation of update rules proceeds analogously with Platt's SMO.

3.2.2. Intra-class Unequal

Finally, we consider the *intra-class unequal* case. This situation is slightly complicated by the fact that the posterior distributions q_u and q_w are not equal, and

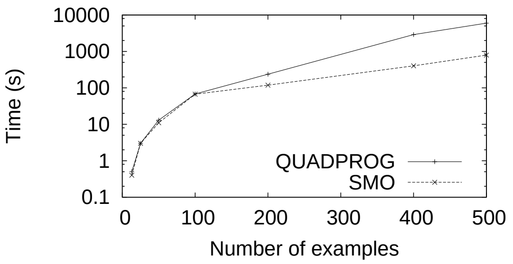

Figure 2. **Running time comparison of SMO and QUADPROG.** Running time as a function of the number of examples for our SMO implementation and Matlab's QUADPROG on the synthetic data set described in Section 4.1.

so no longer factor out and cancel as in (4a). This change requires us to explicitly enforce a constraint on the norm of the dual variable vectors, $\hat{\lambda}_u$ and $\hat{\lambda}_w$. In order to enforce the additional constraint, we must chose an additional sub-axis, l, over which to optimize.

Ultimately, we express the relationship between the four changing axes as a single scalar Δs and derive update rules, similarly to the other cases.

3.3. Newton Step

In addition to the SMO step types we described above, we found that it is necessary to occasionally interleave a second-order step over a larger set of axes. In working with the implementation, we discovered that SMO can get trapped in a local plateau in the objective function. Though the objective and constraints are convex, choosing a minimal set of axes to update results in slow convergence. The equality-constrained Newton step takes a more global view of the objective and progresses past some local problems. Due to the typically small working set, the efficiency of the optimization is not compromised.

3.4. Timing

To evaluate the efficiency of our optimization procedure, we ran a small timing experiment using the synthetic data set described in the next section. We computed the latent MED discriminative projection step for varying numbers of examples. We repeated this procedure several times to obtain mean optimization times for our SMO and for Matlab's QUADPROG. The results are summarized in Figure 2. With 500 examples, SMO is 7.5 times faster than QUADPROG.

4. Results

4.1. Synthetic Data Set

In order to illustrate the characteristics of nonstationary kernel combination, we first present a synthetic two-dimensional problem. We generate a binary labeled data set using a function that is quadratic in part of the input space and linear elsewhere. The function is $f(x) = x^2$ for $|x| < 1$, and $f(x) = 1$ otherwise. Points are translated along the vertical axis to create two classes for the binary problem. We then attempt to learn a decision surface using SDP kernel combination and the proposed nonstationary kernel combination.

The results are depicted in Figure 3. The decision surface of the SDP combination (b) is confounded between achieving large margin and high classification accuracy within the confines of a linear combination of kernels. The NSKC decision surface (c) achieves perfect large-margin classification, illustrating the merit and novelty of the technique.

4.2. Benchmark Data Sets

Next, we validate the nonstationary kernel combination technique on several benchmark data sets from the UCI machine learning repository. We use the Wisconsin breast cancer, sonar, and statlog heart data, all of which were previously used to validate the SDP method (Lanckriet et al., 2002). We use three kernels: a quadratic kernel $k_1(x_1, x_2) = (1 + x_1^T x_2)^2$, a radial basis function (RBF) kernel $k_2(x_1, x_2) = \exp(-0.5(x_1 - x_2)^T(x_1 - x_2)/\sigma)$, and a linear kernel $k_3(x_1, x_2) = x_1^T x_2$. All three kernels are normalized. As in (Lanckriet et al., 2002), we use a hard margin (with $c = 10,000$ to avoid numerical difficulties when $c = \infty$), and the RBF width parameter σ is set to 0.5, 0.1 and 0.5, respectively, for the three tasks. NSKC and maximum likelihood (ML) mixtures use one mixture component per kernel for each class model, i.e., $M = N = 3$. We report mean test set accuracy across five random replications of three-fold cross validation.

Table 1 summarizes the results of these experiments. NSKC achieves the top mean test set accuracy for two of three data sets. We believe that this performance is due, in part, to NSKC's ability to achieve a more regularized classifier by using the lower capacity polynomial and linear kernels for regions of the input space in which they perform well. By contrast, the SDP technique is forced to down-weight those entire kernels and use the RBF kernel with hard margin. The SDP classifier overfits in many cases.

The breast cancer data set is not separable using the

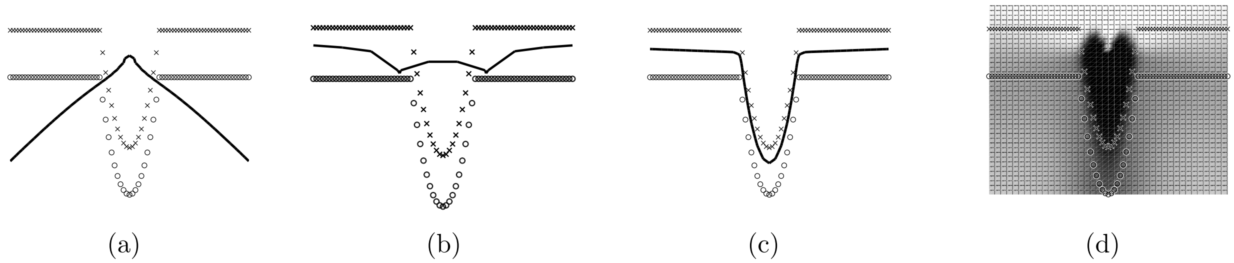

Figure 3. **Nonstationary kernel combination on synthetic data.** The figure illustrates the decision boundary between examples taken from a function that is piecewise linear and quadratic. Panel (a) shows the ML mixture decision boundary; panel (b) shows the SDP decision boundary; panel (c) shows NSKC decision boundary; panel (d) shows the NSKC kernel weight over the input space. NSKC is the only technique to separate the data. In the NSKC solution, note the smoothness of the linear and quadratic regions and the sharp kinks at their intersection.

Table 1. **Comparison of three benchmark data sets.** Accuracy for three UCI dataset under six classification methods. Mean and standard deviations of test set accuracy are shown across fifteen cross-validations (three-fold CV repeated five times). The first three methods are SVMs trained with single kernels, followed by the SDP approach of (Lanckriet et al., 2002), a maximum likelihood mixture of Gaussians classifier, and the NSKC method. In each row, the maximal mean value is indicated in boldface.

Dataset	quadratic	RBF	linear	SDP	ML	NSKC
Breast	0.549 ± 0.091	0.628 ± 0.019	0.543 ± 0.087	0.816 ± 0.015	0.557 ± 0.03	**0.831 ± 0.014**
Sonar	0.814 ± 0.01	0.860 ± 0.009	0.730 ± 0.01	0.860 ± 0.009	0.682 ± 0.022	**0.863 ± 0.008**
Heart	**0.614 ± 0.032**	0.556 ± 0.01	0.524 ± 0.02	0.556 ± 0.01	0.536 ± 0.024	0.605 ± 0.016

polynomial or linear kernels. Nonetheless, these kernels are useful to the combination techniques. Indeed, both the SDP and NSKC methods gain significant advantage by combining kernels.

The sonar data set is the only data set that is separable using each of the three kernels individually. The SDP technique matches the accuracy of the best individual kernel, the RBF. NSKC achieves a small additional gain from nonstationarity.

For the heart data set, the best accuracy is achieved using the polynomial of degree two, with which the data set is not separable. The data is separable using the kernel combination techniques. However, neither SDP nor NSKC make full use of the polynomial kernel. NSKC does seem to apply the polynomial over at least part of the input space.

4.3. Yeast Protein Functional Classification

In a larger experiment, we applied our nonstationary kernel combination to the problem of protein function annotation from a collection of heterogeneous kernels. We compare the latent MED method against SVMs using single kernels and the SDP method in (Lanckriet et al., 2004) using data from that study (`noble.gs.washington.edu/proj/yeast`). We use three kernels, representing gene expression, protein domain content, and protein sequence similarity. We train one-versus-all classifiers for 12 functional classes of yeast genes. We randomly sample from the data set to reduce its size to 500 genes and then perform three-fold cross-validation, repeating the entire procedure five times. For all methods, we use the regularization parameter $c = 10$, as in (Lanckriet et al., 2004). With NSKC, we (as usual) use one component per kernel per class.

Table 2 summarizes the mean AUCs over 15 trials for all methods. NSKC has the highest accuracy for eight of the twelve classes. We speculate NSKC's advantage over SDP in these experiments stems from its ability to capture some of the latent structure in the data. In particular, the 12 yeast functional classes represent the top level of a MIPS functional hierarchy. Thus, these 12 classes contain meaningful substructure and subclasses which NSKC can exploit.

5. Discussion

In this paper we have presented a nonstationary method for combining kernels, based on latent variable modeling. The technique is motivated by the

Table 2. **Comparison of yeast protein function annotation methods.** The table lists, for each functional class (row) and each classification method (column) the mean AUC from five times three-fold cross-validation. The first three columns correspond to SVMs trained on single kernels (gene expression, protein domain content and sequence similarity, respectively). The final two columns contain results for the SDP and nonstationary kernel combination methods. For all methods, standard errors (not shown) are generally on the order of 0.02, except for classes 2 (0.04) and 9 (0.05).

Class	Exp	Dom	Seq	SDP	NSKC
1	0.630	0.717	**0.750**	0.745	0.747
2	0.657	0.664	0.718	0.751	**0.755**
3	0.668	0.706	0.729	0.768	**0.774**
4	0.596	0.756	0.752	0.766	**0.778**
5	0.810	0.773	0.789	0.834	**0.836**
6	0.617	0.690	0.668	0.698	**0.717**
7	0.554	0.715	**0.740**	0.720	0.738
8	0.594	0.636	0.680	0.697	**0.699**
9	0.535	0.564	**0.603**	0.582	0.576
10	0.554	0.616	**0.706**	0.697	0.687
11	0.506	0.470	0.480	0.524	**0.526**
12	0.682	0.896	0.883	0.916	**0.918**

natural intuition that it may be useful for the kernel combination weights to be a function of the input. Our nonstationary kernel combination technique leverages the power and flexibility of generative probabilistic models, while providing the discriminative accuracy of large margin parameter estimation.

We describe a large-margin discriminative optimization for the latent variable model, carried out within the maximum entropy discrimination framework. We present a novel formulation in which each Gaussian component can be implicitly mapped to a distinct feature space, via a Mercer kernel. We derive an efficient implementation of the optimization, based on the sequential minimal optimization algorithm and we compare the timing to a popular quadratic program solver.

To validate the technique, we present empirical results on synthetic data, several benchmarks, and on a protein function annotation task. In general, our nonstationary kernel combination technique shows an advantage over existing techniques.

Acknowledgements

This project was supported by National Science Foundation grants CCR-0312690, IIS-0347499 and IIS-0093302 and National Institute of Health grant R33 HG003070.

References

Ben-Hur, A., & Noble, W. S. (2005). Kernel methods for predicting protein-protein interactions. *Bioinformatics*, *21 suppl 1*, i38–i46.

Borgwardt, K., Ong, C. S., Schoenauer, S., Vishwanathan, S., Smola, A., & Kriegel, H.-P. (2005). Protein function prediction via graph kernels. *Bioinformatics*, *21*, i47–i56.

Jaakkola, T., Meila, M., & Jebara, T. (1999). Maximum entropy discrimination. *Advances in Neural Information Processing Systems.*

Jebara, T. (2004). *Machine learning: Discriminative and generative.* Boston, MA: Kluwer Academic.

Joachims, T. (1999). Transductive inference for text classification using support vector machines. *Proceedings of the Sixteenth International Conference on Machine Learning* (pp. 200–209). San Francisco, CA: Morgan Kaufmann.

Lanckriet, G. R. G., Cristianini, N., Bartlett, P., El Ghaoui, L., & Jordan, M. I. (2002). Learning the kernel matrix with semi-definite programming. *Proceedings of the 19th International Conference on Machine Learning.* Sydney, Australia: Morgan Kauffman.

Lanckriet, G. R. G., Deng, M., Cristianini, N., Jordan, M. I., & Noble, W. S. (2004). Kernel-based data fusion and its application to protein function prediction in yeast. *Proceedings of the Pacific Symposium on Biocomputing* (pp. 300–311). World Scientific.

Nigam, K., McCallum, A. K., Thrun, S., & Mitchell, T. M. (2000). Text classification from labeled and unlabeled documents using EM. *Machine Learning*, *39*, 103–134.

Ong, C. S., Smola, A. J., & Williamson, R. C. (2005). Learning the kernel with hyperkernels. *Journal of Machine Learning Research*, *6*, 1043–1071.

Pavlidis, P., Weston, J., Cai, J., & Grundy, W. N. (2001). Gene functional classification from heterogeneous data. *Proceedings of the Fifth Annual International Conference on Computational Molecular Biology* (pp. 242–248).

Platt, J. C. (1999). Fast training of support vector machines using sequential minimal optimization. In B. Schölkopf, C. J. C. Burges and A. J. Smola (Eds.), *Advances in kernel methods.* MIT Press.

Sonnenburg, S., Rätsch, G., & Schafer, C. (2006). A general and efficient multiple kernel learning algorithm. *Advances in Neural Information Processing Systems.*

Taskar, B., Guestrin, C., & Koller, D. (2003). Max margin markov networks. *Advances in Neural Information Processing Systems.* Cambridge, MA: MIT Press.

Weston, J., Mukherjee, S., Chapelle, O., Pontil, M., Poggio, T., & Vapnik, V. (2001). Feature selection for SVMs. *Advances in Neural Information Processing Systems 13.* Cambridge, MA: MIT Press.