Semi-supervised Mixture of Kernels via LPBoost Methods

Jinbo Bi Glenn Fung Murat Dundar Bharat Rao
Computer Aided Diagnosis and Therapy Solutions
Siemens Medical Solutions, Malvern, PA 19355
jinbo.bi, glenn.fung, murat.dundar, bharat.rao@siemens.com

Abstract

We propose an algorithm to construct classification models with a mixture of kernels from labeled and unlabeled data. Unlike traditional kernel methods which select a kernel according to cross validation performance, we derive classifiers that are a mixture of models, each based on one kernel choice from a library of kernels. The sparse-favoring 1-norm regularization method is employed to restrict the complexity of mixture models and to achieve the sparsity of solutions. By modifying the column generation boosting algorithm LPBoost to a more general linear programming formulation, we are able to efficiently solve mixture-of-kernel problems and automatically select kernel basis functions centered at labeled data as well as unlabeled data. The effectiveness of the proposed approach is proved by experimental results on benchmark datasets and a real-world lung nodule detection system.

1 Introduction

Recent years have seen considerable interests in learning with labeled and unlabeled data since experiments for labeling data are often expensive while vast amount of unlabeled data are easily available. Semi-supervised learning is commonly used to explore the information of input distribution provided by unlabeled data to improve the performance of supervised learning (see a recent survey [22]). In this paper, we propose a boosting algorithm to construct classification models from labeled and unlabeled data based on kernel methods.

Boosting algorithms are well-studied learning methodologies developed in the last decade. They construct classifiers in an incremental fashion by using a weighted "vote" of various simple models obtained from a weak learner. A general setting of boosting algorithms requires three components: a hypothesis set, a weak learner that produces simple models from the hypothesis set, a mechanism that combines the simple model into the final model by minimizing certain cost functional. Successful boosting methods, such as AdaBoost [11] and LPBoost [10], have established applications in many practical domains.

Kernel-based methods have proven to be very effective for solving inference problems in many applications. By introducing a positive semidefinite kernel K, nonlinear models can be created using linear learning algorithms such as in support vector machines (SVM), kernel ridge regression, kernel logistic regression, etc. The model is often written in the form of kernel expansions. For example, the decision boundary f obtained by SVM classification is expressed as

$$f(\mathbf{x}) = \sum_j \alpha_j K(\mathbf{x}, \mathbf{x}_j), \tag{1}$$

where α is the model coefficients and $\mathbf{x}_j$ is the j^{th} training input vector. as a type of basis expansion model with kernel terms $K(\mathbf{x}, \mathbf{x}_j)$ as basis functions centered at $\mathbf{x}_j$. In such kernel methods, the choice of kernel mapping is of crucial importance. Usually, the choice of kernel is determined by predefining the type of kernel (e.g, RBF, and polynomial kernels), and tuning the kernel parameters using cross-validation performance. Cross-validation is expensive and the resulting kernel is not guaranteed to be an excellent choice.

Recent work [14, 8, 17, 9] has attempted to design kernels that adapt to the data of a particular task to be solved. For example, Lanckriet et al. [14] and Bousquet et al. [8] proposed the use of a linear combination of kernels $K = \sum_p \mu_p K_p$ from a family of various kernel functions K_p. To ensure the positive semidefiniteness, the combination coefficients μ_p are either simply required to be nonnegative or determined in the way such that the composite kernel is positive semidefinite, for instance, by solving a semi-definite program as in [14] or by boosting as in [9]. The decision boundary f thus becomes

$$f(\mathbf{x}) = \sum_j \alpha_j \left(\sum_p \mu_p K_p(\mathbf{x}, \mathbf{x}_j) \right). \tag{2}$$

In this paper, we devise algorithms that boost on the columns of kernel matrices (the basis functions) constructed

from labeled and unlabeled data via column generation (CG) techniques. Our goal is to develop effective methods to construct inference models that make use of various geometry of the feature spaces introduced by a family of kernels other than a less expressive feature space induced by a single kernel. The proposed approach is designed for tasks where only a limited amount of labeled data is available whereas a large set of unlabelled data can be easily accessed. The unlabeled data is used jointly with the labeled data as possible centers for the kernel basis functions. From the boosting point of view, this corresponds to incorporating the basis functions constructed by unlabeled data into the hypothesis space $\mathcal{H}$, which tend to produce better predictive models, specially when very few labels can be obtained. The 1-norm regularization method is employed to restrict the capacity of mixture models and to achieve sparsity. We modify LPBoost, a CG-based boosting algorithm [10] to a more general linear programming formulation. The LPBoost modification can thus be used to efficiently optimize the proposed semi-supervised mixture-of-kernel models and further enhance the sparsity. The proposed algorithm becomes more general and is suitable for constructing a much larger variety of inference models.

We outline this paper as follows. In Section 2, we discuss the mixture-of-kernels model and describe its characteristics. Section 3 introduces the semi-supervised learning problem for inference with mixture of kernels. Section 4 extends the existing LPBoost algorithm to a general linear program, and adjusts the modified LPBoost to learn mixture-of-kernel models from labeled and unlabeled data. Experimental results on benchmark problems and a real-world nodule detection system are included in Section 5 to demonstrate the performance of the proposed approach. The last section 6 concludes this paper. Throughout this article, vectors are presumed to be column vectors unless otherwise stated, and denoted using bold-face lower letters. Matrices are denoted by bold-face upper letters.

2 Mixture of Kernels

Other than constructing new kernels or kernel matrices (the Gram matrix induced by a kernel on the training data) as in the model (2), our approach constructs models that are a mixture of models, each based on one kernel choice from a library of kernels. Our algorithm automatically determines the kernel basis functions to be used in the mixture model. Given a library of kernels $\mathcal{S} = \{K_p,\ p = 1, \cdots, P\}$, a Gram matrix $\mathbf{K}^p$ can be calculated for each kernel in $\mathcal{S}$ on sample data with the column $K_p(\cdot, \mathbf{x}_j)$ corresponding to the j^{th} example. The decision boundary represented by the mixture-of-kernel approach is

$$f(\mathbf{x}) = \sum_p \sum_j \alpha_j^p K_p(\mathbf{x}, \mathbf{x}_j). \qquad (3)$$

where α_j^p gives the weight for the j-th column in the p-th kernel matrix. Similar strategies have appeared in some works such as [3, 19] to improve generalization and reduce training and prediction computational costs. MARK [3] optimized a heterogeneous kernel using a gradient descent algorithm in function space. GSVC and POKER [19, 18] grew mixtures of kernel functions from a family of RBF kernels, and the mixture model was designed to be used only with weighted least squares VMS. The proposed approach can be used in conjunction with any linear generalized SVM formulations and algorithms. Moreover, our approach has been extended to solve many quadratic learning formulations with any suitable kernel families [5].

A mixture model of form (3) can potentially give a larger hypothesis space than a model that uses either a single kernel or a composite kernel. Models (3) based on a mixture of kernels are not necessarily equivalent to models (2) based on a composite kernel. Rewriting a composite kernel model (2) as $f(\mathbf{x}) = \sum_p \sum_i \alpha_i \mu_p K_p(\mathbf{x}_i, \mathbf{x})$, we can see it is equivalent to a mixture-of-kernel model (3) with $\alpha_i^p = \alpha_i \mu_p$. The opposite is not necessarily true, i.e., a mixture model is not necessarily equivalent to a composite model since given any composite kernel model (2), for any two kernels K_p and K_q, the ratio α_i^p / α_i^q is fixed to μ_p / μ_q, for all i (assuming $\mu_q \neq 0$ without loss of generality). Note that for a mixture-of-kernel model (3), we do not have this restriction.

Models of form (3) have potential to achieve better approximation capability since they can generate sparse solutions if appropriate regularization conditions are used. In the composite model, a kernel matrix is formed with an optimized μ vector, and then used to construct a kernel expansion as the classification model. For any $\alpha_j > 0$ in the constructed model, the corresponding columns $\mathbf{K}_j^p$ from all kernels specified by μ in the library have to be calculated although some columns or basis functions may not be necssary. In case the solution $\boldsymbol{\alpha}$ is not very sparse (which is true for most quadratic cost objectives), the execution time and computational cost in the testing phase will be drastic. For the mixture-of-kernel model, only necessary kernel basis functions will be selected and included in the classification model.

We discuss the relationship among different basis expansion models, such as models obtained by classic SVMs, mixture-of-kernel approaches and RBF nets [7]. In basis expansion methods [12], such as RBF networks, the model takes a form of $f(\mathbf{x}) = \sum \alpha_j \phi_j(\mathbf{x})$, where each function ϕ_j is a basis function of a form $\exp(-||\mathbf{x} - \mathbf{c}^j||^2 / \sigma_j)$. The centers $\mathbf{c}$ and the bandwidths σ of the basis functions in RBF nets are heuristically determined using unsupervised techniques. Therefore to construct the decision rule, one has to estimate: 1. the number of RBF centers; 2. the estimates of the centers; 3. the linear model parameter $\boldsymbol{\alpha}$,

and 4. the bandwidth parameter σ. Compared with RBF networks, the benefits of using SVMs have been studied in [21, 20]. The first three sets of parameters can be automatically determined by SVMs when learning support vectors. The last parameter is usually obtained by cross-validation tuning. Classic SVMs, however, use only a single parameter σ, which means that all centers (support vectors) are associated with a single choice of σ. Contrary to SVMs, RBF networks estimate a different σ for every center $\mathbf{c}$ of the RBF basis. More generally, the bandwidth σ can be different on different directions. The mixture-of-kernel model has a flexibility in between SVMs and RBF networks. Construction of a mixture-of-kernel model still benefits from the SVM-like algorithms, i.e., parameters except σ can be learned by solving an optimization problem. In addition, the mixture-of-kernel model allows the RBF basis positioned at different centers (support vectors) to associate with different bandwidths.

The mixture-of-kernel approach investigated here has interesting properties concerning its learning and approximation behaviors. These theoretical issues will be addressed elsewhere and more complete discussions on algorithmic design for constructing mixture-of-kernel models are presented in [5]. This paper focuses on the development of an algorithm for learning a mixture-of-kernel model that benefits from unlabeled data.

3 Semi-supervised Mixture of Kernels

We consider the problem of learning models with a mixture of kernels from labeled and unlabeled data. In reality, it is often comparatively easier and cheaper to obtain data with no labels . A learning algorithm may take advantage of unlabelled data to enhance the performance of the inference model to be derived. The problem that we deal with in this article is slightly different from traditional transductive learning [21] where the task is to predict the labels of the unlabeled examples given a training set of labeled examples and a working set of unlabeled examples. No inference model needs to be explicitly derived. In our problem, an inference model is required to predict labels of the test examples which are unseen in the training process. The validation of the algorithm performance is conducted only on an independent test set.

In a semi-supervised setting, besides a set of labeled data, $L = \{(\mathbf{x}_i, y_i),\ i = 1, \cdots, \ell\}$, a set of unlabeled data $U = \{\mathbf{x}_{\ell+j},\ j = 1, \cdots, \ell_u\}$ is also given. The kernel matrix calculated on both labeled and unlabeled data is

$$\mathbf{K}^p = \begin{pmatrix} \mathbf{K}^p_{L,L} & \mathbf{K}^p_{L,U} \\ \mathbf{K}^{pT}_{L,U} & \mathbf{K}^p_{U,U} \end{pmatrix} \tag{4}$$

where $\mathbf{K}^p_{i,j} = K_p(\mathbf{x}_i, \mathbf{x}_j),\ i, j = 1, \cdots, \ell, \ell+1, \cdots, \ell+\ell_u$. The classification model based on a mixture of these kernel matrices takes form of

$$f(\mathbf{x}) = \sum_p \sum_{j=1}^{\ell+\ell_u} \alpha^p_j K_p(\mathbf{x}, \mathbf{x}_j), \tag{5}$$

and the classifier is defined by $\text{sgn}(f(\mathbf{x}))$. Parallel to connection analysis with RBF nets in Section 2, models of form (5) allows basis functions to locate at examples with no labels. The estimates of the centers or support vectors can still be automatically optimized by a SVM-like algorithm. Furthermore, a lot more choices for basis centers come for free from unlabelled examples without any need of unsupervised techniques.

Similar to what has been used in SVMs, we optimize models by minimizing the margin-based 1-norm error metric $\sum_{i=1}^{\ell} \xi_i$ where ξ_i satisfies $y_i f(\mathbf{x}_i) \geq 1 - \xi_i, \forall i = 1, \cdots, \ell$. Without loss of generality, we include an explicit offset b in the model $f(\mathbf{x})$. Note that the offset can be incorporated into the kernel matrix. We write out b explicitly since we do not require the regularization to be taken on b. To achieve good generalization performance, it is important to apply appropriate regularization conditions to the model class. A commonly used regularization condition by single-kernel methods is the reproducing kernel Hilbert space (RKHS) regularization $R(f)$.

$$R(f) = \left\| \sum_i \alpha_i K(\mathbf{x}, \mathbf{x}_i) \right\|^2_{\mathcal{H}} = \boldsymbol{\alpha}^T \mathbf{K} \boldsymbol{\alpha}. \tag{6}$$

where $\mathcal{H}$ is the RKHS induced by the kernel K. The natural extension of the RKHS regularization condition to the semi-supervised mixture-of-kernel model is the following $R(f)$

$$R(f) = \sum_p \boldsymbol{\alpha}^{pT} \mathbf{K}^p \boldsymbol{\alpha}^p. \tag{7}$$

The objective of the learning problem with RKHS regularization is to minimize $\sum_p \boldsymbol{\alpha}^{pT} \mathbf{K}^p \boldsymbol{\alpha}^p + C \sum_{i=1}^{\ell} \xi_i$ where the error term is defined only using the top ℓ rows of the kernel matrix $\mathbf{K}$ as illustrated in (4), and the regularization term uses the entire kernel matrix $\mathbf{K}$ including the part produced by unlabeled data.

The RKHS regularization condition requires positive semi-definiteness (PSD) of each of the kernel matrices K_p, and solving the resulting optimization problem can be computationally expensive. To remove the SDP requirement and achieve computational efficiency, we can apply other regularization conditions, such as penalizing the 1-norm or 2-norm of $\boldsymbol{\alpha}$, that are equally suitable for capacity control [15, 4]. These regularization methods are more generally applicable since they do not require the kernel matrix to be positive semi-definite. In particular, we are in favor of the 1-norm regularization $||\boldsymbol{\alpha}||_1 = \sum |\alpha_j|$ since it is well

known that the 1-norm regularization leads to sparse solutions, which in the mixture-of-kernels model, is very desirable. Furthermore, we prove that the RKHS norm can be bounded using the 1-norm $\sum_i |\alpha_i^p|$.

$$\begin{aligned}
&\|\textstyle\sum_i \alpha_i^p K_p(\mathbf{x}, \mathbf{x}_i)\|_{\mathcal{H}_p}^2 = \sum_i \alpha_i^p \sum_j \alpha_j^p K_p(\mathbf{x}_i, \mathbf{x}_j) \\
&\leq (\textstyle\sum_i |\alpha_i^p|) \sup_{i,j} \left|\sum_j \alpha_j^p K_p(\mathbf{x}_i, \mathbf{x}_j)\right| \\
&\leq \textstyle\sum_i |\alpha_i^p| \left(\sup_i K_p(\mathbf{x}_i, \mathbf{x}_i)\right)^{1/2} \left\|\sum_i \alpha_i^p K_p(\mathbf{x}_i, \mathbf{x})\right\|_{\mathcal{H}_p}.
\end{aligned}$$

Thus,

$$\left\|\sum_i \alpha_i^p K_p(\mathbf{x}, \mathbf{x}_i)\right\|_{\mathcal{H}_p} \leq \sum_i |\alpha_i^p| \left(\sup_i K_p(\mathbf{x}_i, \mathbf{x}_i)\right)^{1/2}.$$

Consequently, the 1-norm regularization is less restrictive than RKHS regularization, and hence gives a more expressive model.

For notational convenience, let us line up all kernel matrices from $\mathcal{S}$ together $\mathbf{K} = [\mathbf{K}^1\ \mathbf{K}^2\ \cdots\ \mathbf{K}^P]$, and re-index the columns in $\mathbf{K}$. Let index j run through the columns and index i run along the rows. Hence $\mathbf{K}_{i\cdot}$ denotes the i^{th} row of $\mathbf{K}$, and $\mathbf{K}_{\cdot j}$ denotes the j^{th} column. There are $d = (\ell + \ell_u) \times p$ columns in total. We thus formulate the learning problem as

$$\begin{array}{ll}
\min_{\boldsymbol{\alpha},\boldsymbol{\xi}} & \sum_{j=1}^{d} |\alpha_j| + C \sum_{i=1}^{\ell} \xi_i \\
\text{s.t.} & y_i \left(\sum_j \mathbf{K}_{ij} \alpha_j + b\right) + \xi_i \geq 1, \\
& \xi_i \geq 0, \ \ i = 1, \cdots, \ell,
\end{array} \tag{8}$$

where the j^{th} column now only consists of the first ℓ elements of that in $\mathbf{K}$. There exist much more columns (variables $\boldsymbol{\alpha}$) in problem (8) than the number of constraints. In other words, compared with inductive learning, this formulation has significantly more basis functions to choose considering the size of labeled data available, it may require extra steps for capacity control. If an extra validation set is available, the optimization of (8) can be terminated when the performance on the validation set cannot be improved further. In other words, our algorithm will stop when adding a new kernel basis function does not improve performance on the validation set. This is similar to the early stopping technique often used in neural network training.

4 Generalized LPBoost

The CG techniques have been widely used for solving large-scale linear programs (LPs) or difficult integer programs since 1950s [16]. In the primal space, the CG method solves LPs on a subset of variables $\boldsymbol{\alpha}$, which means not all columns of the kernel matrix are generated at once and used to construct the function f. Columns are generated iteratively and added to the problem to achieve optimality. In the dual space, a column in the primal problem corresponds to a constraint in the dual problem. When a column is not included in the primal, the corresponding constraint does not appear in the dual. If a constraint absent from the dual problem is violated by the solution to the restricted problem, this constraint (a cutting plane) needs to be included in the dual problem to further restrict its feasible region. Thus these techniques are also referred to as cutting plane methods [1]. We first briefly review the existing LPBoost with 1-norm regularization. Then we propose our modification that allows us to consider kernels that do not necessarily comply with the PSD requirement.

If the hypothesis $\mathbf{K}_{\cdot j}\alpha_j$ based on a single column of the matrix $\mathbf{K}$ is regarded as a weak model or base classifier, we can rewrite LPBoost using our notation and following the statement in [10]:

$$\begin{array}{ll}
\min_{\boldsymbol{\alpha},\boldsymbol{\xi}} & \sum_{j=1}^{d} \alpha_j + C \sum_{i=1}^{\ell} \xi_i \\
\text{s.t.} & y_i \sum_j \mathbf{K}_{ij} \alpha_j + \xi_i \geq 1, \ \ \xi_i \geq 0, \ \ i = 1, \cdots, \ell \\
& \alpha_j \geq 0, \ \ j = 1, \cdots, d,
\end{array} \tag{9}$$

where $C > 0$ is the regularization parameter. The dual of LP (9) is

$$\begin{array}{ll}
\max_{\boldsymbol{\beta}} & \sum_{i=1}^{\ell} \beta_i \\
\text{s.t.} & \sum_{i=1}^{\ell} \beta_i y_i \mathbf{K}_{ij} \leq 1, \ \ j = 1, \cdots, d, \\
& 0 \leq \beta_i \leq C, \ \ i = 1, \cdots, \ell.
\end{array} \tag{10}$$

These problems are referred to as the master problems. The CG method solves LPs by incrementally selecting a subset of columns from the simplex tableau and optimizing the tableau restricted on the subset of variables (each corresponding to a selected column). After a primal-dual solution $(\hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\xi}}, \hat{\boldsymbol{\beta}})$ to the restricted problem is obtained, we solve

$$\tau = \max_j \sum_i \hat{\beta}_i y_i \mathbf{K}_{ij}, \tag{11}$$

where j runs over all columns of $\mathbf{K}$. If $\tau \leq 1$, the solution for the restricted problem is optimal to the master problems. If $\tau > 1$, then the solution to (11) provides a column to be included in the restricted problem.

As illustrated in problem (8), kernel methods in general do not require the model coefficients $\alpha_i = \hat{\alpha}_i y_i$ to be non-negative. Moreover, an explicit offset term b can be included in the function f. To form a LP from problem (8), we rewrite $\alpha_j = u_j - v_j$ where $u_j,\ v_j \geq 0$. Then $|\alpha_j| = u_j + v_j$ if either u_j or v_j has to be 0. The LP is then

formulated in variables $\mathbf{u}$, $\mathbf{v}$, b, $\boldsymbol{\xi}$ as

$$\begin{aligned} \min_{\mathbf{u},\mathbf{v},\boldsymbol{\xi}} \quad & \sum_{j=1}^{d}(u_j+v_j) + C\sum_{i=1}^{\ell}\xi_i \\ \text{s.t.} \quad & y_i\left(\sum_j \mathbf{K}_{ij}(u_j-v_j)+b\right)+\xi_i \geq 1, \\ & \xi_i \geq 0,\ \ i=1,\cdots,\ell, \\ & u_j,\ v_j \geq 0,\ \ j=1,\cdots,d. \end{aligned} \tag{12}$$

Solving the above LP yields solutions equivalent to those obtained by (8) because in the optimal solution, at least one of the two variables u_j and v_j will be zero for all $i = 1, \cdots, d$. Otherwise, assume $u_j > v_j > 0$ without loss of generality, and we can find a better solution by setting another feasible solution $\hat{u_j} = u_j - v_j$ and $\hat{v_j} = 0$. Then $\hat{u_j} + \hat{v_j} = u_j - v_j < u_j + v_j$ contradicting the optimality of $(\mathbf{u}, \mathbf{v})$.

There are two variables u_j, v_j corresponding to each column $\mathbf{K}_{\cdot j}$ of the kernel matrix in problem (12). Correspondingly the Lagrangian dual problem has two constraints for the column $\mathbf{K}_{\cdot j}$, i.e., $\sum_{i=1}^{\ell}\beta_i y_i \mathbf{K}_{ij} \leq 1$ and $-\sum_{i=1}^{\ell}\beta_i y_i \mathbf{K}_{ij} \leq 1$. Combining both constraints, we have $-1 \leq \sum_{i=1}^{\ell}\beta_i y_i \mathbf{K}_{ij} \leq 1$. Hence the dual problem becomes:

$$\begin{aligned} \max_{\boldsymbol{\beta}} \quad & \sum_{i=1}^{\ell}\beta_i \\ \text{s.t.} \quad & -1 \leq \sum_{i=1}^{\ell}\beta_i y_i \mathbf{K}_{ij} \leq 1,\ \ j=1,\cdots,d, \\ & \sum_{i=1}^{\ell}\beta_i y_i = 0, \\ & 0 \leq \beta_i \leq C,\ \ i=1,\cdots,\ell. \end{aligned} \tag{13}$$

The CG method partitions the variables α_j into two sets, the working set W that is used to build the model and the remaining set denoted as N that is eliminated from the model as the corresponding columns are not generated. Each CG step optimizes a subproblem over the working set W of variables and then selects a column from N based on the current solution to add to W. At each iteration, α_j in N can be interpreted as $\alpha_j = 0$, or accordingly, $u_j,\ v_j = 0$. Hence once a solution $\boldsymbol{\alpha}^W = \mathbf{u}^W - \mathbf{v}^W$ to the restricted problem is obtained, $\hat{\boldsymbol{\alpha}} = (\boldsymbol{\alpha}^W\ \boldsymbol{\alpha}^N = 0)$ is feasible to the master LP (12). The following statement examines when an optimal solution for the master problem is obtained in the CG procedure.

Proposition 1 (Optimality of LP CG) *Let $(\hat{\mathbf{u}}, \hat{\mathbf{v}}, \hat{\boldsymbol{\xi}}, \hat{\boldsymbol{\beta}})$ be the primal-dual solution to the current restricted problems with variable b included in W. The solution is optimal to LP (12) if and only if for all $j \in N$, $\left|\sum_i \hat{\beta}_i y_i \mathbf{K}_{ij}\right| \leq 1$.*

To show the optimality is achieved, we need to confirm primal feasibility, dual feasibility and the equality of primal and dual objectives. Recall how we define $\hat{\mathbf{u}} = (\mathbf{u}^W\ \mathbf{u}^N = 0)$ and $\hat{\mathbf{v}} = (\mathbf{v}^W\ \mathbf{v}^N = 0)$, so $(\hat{\mathbf{u}}, \hat{\mathbf{v}}, \hat{\boldsymbol{\xi}})$ is feasible for LP (12). Since the solution is optimal to the restricted problems, the primal objective is equal to the dual objective. Now the key issue to evaluate is the dual feasibility. Since $\hat{\boldsymbol{\beta}}$ is optimal for the restricted problem, it satisfies all constraints of the restricted dual. Hence the dual feasibility is validated if $\left|\sum_i \hat{\beta}_i y_i \mathbf{K}_{ij}\right| \leq 1,\ j \in N$.

Any column that violates dual feasibility can be added. For LPs, a common heuristic is to choose the column $\mathbf{K}_{\cdot j}$ that maximizes $\left|\sum_i \hat{\beta}_i y_i \mathbf{K}_{ij}\right|$ over all $j \in N$. In other words, the column $\mathbf{K}_{\cdot j}$ that solves

$$\tau = \max_{j \in N}\left|\sum_i \hat{\beta}_i y_i \mathbf{K}_{ij}\right| \tag{14}$$

will be included in the restricted problem. Compared with original LPBoost, our method is in general equivalent to enclosing negations of weak models $\mathbf{K}_{\cdot j}$ in the hypothesis set. We describe the modified LPBoost for semi-supervised learning with a mixture of kernels (SSMK) in Algorithm 1 where $\mathbf{e}$ is a vector of ones of appropriate dimension corresponding to the bias term b.

Algorithm 1 SSMK: CG Boosting for LP (12)

1. Initialize the first column $\mathbf{K}_0 = \mathbf{e}$, specify the tolerance tol
2. For $t = 1$ to T, do
3. Solve problem (12) with $\mathbf{K}_{t-1}$, obtain solution $(\mathbf{u}^t, \mathbf{v}^t, \boldsymbol{\xi}^t, \boldsymbol{\beta}^t)$
4. Solve problem (14) to obtain τ, and let $\mathbf{z}$ be the solution
5. If $\tau \leq 1 + tol$, optimal, break from loop, otherwise, $\mathbf{K}_t = [\mathbf{K}_{t-1}\ \ \mathbf{z}]$, continue
6. End of loop
7. $\hat{\boldsymbol{\alpha}} = \mathbf{u}^t - \mathbf{v}^t$.

After the column has been generated, we can either solve the updated primal problem or the updated dual problem. Any suitable algorithm can be used to solve the primal or dual restricted problem. The original LPBoost [10] solves the dual problem at each iteration. From the optimization point of view, it is not clear if solving the dual problem will be computationally cheaper than solving the primal. In Algorithm 1, we solve the primal problem since the current primal-dual solution is primal feasible to the updated problem, and there is no need to find the first feasible solution for the updated primal. Therefore solving each restricted primal can be cheap in CG algorithms. Another advantage of using the LPBoost-based approach is that at each iteration, only a small subset of the basis functions has to be kept in memory which is a limitation of traditional SVM formulations.

5 Experimental Study

In this section, we validate the proposed approach by investigating its prediction accuracy, optimization efficiency as well as sparsity of solutions in several experiments. We compare the performance of our approach SSMK to the models obtained by inductive learning with a mixture of kernels (MK) on two benchmark datasets and a real-world medical imaging dataset. Two types of kernels are explored: the linear kernel and RBF kernel. For the RBF kernel, we employ a fixed strategy to find the bandwidth. First we calculate the mean of $||\mathbf{x}_i - \mathbf{x}_j||^2$ where i, j run through the labeled examples, and then set bandwidth equal to square root of the mean value. More thorough comparison between inductive MK models and models based on single kernels can be found in [5].

5.1 Benchmark data

Two large databases were used in experiments. The Forest data came from UCI KDD Archive with 54 variables, 495,141 examples. The NIST hand-written digit database was downloaded from *http://yann.lecun.com/exdb/mnist/*. Each digit is represented by a 28×28 image so there are 784 variables in total. The database contains 60,000 digits. We created binary classification problems by distinguishing between spruce-fir and lodgepole pine cover types on Forest data, and discriminating between odd numbers and even numbers on NIST Digits data. The distribution of classes in both problems is balanced, so we mainly report misclassification rates in Figure 1. Commercial optimization package ILOG CPLEX 8.0 [13] was used as the LP solver.

We generated datasets in the following way: preserve $\ell_t = 2000$ examples randomly drawn from the databases as test sets (T); randomly take $\ell_u = 500$ examples from the remaining data as unlabelled data (U) used in model construction; then randomly take ℓ =10, 20, 50, 100, 200, 300, 400, 500 examples as training data (L) with labels. We performed 10 trials, and present error rates averaged on these trials in Figure 1. The data was preprocessed by normalizing each original feature to have mean 0 and standard deviation 1. In the training phase, the computational complexity of the LPs can be roughly measured by the size of the kernel matrices. For example, if we choose 100 labeled examples, the size of the kernel matrix used in LP(12) is 100×1200 considering there are 500 unlabelled data and 2 types of kernels. All results were obtained by setting regularization parameter C to 10 for Digits data and to 1 for Forest data. Other choices of C produced similar comparison results between our approach SSMK and the inductive MK model with no use of unlabeled data.

Clearly, from Figure 1, the use of unlabeled data in LP(12) helps improve the prediction performance both on

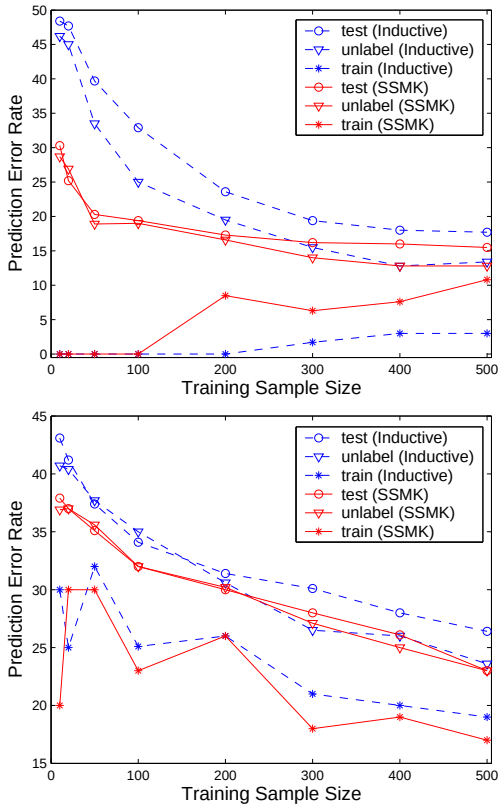

Figure 1. Plots of error percentage versus training size ℓ for Digits *up* and Forest *bottom*.

the unlabeled data which is used in training with no labels and the test data which is completely blinded to the training process. With more and more labeled data available, the difference between inductive MK and SSMK is generally reduced. It is reasonable to see that performance on unlabeled data is slightly better than test prediction for SSMK. Table 1 provides more detailed comparison between inductive MK (IMK) and SSMK. By applying the generalized LPBoost, we dramatically reduced the size of optimization problems to be solved and thus gained computational efficiency. For example, when $\ell = 100$ for Digits, we solved a problem of very large size 100×1200 in IMK, and 47 linear kernel basis and 2 RBF basis were used in the final model (see Table 1). In SSMK with the CG procedure, we need to solve problems of much smaller size (the largest size = 100×34 during the 34 (given by (SSMK)CG-Iter in Table 1) iterations. The resulting model used only 26 linear kernel columns. Hence SSMK had potentials to find more sparse solutions although both IMK and SSMK obtained sparse solutions due to the use of 1-norm regularization. We can see SSMK benefited from the use of unlabeled data by referencing the last two rows of Table 1. These two terms give

Table 1. Efficiency comparison between SSMK and inductive MK.

Data	Digits			Forest		
ℓ	100	200	500	100	200	500
Problem size	100×1200	200×1400	500×2000	100×1200	200×1400	500×2000
(IMK)linear	47	107	127	15	18	23
(IMK)RBF	2	5	4	0	0	3
(SSMK)CG-Iter	34	38	37	23	30	48
(SSMK) subproblem size	100×34	200×38	500×37	100×23	200×30	500×48
(SSMK)linear	26	30	35	16	18	23
(SSMK)RBF	0	0	0	0	1	2
(SSMK)K_L	9	5	7	5	5	9
(SSMK)K_U	17	25	28	11	14	16

the numbers of kernel columns generated from the labeled data (S)K_L and the unlabeled data (S)K_U at the end of CG optimization.

5.2 Medical imaging data for CAD

We also tested the proposed algorithm on a real-world data set used in our Computer-Aided Detection (CAD) system for identifying lung nodules. Typical CAD data sets for classification are large (several thousand candidates) and unbalanced (significantly fewer than 1% of the candidates are "positive"). To be accepted by physicians, CAD systems must generalize well with extremely high sensitivity and very few false positives. Researchers often generate a large amount of candidates for structures of interest in the image - for example, cancerous nodules in a CT (computed tomography) volume of a patients lung, or polyps in a CT volume of colons. Only a small subset of candidates actually correspond to nodules or polyps. It is extremely expensive and tedious to ask physicians to label the obtained candidates with high confidence. The typical workflow for a CAD system when used to identify structures in a new patient image is: 1. identify candidate structures in the image; 2. extract features for each candidate; 3. classify candidates as positive or negative (using a classifier trained on the feature matrix generated by the training candidates); 4. display a positive candidate.

In our study, we used a dataset of 112 high-resolution CT images that was obtained from one North American and three European sites. The CT data was acquired with the following scanners: Siemens VolumeZoom CT, a Sensation 16, and a Sensation 64 scanner, all with 100kV, 120kV, or 140kV, and an exposure from 20mAs to 253mAs. All volumes were reconstructed with a B50f or B60f reconstruction kernel. The slice thickness ranged in 1.0mm and 1.25mm with an axial resolution between 0.5mm and 0.8mm. The ground truth labels used in this study were established by 3 radiologists (1 expert and 2 experienced readers). Our in-house candidate generation algorithm generated 22405 candidates and 42 features were calculated. We split the candidate set into the labeled and unlabeled training sets of 5000 candidates respectively for each, and all remaining candidates are in the test set. Instead of computing the linear kernel, we used the data matrix itself to form $\mathbf{K} = [\mathbf{X} \;\; \mathbf{K}_{rbf}]$ where $\mathbf{X}$ is the input data and $\mathbf{K}_{rbf}$ is the RBF kernel matrix. Figure 2 shows the ROC curves plotting sensitivity versus false positive value per volume. For the respective classifiers obtained by IMK and SSMK, figure 2 shows two curves, one for unlabeled data performance, the other for test performance. We can see that with more choices of basis functions constructed from unlabeled data, the prediction accuracy is improved, especially over the small false positive rates from 1 to 3 which is the desired range of prediction accuracy for lung nodule detection system. Due to the 1-norm regularization condition, SSMK actually performed as a feature selection algorithm. Both IMK and SSMK selected small subsets of features or basis functions (around 25 features), and SSMK ran 100 times faster than solving problem (12) without the CG boosting scheme.

6 Conclusions

We have explored the mixture-of-kernels model in semi-supervised learning settings. By using kernels with different geometric properties, allowing kernels to associate with different bandwidths or parameter values, and allowing the basis functions introduced by kernels to locate at unlabelled data points, we enhance the capability of kernel methods to adapt to tasks with various target functions. The optimization problems involved in model construction can be solved in a very efficient way through the exploitation of column generation techniques.

SVM-like algorithms have been previously proposed for transductive learning as integer programs [2] and semidefinite programs [6] using overall risk minimization posed by Vapnik [21]. To achieve computational efficiency, our

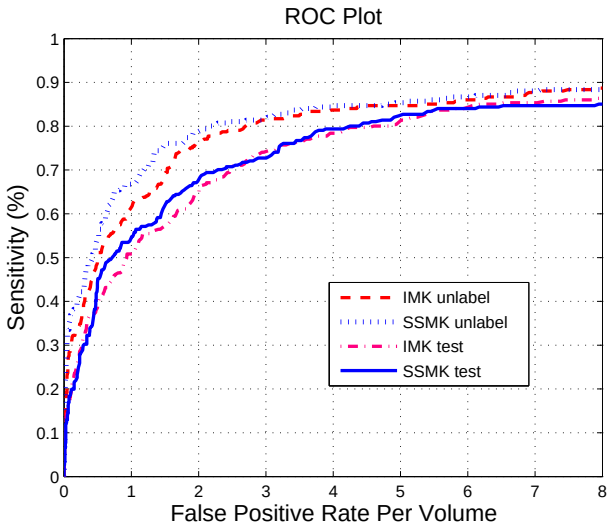

Figure 2. Plots of ROC curves for lungCAD.

method does not minimize the overall risk. One of the future work is to extend the CG approach to integer programs formed for Vapnik's transduction problems. Another direction is to incorporate our approach in active learning tasks where a set of unlabelled examples need to be identified and given priority to obtain their observed labels in experimental investigation. The columns that are selected by the SSMK algorithm provide clues for which examples should be examined in the subsequent experiment.

References

[1] M. S. Bazaraa, H. D. Sherali, and C. M. Shetty. *Nonlinear Programming: Theory and Algorithms*. John Wiley & Sons, Inc., New York, NY, 1993.

[2] K. Bennett and A. Demiriz. Semi-supervised support vector machines. In D. A. Cohn, M. S. Kearns, and S. A. Solla, editors, *Advances in Neural Information Processing Systems 12*, pages 368–374. MIT Press, Cambridge, MA, 1998.

[3] K. Bennett, M. Momma, and M. Embrechts. MARK: A boosting algorithm for heterogeneous kernel models. In *Proceedings of SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 24–31, 2002.

[4] K. P. Bennett. *Machine Learning via Mathematical Programming*. PhD thesis, University of Wisconsin, Madison, Wisconsin, 1993.

[5] J. Bi, T. Zhang, and K. P. Bennett. Column-generation boosting methods for mixture of kernels. In R. Kohavi, J. Gehrke, W. DuMouchel, and J. Ghosh, editors, *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 521–526, 2004.

[6] T. D. Bie and N. Cristianini. Convex methods for transduction. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16*. MIT Press, Cambridge, MA, 2004.

[7] C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, New York, 1995.

[8] O. Bousquet and D. J. L. Herrmann. On the complexity of learning the kernel matrix. In S. T. S. Becker and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 399–406. MIT Press, Cambridge, MA, 2003.

[9] K. Crammer, J. Keshet, and Y. Singer. Kernel design using boosting. In S. T. S. Becker and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 537–544. MIT Press, Cambridge, MA, 2003.

[10] A. Demiriz, K. P. Bennett, and J. Shawe-Taylor. Linear programming boosting via column generation. *Machine Learning*, 46(1–3):225–254, 2002.

[11] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55:119–139, 1997.

[12] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: data mining, inference and prediction*. Springer-Verlag, New York, 2001.

[13] ILOG. *CPLEX Optimizer*. ILOG CPLEX Division, 889 Alder Avenue, Incline Village, Nevada, *http://www.cplex.com/*, 2004.

[14] G. Lanckriet, N. Cristianini, P. Bartlett, L. Ghaoui, and M. I. Jordan. Learning the kernel matrix with semidefinite programming. *Journal of Machine Learning Research*, 5:27–72, 2003.

[15] O. L. Mangasarian. Generalized support vector machines. In P. Bartlett, B. Schölkopf, D. Schuurmans, and A. Smola, editors, *Advances in Large Margin Classifiers*, pages 135–146. MIT Press, 2000.

[16] S. G. Nash and A. Sofer. *Linear and Nonlinear Programming*. McGraw-Hill, New York, NY, 1996.

[17] C. S. Ong, A. J. Smola, and R. C. Williamson. Hyperkernels. In S. T. S. Becker and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 478–485. MIT Press, Cambridge, MA, 2003.

[18] E. Parrado-Hernandez, J. Arenas-Garca, I. Mora-Jimenez, , and A. Navia-Vazquez. On problem oriented kernel refining. *Neurocomputing*, 55:135–150, 2003.

[19] E. Parrado-Hernandez, I. Mora-Jimenez, J. Arenas-Garcia, A. R. Figueiras-Vidal, and A. Navia-Vazquez. Growing support vector classifiers with controlled complexity. *Pattern Recognition Letters*, 36:1479–1484, 2003.

[20] B. Schölkopf, K.Sung, C. Burges, F. Girosi, P. Niyogi, T. Poggio, and V. Vapnik. Comparing support vector machines with gaussian kernels to radial basis function classifiers. *IEEE Transactions on Signal Processing*, 45(11):2758–2765, 1997.

[21] V. N. Vapnik. *Statistical Learning Theory*. John Wiley & Sons, Inc., New York, 1998.

[22] X. Zhu. Semi-supervised learning with graphs. In *Doctoral Thesis, CMU-LTI-05-192*. Carnegie Mellon University, 2005.