Learning the Unified Kernel Machines for Classification

Steven C. H. Hoi
CSE, Chinese University of Hong Kong
chhoi@cse.cuhk.edu.hk

Michael R. Lyu
CSE, Chinese University of Hong Kong
lyu@cse.cuhk.edu.hk

Edward Y. Chang
ECE, University of California, Santa Barbara
echang@ece.ucsb.edu

ABSTRACT

Kernel machines have been shown as the state-of-the-art learning techniques for classification. In this paper, we propose a novel general framework of learning the Unified Kernel Machines (UKM) from both labeled and unlabeled data. Our proposed framework integrates supervised learning, semi-supervised kernel learning, and active learning in a unified solution. In the suggested framework, we particularly focus our attention on designing a new semi-supervised kernel learning method, i.e., Spectral Kernel Learning (SKL), which is built on the principles of kernel target alignment and unsupervised kernel design. Our algorithm is related to an equivalent quadratic programming problem that can be efficiently solved. Empirical results have shown that our method is more effective and robust to learn the semi-supervised kernels than traditional approaches. Based on the framework, we present a specific paradigm of unified kernel machines with respect to Kernel Logistic Regresions (KLR), i.e., Unified Kernel Logistic Regression (UKLR). We evaluate our proposed UKLR classification scheme in comparison with traditional solutions. The promising results show that our proposed UKLR paradigm is more effective than the traditional classification approaches.

Categories and Subject Descriptors

I.5.2 [**PATTERN RECOGNITION**]: Design Methodology—*Classifier design and evaluation*; H.2.8 [**Database Management**]: Database Applications—*Data mining*

General Terms

Design, Algorithms, Experimentation

Keywords

Classification, Kernel Machines, Spectral Kernel Learning, Supervised Learning, Semi-Supervised Learning, Unsupervised Kernel Design, Kernel Logistic Regressions, Active Learning

KDD'06, August 20–23, 2006, Philadelphia, Pennsylvania, USA.

1. INTRODUCTION

Classification is a core data mining technique and has been actively studied in the past decades. In general, the goal of classification is to assign unlabeled testing examples with a set of predefined categories. Traditional classification methods are usually conducted in a supervised learning way, in which only labeled data are used to train a predefined classification model. In literature, a variety of statistical models have been proposed for classification in the machine learning and data mining communities. One of the most popular and successful methodologies is the kernel-machine techniques, such as Support Vector Machines (SVM) [25] and Kernel Logistic Regressions (KLR) [29]. Like other early work for classification, traditional kernel-machine methods are usually performed in the supervised learning way, which consider only the labeled data in the training phase.

It is obvious that a good classification model should take advantages on not only the labeled data, but also the unlabeled data when they are available. Learning on both labeled and unlabeled data has become an important research topic in recent years. One way to exploit the unlabeled data is to use active learning [7]. The goal of active learning is to choose the most informative example from the unlabeled data for manual labeling. In the past years, active learning has been studied for many classification tasks [16].

Another emerging popular technique to exploit unlabeled data is semi-supervised learning [5], which has attracted a surge of research attention recently [30]. A variety of machine-learning techniques have been proposed for semi-supervised learning, in which the most well-known approaches are based on the graph Laplacians methodology [28, 31, 5]. While promising results have been popularly reported in this research topic, there is so far few comprehensive semi-supervised learning scheme applicable for large-scale classification problems.

Although supervised learning, semi-supervised learning and active learning have been studied separately, so far there is few comprehensive scheme to combine these techniques effectively together for classification tasks. To this end, we propose a general framework of learning the Unified Kernel Machines (UKM) [3, 4] by unifying supervised kernel-machine learning, semi-supervised learning, unsupervised kernel design and active learning together for large-scale classification problems.

The rest of this paper is organized as follows. Section 2 reviews related work of our framework and proposed solutions. Section 3 presents our framework of learning the unified ker-

nel machines. Section 4 proposes a new algorithm of learning semi-supervised kernels by Spectral Kernel Learning (SKL). Section 5 presents a specific UKM paradigm for classification, i.e., the Unified Kernel Logistic Regression (UKLR). Section 6 evaluates the empirical performance of our proposed algorithm and the UKLR classification scheme. Section 7 sets out our conclusion.

2. RELATED WORK

Kernel machines have been widely studied for data classification in the past decade. Most of earlier studies on kernel machines usually are based on supervised learning. One of the most well-known techniques is the Support Vector Machines, which have achieved many successful stories in a variety of applications [25]. In addition to SVM, a series of kernel machines have also been actively studied, such as Kernel Logistic Regression [29], Boosting [17], Regularized Least-Square (RLS) [12] and Minimax Probability Machines (MPM) [15], which have shown comparable performance with SVM for classification. The main theoretical foundation behind many of the kernel machines is the theory of regularization and reproducing kernel Hilbert space in statistical learning [17, 25]. Some theoretical connections between the various kernel machines have been explored in recent studies [12].

Semi-supervised learning has recently received a surge of research attention for classification [5, 30]. The idea of semi-supervised learning is to use both labeled and unlabeled data when constructing the classifiers for classification tasks. One of the most popular solutions in semi-supervised learning is based on the graph theory [6], such as Markov random walks [22], Gaussian random fields [31], Diffusion models [13] and Manifold learning [2]. They have demonstrated some promising results on classification.

Some recent studies have begun to seek connections between the graph-based semi-supervised learning and the kernel machine learning. Smola and Kondor showed some theoretical understanding between kernel and regularization based on the graph theory [21]. Belkin et al. developed a framework for regularization on graphs and provided some analysis on generalization error bounds [1]. Based on the emerging theoretical connections between kernels and graphs, some recent work has proposed to learn the semi-supervised kernels by graph Laplacians [32]. Zhang et al. recently provided a theoretical framework of unsupervised kernel design and showed that the graph Laplacians solution can be considered as an equivalent kernel learning approach [27]. All of the above studies have formed the solid foundation for semi-supervised kernel learning in this work.

To exploit the unlabeled data, another research attention is to employ active learning for reducing the labeling efforts in classification tasks. Active learning, or called pool-based active learning, has been proposed as an effective technique for reducing the amount of labeled data in traditional supervised classification tasks [19]. In general, the key of active learning is to choose the most informative unlabeled examples for manual labeling. A lot of active learning methods have been proposed in the community. Typically they measure the classification uncertainty by the amount of disagreement to the classification model [9, 10] or measure the distance of each unlabeled example away from the classification boundary [16, 24].

3. FRAMEWORK OF LEARNING UNIFIED KERNEL MACHINES

In this section, we present the framework of learning the unified kernel machines by combining supervised kernel machines, semi-supervised kernel learning and active learning techniques into a unified solution. Figure 1 gives an overview of our proposed scheme. For simplicity, we restrict our discussions to classification problems.

Let $\mathcal{M}(K, \alpha)$ denote a kernel machine that has some underlying probabilistic model, such as kernel logistic regressions (or support vector machines). In general, a kernel machine contains two components, i.e., the kernel K (either a kernel function or simply a kernel matrix), and the model parameters α. In traditional supervised kernel-machine learning, the kernel K is usually a known parametric kernel function and the goal of the learning task is usually to determine the model parameter α. This often limits the performance of the kernel machine if the specified kernel is not appropriate.

To this end, we propose a unified scheme to learn the unified kernel machines by learning on both the kernel K and the model parameters α together. In order to exploit the unlabeled data, we suggest to combine semi-supervised kernel learning and active learning techniques together for learning the unified kernel machines effectively from the labeled and unlabeled data. More specifically, we outline a general framework of learning the unified kernel machine as follows.

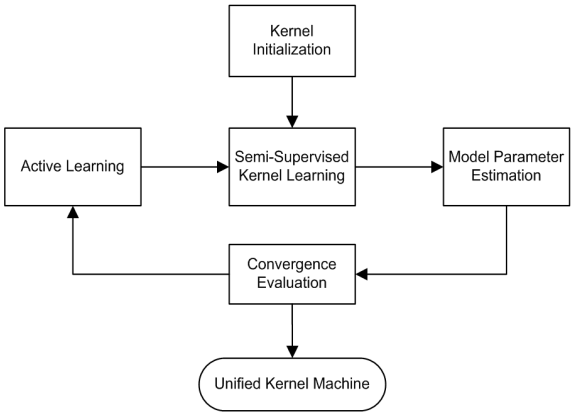


Figure 1: Learning the Unified Kernel Machines

Let L denote the labeled data and U denote the unlabeled data. The goal of the unified kernel machine learning task is to learn the kernel machine $\mathcal{M}(K^*, \alpha^*)$ that can classify the data effectively. Specifically, it includes the following five steps:

- **Step 1. Kernel Initialization**

  The first step is to initialize the kernel component K_0 of the kernel machine $\mathcal{M}(K_0, \alpha_0)$. Typically, users can specify the initial kernel K_0 (function or matrix) with a standard kernel. When some domain knowledge is available, users can also design some kernel with domain knowledge (or some data-dependent kernels).

- **Step 2. Semi-Supervised Kernel Learning**

  The initial kernel may not be good enough to classify the data correctly. Hence, we suggest to employ

the semi-supervised kernel learning technique to learn a new kernel K by engaging both the labeled L and unlabeled data U available.

- **Step 3. Model Parameter Estimation**

  When the kernel K is known, to estimate the parameters of the kernel machines based on some model assumption, such as Kernel Logistic Regression or Support Vector Machines, one can simply employ the standard supervised kernel-machine learning to solve the model parameters α.

- **Step 4. Active Learning**

  In many classification tasks, labeling cost is expensive. Active learning is an important method to reduce human efforts in labeling. Typically, we can choose a batch of most informative examples S that can most effectively update the current kernel machine $\mathcal{M}(K, \alpha)$.

- **Step 5. Convergence Evaluation**

  The last step is the convergence evaluation in which we check whether the kernel machine is good enough for the classification task. If not, we will repeat the above steps until a satisfied kernel machine is acquired.

This is a general framework of learning unified kernel machines. In this paper, we focus our main attention on the the part of semi-supervised kernel learning technique, which is a core component of learning the unified kernel machines.

4. SPECTRAL KERNEL LEARNING

We propose a new semi-supervised kernel learning method, which is a fast and robust algorithm for learning semi-supervised kernels from labeled and unlabeled data. In the following parts, we first introduce the theoretical motivations and then present our spectral kernel learning algorithm. Finally, we show the connections of our method to existing work and justify the effectiveness of our solution from empirical observations.

4.1 Theoretical Foundation

Let us first consider a standard supervised kernel learning problem. Assume that the data (X, Y) are drawn from an unknown distribution $\mathcal{D}$. The goal of supervised learning is to find a prediction function $p(\mathbf{X})$ that minimizes the following expected true loss:

$$E_{(X,Y)\sim\mathcal{D}}\mathcal{L}(p(X), Y),$$

where $E_{(X,Y)\sim\mathcal{D}}$ denotes the expectation over the true underlying distribution $\mathcal{D}$. In order to achieve a stable estimation, we usually need to restrict the size of hypothesis function family. Given l training examples $(\mathbf{x}_1,y_1),\ldots,(\mathbf{x}_l,y_l)$, typically we train a prediction function $\hat{p}$ in a reproducing Hilbert space $\mathcal{H}$ by minimizing the empirical loss [25]. Since the reproducing Hilbert space can be large, to avoid overfitting problems, we often consider a regularized method as follow:

$$\hat{p} = \arg\inf_{p\in\mathcal{H}} \left(\frac{1}{l}\sum_{i=1}^{l} \mathcal{L}(p(\mathbf{x}_i), y_i) + \lambda ||p||^2_{\mathcal{H}} \right), \qquad (1)$$

where λ is a chosen positive regularization parameter. It can be shown that the solution of (1) can be represented as the following kernel method:

$$\hat{p}(\mathbf{x}) = \sum_{i=1}^{l} \hat{\alpha}_i k(\mathbf{x}_i, \mathbf{x})$$

$$\boldsymbol{\alpha} = \arg\inf_{\boldsymbol{\alpha}\in\mathbb{R}^l} \left(\frac{1}{l}\sum_{i=1}^{l} \mathcal{L}\left(p(\mathbf{x}_i), y_i\right) + \lambda \sum_{i,j=1}^{l} \alpha_i\alpha_j k(\mathbf{x}_i, \mathbf{x}_j) \right),$$

where $\boldsymbol{\alpha}$ is a parameter vector to be estimated from the data and k is a kernel, which is known as *kernel function*. Typically a *kernel* returns the inner product between the mapping images of two given data examples, such that $k(\mathbf{x}_i, \mathbf{x}_j) = \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle$ for $\mathbf{x}_i, \mathbf{x}_j \in \mathcal{X}$.

Let us now consider a semi-supervised learning setting. Given labeled data $\{(\mathbf{x}_i, y_i)\}_{i=1}^{l}$ and unlabeled data $\{\mathbf{x}_j\}_{j=l+1}^{n}$, we consider to learn the real-valued vectors $f \in \mathbb{R}^n$ by the following semi-supervised learning method:

$$\hat{f} = \arg\inf_{f\in\mathbb{R}^n} \left(\frac{1}{n}\sum_{i=1}^{n} \mathcal{L}(f_i, y_i) + \lambda f^\top K^{-1} f \right), \qquad (2)$$

where K is an $n \times n$ kernel matrix with $K_{i,j} = k(\mathbf{x}_i, \mathbf{x}_j)$. Zhang et al. [27] proved that the solution of the above semi-supervised learning is equivalent to the solution of standard supervised learning in (1), such that

$$\hat{f}_j = \hat{p}(\mathbf{x}_j) \quad j = 1, \ldots, n. \qquad (3)$$

The theorem offers a principle of unsupervised kernel design: one can design a new kernel $\bar{k}(\cdot,\cdot)$ based on the unlabeled data and then replace the original kernel k by $\bar{k}$ in the standard supervised kernel learning. More specifically, the framework of spectral kernel design suggests to design the new kernel matrix $\bar{K}$ by a function g as follows:

$$\bar{K} = \sum_{i=1}^{n} g(\lambda_i)\mathbf{v}_i\mathbf{v}_i^\top, \qquad (4)$$

where $(\lambda_i, \mathbf{v}_i)$ are the eigen-pairs of the original kernel matrix K, and the function $g(\cdot)$ can be regarded as a filter function or a transformation function that modifies the spectra of the kernel. The authors in [27] show a theoretical justification that designing a kernel matrix with faster spectral decay rates should result in better generalization performance, which offers an important principle in learning an effective kernel matrix.

On the other hand, there are some recent papers that have studied theoretical principles for learning effective kernel functions or matrices from labeled and unlabeled data. One important work is the *kernel target alignment*, which can be used not only to assess the relationship between the feature spaces by two kernels, but also to measure the similarity between the feature space by a kernel and the feature space induced by labels [8]. Specifically, given two kernel matrices K_1 and K_2, their relationship is defined by the following score of *alignment*:

DEFINITION 1. **Kernel Alignment:** *The empirical alignment of two given kernels K_1 and K_2 with respect to the sample set S is the quantity:*

$$\hat{A}(K_1, K_2) = \frac{\langle K_1, K_2\rangle_F}{\sqrt{\langle K_1, K_1\rangle_F \langle K_2, K_2\rangle_F}} \qquad (5)$$

where K_i is the kernel matrix induced by the kernel k_i and $\langle \cdot, \cdot \rangle$ is the Frobenius product between two matrices, i.e., $\langle K_1, K_2 \rangle_F = \sum_{i,j=1}^{n} k_1(\mathbf{x}_i, \mathbf{x}_j) k_2(\mathbf{x}_i, \mathbf{x}_j)$.

The above definition of *kernel alignment* offers a principle to learn the kernel matrix by assessing the relationship between a given kernel and a *target kernel* induced by the given labels. Let $\mathbf{y} = \{y_i\}_{i=1}^{l}$ denote a vector of labels in which $y_i \in \{+1, -1\}$ for binary classification. Then the target kernel can be defined as $T = \mathbf{y}\mathbf{y}^\top$. Let K be the kernel matrix with the following structure

$$K = \begin{pmatrix} K_{tr} & K_{trt} \\ K_{trt}^\top & K_t \end{pmatrix} \tag{6}$$

where $K_{ij} = \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle$, K_{tr} denotes the matrix part of "train-data block" and K_t denotes the matrix part of "test-data block."

The theory in [8] provides the principle of learning the kernel matrix, i.e., looking for a kernel matrix K with good generalization performance is equivalent to finding the matrix that maximizes the following empirical kernel alignment score:

$$\hat{A}(K_{tr}, T) = \frac{\langle K_{tr}, T \rangle_F}{\sqrt{\langle K_{tr}, K_{tr} \rangle_F \langle T, T \rangle_F}} \tag{7}$$

This principle has been used to learn the kernel matrices with multiple kernel combinations [14] and also the semi-supervised kernels from graph Laplacians [32]. Motivated by the related theoretical work, we propose a new spectral kernel learning (SKL) algorithm which learns spectra of the kernel matrix by obeying both the principle of unsupervised kernel design and the principle of kernel target alignment.

4.2 Algorithm

Assume that we are given a set of labeled data $L = \{\mathbf{x}_i, y_i\}_{i=1}^{l}$, a set of unlabeled data $U = \{\mathbf{x}_i\}_{i=l+1}^{n}$, and an initial kernel matrix K. We first conduct the eigen-decomposition of the kernel matrix:

$$K = \sum_{i=1}^{n} \lambda_i \mathbf{v}_i \mathbf{v}_i^\top \ , \tag{8}$$

where $(\lambda_i, \mathbf{v}_i)$ are eigen pairs of K and are assumed in a decreasing order, i.e., $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n$. For efficiency consideration, we select the top d eigen pairs, such that

$$K_d = \sum_{i=1}^{d} \lambda_i \mathbf{v}_i \mathbf{v}_i^\top \approx K \ , \tag{9}$$

where the parameter $d \ll n$ is a dimension cutoff factor that can be determined by some criteria, such as the cumulative eigen energy.

Based on the principle of unsupervised kernel design, we consider to learn the kernel matrix as follows

$$\bar{K} = \sum_{i=1}^{d} \mu_i \mathbf{v}_i \mathbf{v}_i^\top \ , \tag{10}$$

where $\mu_i \geq 0$ are spectral coefficients of the new kernel matrix. The goal of spectral kernel learning (SKL) algorithm is to find the optimal spectral coefficients μ_i for the following optimization

$$\begin{aligned} \max_{\bar{K}, \mu} \quad & \hat{A}(\bar{K}_{tr}, T) \\ \text{subject to} \quad & \bar{K} = \textstyle\sum_{i=1}^{d} \mu_i \mathbf{v}_i \mathbf{v}_i^\top \\ & trace(\bar{K}) = 1 \\ & \mu_i \geq 0, \\ & \mu_i \geq C\mu_{i+1}, i = 1, \ldots, d-1 \ , \end{aligned} \tag{11}$$

where C is introduced as a decay factor that satisfies $C \geq 1$, $\mathbf{v}_i$ are top d eigen vectors of the original kernel matrix K, $\bar{K}_{tr}$ is the kernel matrix restricted to the (labeled) training data and T is the target kernel induced by labels. Note that C is introduced as an important parameter to control the decay rate of spectral coefficients that will influence the overall performance of the kernel machine.

The above optimization problem belongs to convex optimization and is usually regarded as a semi-definite programming problem (SDP) [14], which may not be computationally efficient. In the following, we turn it into a Quadratic Programming (QP) problem that can be solved much more efficiently.

By the fact that the objective function (7) is invariant to the constant term $\langle T, T \rangle_F$, we can rewrite the objective function into the following form

$$\frac{\langle \bar{K}_{tr}, T \rangle_F}{\sqrt{\langle \bar{K}_{tr}, \bar{K}_{tr} \rangle_F}} \ . \tag{12}$$

The above alignment is invariant to scales. In order to remove the trace constraint in (11), we consider the following alternative approach. Instead of maximizing the objective function (12) directly, we can fix the numerator to 1 and then minimize the denominator. Therefore, we can turn the optimization problem into:

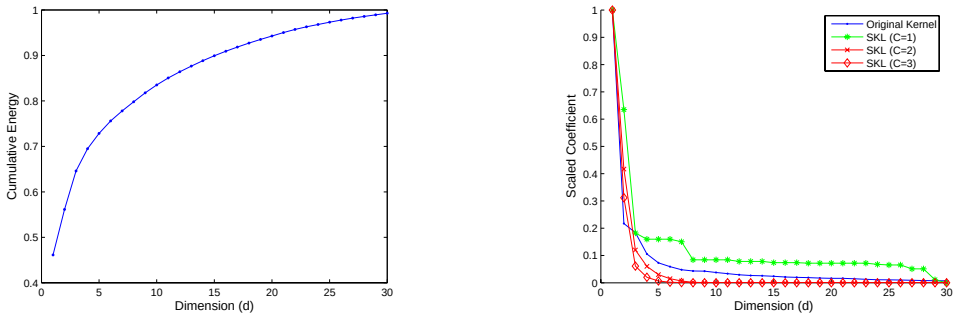
$$\begin{aligned} \min_{\mu} \quad & \sqrt{\langle \bar{K}_{tr}, \bar{K}_{tr} \rangle_F} \\ \text{subject to} \quad & \bar{K} = \textstyle\sum_{i=1}^{d} \mu_i \mathbf{v}_i \mathbf{v}_i^\top \\ & \langle \bar{K}_{tr}, T \rangle_F = 1 \\ & \mu_i \geq 0, \\ & \mu_i \geq C\mu_{i+1}, i = 1, \ldots, d-1 \ . \end{aligned} \tag{13}$$

This minimization problem without the trace constraint is equivalent to the original maximization problem with the trace constraint.

Let $vec(A)$ denote the column vectorization of a matrix A and let $D = [vec(V_{1,tr}) \ldots vec(V_{d,tr})]$ be a constant matrix with size of $l^2 \times d$, in which the d matrices of $V_i = \mathbf{v}_i \mathbf{v}_i^\top$ are with size of $l \times l$. It is not difficult to show that the above problem is equivalent to the following optimization

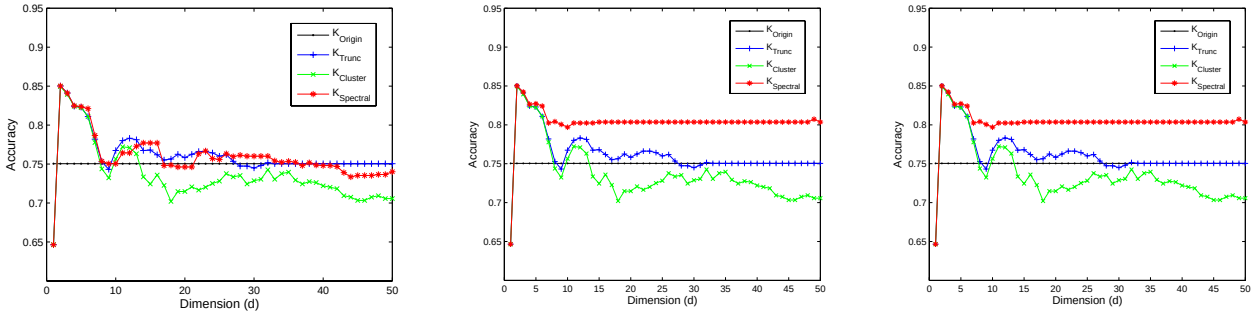
$$\begin{aligned} \min_{\mu} \quad & ||D\mu|| \\ \text{subject to} \quad & vec(T)^\top D\mu = 1 \\ & \mu_i \geq 0 \\ & \mu_i \geq C\mu_{i+1}, i = 1, \ldots, d-1 \ . \end{aligned} \tag{14}$$

Minimizing the norm is then equivalent to minimizing the squared norm. Hence, we can obtain the final optimization



(a) Cumulative eigen energy (b) Spectral coefficients

Figure 2: Illustration of cumulative eigen energy and the spectral coefficients of different decay factors on the Ionosphere dataset. The initial kernel is a linear kernel and the number of labeled data is 20.



(a) C=1 (b) C=2 (c) C=3

Figure 3: Classification performance of semi-supervised kernels with different decay factors on the Ionosphere dataset. The initial kernel is a linear kernel and the number of labeled data is 20.

problem as

$$\begin{aligned} \min_{\mu} \quad & \mu^\top D^\top D\mu \\ \text{subject to} \quad & vec(T)^\top D\mu = 1 \\ & \mu_i \geq 0 \\ & \mu_i \geq C\mu_{i+1}, i = 1, \ldots, d-1 \ . \end{aligned}$$

This is a standard Quadratic Programming (QP) problem that can be solved efficiently.

4.3 Connections and Justifications

The essential of our semi-supervised kernel learning method is based on the theories of unsupervised kernel design and kernel target alignment. More specifically, we consider a dimension-reduction effective method to learn the semi-supervised kernel that maximizes the kernel alignment score. By examining the work on unsupervised kernel design, the following two pieces of work can be summarized as a special case of spectral kernel learning framework:

- **Cluster Kernel**

  This method adopts a "[1,...,1,0,...,0]" kernel that has been used in spectral clustering [18]. It sets the top spectral coefficients to 1 and the rest to 0, i.e.,

$$\mu_i = \begin{cases} 1 & \text{for} \quad i \leq d \\ 0 & \text{for} \quad i > d \end{cases} \ . \tag{15}$$

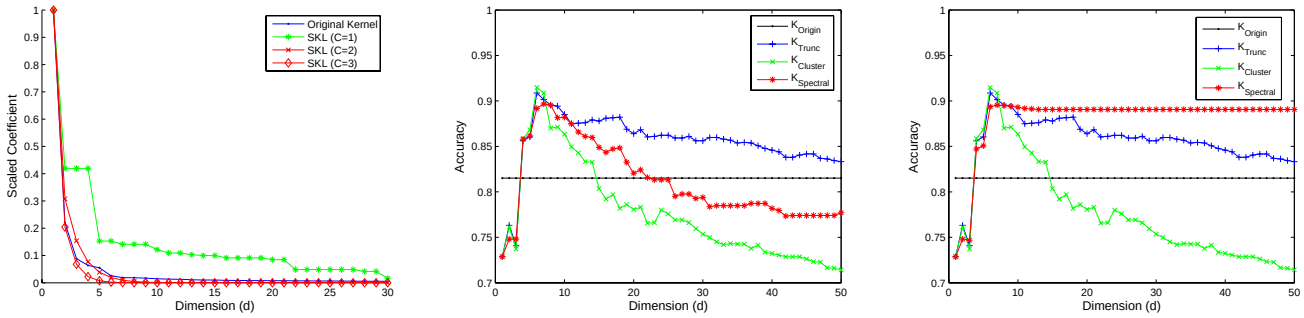
  For a comparison, we refer to this method as "Cluster kernel" denoted by $K_{\texttt{Cluster}}$.

- **Truncated Kernel**

  Another method is called the truncated kernel that keeps only the top d spectral coefficients

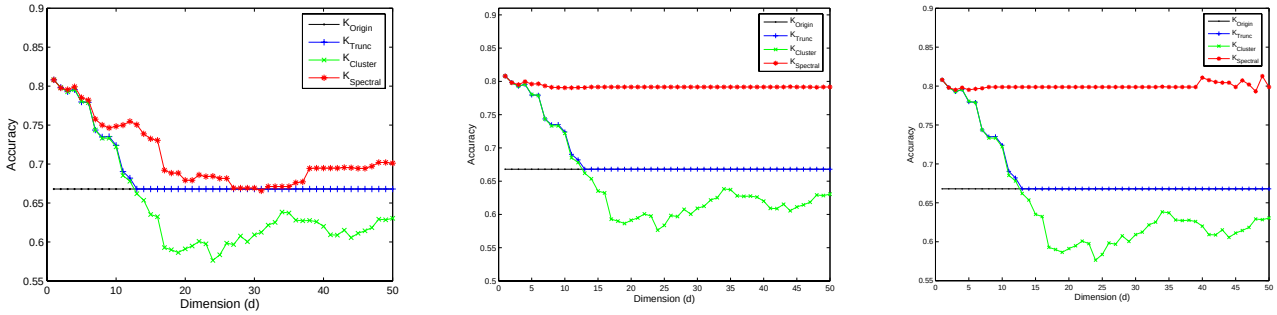
$$\mu_i = \begin{cases} \lambda_i & \text{for} \quad i \leq d \\ 0 & \text{for} \quad i > d \end{cases} \ , \tag{16}$$

  where λ_i are the top eigen values of an initial kernel. We can see that this is exactly the method of kernel principal component analysis [20] that keeps only the d most significant principal components of a given kernel. For a comparison, we denote this method as $K_{\texttt{Trunc}}$.



(a) Spectral coefficients (b) C=1 (c) C=2

Figure 4: Example of Spectral coefficients and performance impacted by different decay factors on the Ionosphere dataset. The initial kernel is an RBF kernel and the number of labeled data is 20.



(a) C=1 (b) C=2 (c) C=3

Figure 5: Classification performance of semi-supervised kernels with different decay factors on the Heart dataset. The initial kernel is a linear kernel and the number of labeled data is 20.

In our case, in comparison with semi-supervised kernel learning methods by graph Laplacians, our work is similar to the approach in [32], which learns the spectral transformation of graph Laplacians by kernel target alignment with order constraints. However, we should emphasize two important differences that will explain why our method can work more effectively.

First, the work in [32] belongs to traditional graph based semi-supervised learning methods which assume the kernel matrix is derived from the spectral decomposition of graph Laplacians. Instead, our spectral kernel learning method learns on any initial kernel and assume the kernel matrix is derived from the spectral decomposition of the normalized kernel.

Second, compared to the kernel learning method in [14], the authors in [32] proposed to add order constraints into the optimization of kernel target alignment [8] to enforce the constraints of graph smoothness. In our case, we suggest a decay factor C to constrain the relationship of spectral coefficients in the optimization that can make the spectral coefficients decay faster. In fact, if we ignore the difference of graph Laplacians and assume that the initial kernel in our method is given as $K \approx L^{-1}$, we can see that the method in [32] can be regarded as a special case of our method when the decay factor C is set to 1 and the dimension cut-off parameter d is set to n.

4.4 Empirical Observations

To argue that $C = 1$ in the spectral kernel learning algorithm may not be a good choice for learning an effective kernel, we illustrate some empirical examples to justify the motivation of our spectral kernel learning algorithm. One goal of our spectral kernel learning methodology is to attain a fast decay rate of the spectral coefficients of the kernel matrix. Figure 2 illustrates an example of the change of the resulting spectral coefficients using different decay factors in our spectral kernel learning algorithms. From the figure, we can see that the curves with larger decay factors ($C = 2, 3$) have faster decay rates than the original kernel and the one using $C = 1$. Meanwhile, we can see that the cumulative eigen energy score converges to 100% quickly when the number of dimensions is increased. This shows that we may use much small number of eigen-pairs in our semi-supervised kernel learning algorithm for large-scale problems.

To examine more details in the impact of performance with different decay factors, we evaluate the classification

performance of spectral kernel learning methods with different decay factors in Figure 3. In the figure, we compare the performance of different kernels with respect to spectral kernel design methods. We can see that two unsupervised kernels, $K_{\texttt{Trunc}}$ and $K_{\texttt{Cluster}}$, tend to perform better than the original kernel when the dimension is small. But their performances are not very stable when the number of dimensions is increased. For comparison, the spectral kernel learning method achieves very stable and good performance when the decay factor C is larger than 1. When the decay factor is equal to 1, the performance becomes unstable due to the slow decay rates observed from our previous results in Figure 3. This observation matches the theoretical justification [27] that a kernel with good performance usually favors a faster decay rate of spectral coefficients.

Figure 4 and Figure 5 illustrate more empirical examples based on different initial kernels, in which similar results can be observed. Note that our suggested kernel learning method can learn on any valid kernel, and different initial kernels will impact the performance of the resulting spectral kernels. It is usually helpful if the initial kernel is provided with domain knowledge.

5. UNIFIED KERNEL LOGISTIC REGRESSION

In this section, we present a specific paradigm based on the proposed framework of learning unified kernel machines. We assume the underlying probabilistic model of the kernel machine is Kernel Logistic Regression (KLR). Based on the UKM framework, we develop the Unified Kernel Logistic Regression (UKLR) paradigm to tackle classification tasks. Note that our framework is not restricted to the KLR model, but also can be widely extended for many other kernel machines, such as Support Vector Machine (SVM) and Regularized Least-Square (RLS) classifiers.

Similar to other kernel machines, such as SVM, a KLR problem can be formulated in terms of a standard regularized form of *loss+penalty* in the reproducing kernel Hilbert space (RKHS):

$$\min_{f\in\mathcal{H}_K} \frac{1}{l}\sum_{i=1}^{l} \ln(1+e^{-y_i f(\mathbf{x}_i)}) + \frac{\lambda}{2}||f||^2_{\mathcal{H}_K} , \qquad (17)$$

where $\mathcal{H}_K$ is the RKHS by a kernel K and λ is a regularization parameter. By the representer theorem, the optimal $f(\mathbf{x})$ has the form:

$$f(\mathbf{x}) = \sum_{i=1}^{l} \alpha_i K(\mathbf{x}, \mathbf{x}_i) , \qquad (18)$$

where α_i are model parameters. Note that we omit the constant term in $f(x)$ for simplified notations. To solve the KLR model parameters, there are a number of available techniques for effective solutions [29].

When the kernel K and the model parameters α are available, we use the following solution for active learning, which is simple and efficient for large-scale problems. More specifically, we measure the information entropy of each unlabeled data example as follows

$$H(\mathbf{x};\alpha,K) = -\sum_{i=1}^{N_C} p(C_i|\mathbf{x}) log(p(C_i|\mathbf{x})) , \qquad (19)$$

Algorithm: Unified Kernel Logistic Regresssion
Input

- K_0: Initial normalized kernel
- L: Set of labeled data
- U: Set of unlabeled data

Repeat

- Spectral Kernel Learning
  $K \leftarrow$ Spectral_Kernel(K_0, L, U);
- KLR Parameter Estimation
  $\alpha \leftarrow$ KLR_Solver(L, K);
- Convergence Test
  If (*converged*), **Exit** Loop;
- Active Learning
  $\mathbf{x}^* \leftarrow max_{\mathbf{x}\in U} H(\mathbf{x};\alpha,K)$
  $L^* \leftarrow L \cup \{\mathbf{x}^*\}$, $U^* \leftarrow U - \{\mathbf{x}^*\}$

Until converged.
Output

- UKLR = $\mathcal{M}(K,\alpha)$.

Figure 6: The UKLR Algorithm.

where N_C is the number of classes and C_i denotes the i^{th} class and $p(C_i|\mathbf{x})$ is the probability of the data example $\mathbf{x}$ belonging to the i^{th} class which can be naturally obtained by the current KLR model (α, K). The unlabeled data examples with maximum values of entropy will be considered as the most informative data for labeling.

By unifying the spectral kernel learning method proposed in Section 3, we summarize the proposed algorithm of Unified Kernel Logistic Regression (UKLR) in Figure 6. In the algorithm, note that we can usually initialize a kernel by a standard kernel with appropriate parameters determined by cross validation or by a proper design of the initial kernel with domain knowledge.

6. EXPERIMENTAL RESULTS

We discuss our empirical evaluation of the proposed framework and algorithms for classification. We first evaluate the effectiveness of our suggested spectral kernel learning algorithm for learning semi-supervised kernels and then compare the performance of our unified kernel logistic regression paradigm with traditional classification schemes.

6.1 Experimental Testbed and Settings

We use the datasets from UCI machine learning repository[1]. Four datasets are employed in our experiments. Table 1 shows the details of four UCI datasets in our experiments.

For experimental settings, to examine the influences of different training sizes, we test the compared algorithms on four different training set sizes for each of the four UCI datasets. For each given training set size, we conduct 20 random trials in which a labeled set is randomly sampled

[1]www.ics.uci.edu/ mlearn/MLRepository.html

Table 1: List of UCI machine learning datasets.

Dataset	#Instances	#Features	#Classes
Heart	270	13	2
Ionosphere	351	34	2
Sonar	208	60	2
Wine	178	13	3

from the whole dataset and all classes must be present in the sampled labeled set. The rest data examples of the dataset are then used as the testing (unlabeled) data. To train a classifier, we employ the standard KLR model for classification. We choose the bounds on the regularization parameters via cross validation for all compared kernels to avoid an unfair comparison. For multi-class classification, we perform one-against-all binary training and testing and then pick the class with the maximum class probability.

6.2 Semi-Supervised Kernel Learning

In this part, we evaluate the performance of our spectral kernel learning algorithm for learning semi-supervised kernels. We implemented our algorithm by a standard Matlab Quadratic Programming solver (quadprog). The dimension-cut parameter d in our algorithm is simply fixed to 20 without further optimizing. Note that one can easily determine an appropriate value of d by examining the range of the cumulative eigen energy score in order to reduce the computational cost for large-scale problems. The decay factor C is important for our spectral kernel learning algorithm. As we have shown examples before, C must be a positive real value greater than 1. Typically we favor a larger decay factor to achieve better performance. But it must not be set too large since the too large decay factor may result in the overly stringent constraints in the optimization which gives no solutions. In our experiments, C is simply fixed to constant values (greater than 1) for the engaged datasets.

For a comparison, we compare our SKL algorithms with the state-of-the-art semi-supervised kernel learning method by graph Laplacians [32], which is related to a quadratically constrained quadratic program (QCQP). More specifically, we have implemented two graph Laplacians based semi-supervised kernels by order constraints [32]. One is the order-constrained graph kernel (denoted as "Order") and the other is the improved order-constrained graph kernel (denoted as "Imp-Order"), which removes the constraints from constant eigenvectors. To carry a fair comparison, we use the top 20 smallest eigenvalues and eigenvectors from the graph Laplacian which is constructed with 10-NN unweighted graphs. We also include three standard kernels for comparisons.

Table 2 shows the experimental results of the compared kernels (3 standard and 5 semi-supervised kernels) based on KLR classifiers on four UCI datasets with different sizes of labeled data. Each cell in the table has two rows: the upper row shows the average testing set accuracies with standard errors; and the lower row gives the average run time in *seconds* for learning the semi-supervised kernels on a 3GHz desktop computer. We conducted a paired t-test at significance level of 0.05 to assess the statistical significance of the test set accuracy results. From the experimental results, we found that the two order-constrained based graph kernels perform well in the *Ionosphere* and *Wine* datasets, but they do not achieve important improvements on the *Heart* and *Sonar* datasets. Among all the compared kernels, the semi-supervised kernels by our spectral kernel learning algorithms achieve the best performances. The semi-supervised kernel initialized with an RBF kernel outperforms other kernels in most cases. For example, in *Ionosphere* dataset, an RBF kernel with 10 initial training examples only achieves 73.56% test set accuracy, and the SKL algorithm can boost the accuracy significantly to 83.36%. Finally, looking into the time performance, the average run time of our algorithm is less than 10% of the previous QCQP algorithms.

6.3 Unified Kernel Logistic Regression

In this part, we evaluate the performance of our proposed paradigm of unified kernel logistic regression (UKLR). As a comparison, we implement two traditional classification schemes: one is traditional KLR classification scheme that is trained on randomly sampled labeled data, denoted as "KLR+Rand." The other is the active KLR classification scheme that actively selects the most informative examples for labeling, denoted as "KLR+Active." The active learning strategy is based on a simple maximum entropy criteria given in the pervious section. The UKLR scheme is implemented based on the algorithm in Figure 6.

For active learning evaluation, we choose a batch of 10 most informative unlabeled examples for labeling in each trial of evaluations. Table 3 summarizes the experimental results of average test set accuracy performances on four UCI datasets. From the experimental results, we can observe that the active learning classification schemes outperform the randomly sampled classification schemes in most cases. This shows the suggested simple active learning strategy is effectiveness. Further, among all compared schemes, the suggested UKLR solution significantly outperforms other classification approaches in most cases. These results show that the unified scheme is effective and promising to integrate traditional learning methods together in a unified solution.

6.4 Discussions

Although the empirical results have shown that our scheme is promising, some open issues in our solution need be further explored in future work. One problem is to investigate more effective active learning methods in selecting the most informative examples for labeling. One solution to this issue is to employ the batch mode active learning methods that can be more efficient for large-scale classification tasks [11, 23, 24]. Moreover, we will study more effective kernel learning algorithms without the assumption of spectral kernels. Further, we may examine the theoretical analysis of generalization performance of our method [27]. Finally, we may combine some kernel machine speedup techniques to deploy our scheme efficiently for large-scale applications [26].

7. CONCLUSION

This paper presented a novel general framework of learning the Unified Kernel Machines (UKM) for classification. Different from traditional classification schemes, our UKM framework integrates supervised learning, semi-supervised learning, unsupervised kernel design and active learning in a unified solution, making it more effective for classification tasks. For the proposed framework, we focus our attention on tackling a core problem of learning semi-supervised kernels from labeled and unlabeled data. We proposed a Spec-

Table 2: Classification performance of different kernels using KLR classifiers on four datasets. The mean accuracies and standard errors are shown in the table. 3 standard kernels and 5 semi-supervised kernels are compared. Each cell in the table has two rows. The upper row shows the test set accuracy with standard error; the lower row gives the average time used in learning the semi-supervised kernels ("Order" and "Imp-Order" kernels are sovled by SeDuMi/YALMIP package; "SKL" kernels are solved directly by the Matlab quadprog function.

Train Size	Standard Kernels			Semi-Supervised Kernels				
	Linear	Quadratic	RBF	Order	Imp-Order	SKL(Linear)	SKL(Quad)	SKL(RBF)
Heart								
10	67.19 ± 1.94	71.90 ± 1.23	70.04 ± 1.61	63.60 ± 1.94	63.60 ± 1.94	70.58 ± 1.63	72.33 ± 1.60	**73.37 ± 1.50**
	—	—	—	(0.67)	(0.81)	(0.07)	(0.06)	(0.06)
20	67.40 ± 1.87	70.36 ± 1.51	72.64 ± 1.37	65.88 ± 1.69	65.88 ± 1.69	76.26 ± 1.29	75.36 ± 1.30	**76.30 ± 1.33**
	—	—	—	(0.71)	(0.81)	(0.06)	(0.06)	(0.06)
30	75.42 ± 0.88	70.71 ± 0.83	74.40 ± 0.70	71.73 ± 1.14	71.73 ± 1.14	78.42 ± 0.59	78.65 ± 0.52	**79.23 ± 0.58**
	—	—	—	(0.95)	(0.97)	(0.06)	(0.06)	(0.06)
40	78.24 ± 0.89	71.28 ± 1.10	78.48 ± 0.77	75.48 ± 0.69	75.48 ± 0.69	80.61 ± 0.45	80.26 ± 0.45	**80.98 ± 0.51**
	—	—	—	(1.35)	(1.34)	(0.07)	(0.07)	(0.07)
Ionosphere								
10	73.71 ± 1.27	71.30 ± 1.70	73.56 ± 1.91	71.86 ± 2.79	71.86 ± 2.79	75.53 ± 1.75	69.25 ± 1.67	**83.36 ± 1.31**
	—	—	—	(0.90)	(0.87)	(0.05)	(0.05)	(0.05)
20	75.62 ± 1.24	76.00 ± 1.58	81.71 ± 1.74	83.04 ± 2.10	83.04 ± 2.10	78.78 ± 1.60	80.30 ± 1.77	**88.55 ± 1.32**
	—	—	—	(0.87)	(0.79)	(0.05)	(0.06)	(0.05)
30	76.59 ± 0.82	79.10 ± 1.46	86.21 ± 0.84	87.20 ± 1.16	87.20 ± 1.16	82.18 ± 0.56	83.08 ± 1.36	**90.39 ± 0.84**
	—	—	—	(0.93)	(0.97)	(0.05)	(0.05)	(0.05)
40	77.97 ± 0.79	82.93 ± 1.33	89.39 ± 0.65	90.56 ± 0.64	90.56 ± 0.64	83.26 ± 0.53	87.03 ± 1.02	**92.14 ± 0.46**
	—	—	—	(1.34)	(1.38)	(0.05)	(0.04)	(0.04)
Sonar								
10	63.01 ± 1.47	62.85 ± 1.53	60.76 ± 1.80	59.67 ± 0.89	59.67 ± 0.89	64.27 ± 1.91	64.37 ± 1.64	**65.30 ± 1.78**
	—	—	—	(0.63)	(0.63)	(0.08)	(0.07)	(0.07)
20	68.09 ± 1.11	69.55 ± 1.22	67.63 ± 1.15	64.68 ± 1.57	64.68 ± 1.57	70.61 ± 1.14	69.79 ± 1.30	**71.76 ± 1.07**
	—	—	—	(0.68)	(0.82)	(0.07)	(0.07)	(0.08)
30	66.40 ± 1.06	69.80 ± 0.93	68.23 ± 1.48	66.54 ± 0.79	66.54 ± 0.79	70.20 ± 1.48	68.48 ± 1.59	**71.69 ± 0.87**
	—	—	—	(0.88)	(1.02)	(0.07)	(0.07)	(0.07)
40	64.94 ± 0.74	71.37 ± 0.52	71.61 ± 0.89	69.82 ± 0.82	69.82 ± 0.82	72.35 ± 1.06	71.28 ± 0.96	**72.89 ± 0.68**
	—	—	—	(1.14)	(1.20)	(0.07)	(0.08)	(0.07)
Wine								
10	82.26 ± 2.18	85.89 ± 1.73	87.80 ± 1.63	87.44 ± 2.21	87.44 ± 2.21	86.49 ± 2.48	86.55 ± 2.40	**93.72 ± 0.65**
	—	—	—	(1.02)	(0.86)	(0.09)	(0.09)	(0.09)
20	86.39 ± 1.39	86.96 ± 1.30	93.77 ± 0.99	92.72 ± 1.32	92.72 ± 1.32	88.86 ± 3.31	93.39 ± 0.59	**95.63 ± 0.45**
	—	—	—	(0.92)	(0.91)	(0.09)	(0.09)	(0.09)
30	92.50 ± 0.76	87.43 ± 0.63	94.63 ± 0.50	93.99 ± 0.53	93.99 ± 0.53	93.99 ± 1.55	94.63 ± 0.50	**96.32 ± 0.33**
	—	—	—	(1.28)	(1.27)	(0.09)	(0.10)	(0.09)
40	94.96 ± 0.65	88.80 ± 0.93	96.38 ± 0.35	96.34 ± 0.33	96.34 ± 0.33	95.98 ± 0.41	95.25 ± 0.47	**96.74 ± 0.27**
	—	—	—	(1.41)	(1.39)	(0.08)	(0.08)	(0.10)

tral Kernel Learning (SKL) algorithm, which is more effective and efficient for learning kernels from labeled and unlabeled data. Under the framework, we developed a paradigm of unified kernel machine based on Kernel Logistic Regression, i.e., Unified Kernel Logistic Regression (UKLR). Empirical results demonstrated that our proposed solution is more effective than the traditional classification approaches.

8. ACKNOWLEDGMENTS

The work described in this paper was fully supported by two grants, one from the Shun Hing Institute of Advanced Engineering, and the other from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. CUHK4205/04E).

9. REFERENCES

[1] M. Belkin and I. M. andd P. Niyogi. Regularization and semi-supervised learning on large graphs. In *COLT*, 2004.

[2] M. Belkin and P. Niyogi. Semi-supervised learning on riemannian manifolds. *Machine Learning*, 2004.

[3] E. Chang, S. C. Hoi, X. Wang, W.-Y. Ma, and M. Lyu. A unified machine learning framework for large-scale personalized information management. In *The 5th Emerging Information Technology Conference*, NTU Taipei, 2005.

[4] E. Chang and M. Lyu. Unified learning paradigm for web-scale mining. In *Snowbird Machine Learning Workshop*, 2006.

[5] O. Chapelle, A. Zien, and B. Scholkopf. *Semi-supervised learning*. MIT Press, 2006.

[6] F. R. K. Chung. *Spectral Graph Theory*. American Mathematical Soceity, 1997.

[7] D. A. Cohn, Z. Ghahramani, and M. I. Jordan. Active learning with statistical models. In *NIPS*, volume 7, pages 705–712, 1995.

[8] N. Cristianini, J. Shawe-Taylor, and A. Elisseeff. On kernel-target alignment. *JMLR*, 2002.

Table 3: Classification performance of different classification schemes on four UCI datasets. The mean accuracies and standard errors are shown in the table. "KLR" represents the initial classifier with the initial train size; other three methods are trained with additional 10 random/active examples.

Train Size	Linear Kernel				RBF Kernel			
	KLR	KLR+Rand	KLR+Active	UKLR	KLR	KLR+Rand	KLR+Active	UKLR
Heart								
10	67.19 ± 1.94	68.22 ± 2.16	69.22 ± 1.71	**77.24** ± 0.74	70.04 ± 1.61	72.24 ± 1.23	75.36 ± 0.60	**78.44** ± 0.88
20	67.40 ± 1.87	73.79 ± 1.29	73.77 ± 1.27	**79.27** ± 1.00	72.64 ± 1.37	75.10 ± 0.74	76.23 ± 0.81	**79.88** ± 0.90
30	75.42 ± 0.88	77.70 ± 0.92	78.65 ± 0.62	**81.13** ± 0.42	74.40 ± 0.70	76.43 ± 0.68	76.61 ± 0.61	**81.48** ± 0.41
40	78.24 ± 0.89	79.30 ± 0.75	80.18 ± 0.79	**82.55** ± 0.28	78.48 ± 0.77	78.50 ± 0.53	79.95 ± 0.62	**82.66** ± 0.36
Ionosphere								
10	73.71 ± 1.27	74.89 ± 0.95	75.91 ± 0.96	**77.31** ± 1.23	73.56 ± 1.91	82.57 ± 1.78	82.76 ± 1.37	**90.48** ± 0.83
20	75.62 ± 1.24	77.09 ± 0.67	77.51 ± 0.66	**81.42** ± 1.10	81.71 ± 1.74	85.95 ± 1.30	88.22 ± 0.78	**91.28** ± 0.94
30	76.59 ± 0.82	78.41 ± 0.79	77.91 ± 0.77	**84.49** ± 0.37	86.21 ± 0.84	89.04 ± 0.66	90.32 ± 0.56	**92.35** ± 0.59
40	77.97 ± 0.79	79.05 ± 0.49	80.30 ± 0.79	**84.49** ± 0.40	89.39 ± 0.65	90.55 ± 0.59	91.83 ± 0.49	**93.89** ± 0.28
Sonar								
10	61.19 ± 1.56	63.72 ± 1.65	65.51 ± 1.55	**66.12** ± 1.94	57.40 ± 1.48	60.19 ± 1.32	59.49 ± 1.46	**67.13** ± 1.58
20	67.31 ± 1.07	68.85 ± 0.84	69.38 ± 1.05	**71.60** ± 0.91	62.93 ± 1.36	64.72 ± 1.24	64.52 ± 1.07	**72.30** ± 0.98
30	66.10 ± 1.08	67.59 ± 1.14	69.79 ± 0.86	**71.40** ± 0.80	63.03 ± 1.32	63.72 ± 1.51	66.67 ± 1.53	**72.26** ± 0.98
40	66.34 ± 0.82	68.16 ± 0.81	70.19 ± 0.90	**73.04** ± 0.69	66.70 ± 1.25	68.70 ± 1.19	67.56 ± 0.90	**73.16** ± 0.88
Wine								
10	82.26 ± 2.18	87.31 ± 1.01	**89.05 ± 1.07**	87.31 ± 1.03	87.80 ± 1.63	92.75 ± 1.27	94.49 ± 0.54	**94.87** ± 0.49
20	86.39 ± 1.39	93.99 ± 0.40	93.82 ± 0.71	**94.43** ± 0.54	93.77 ± 0.99	95.57 ± 0.38	**97.13 ± 0.18**	96.76 ± 0.26
30	92.50 ± 0.76	95.25 ± 0.47	**96.96 ± 0.40**	96.12 ± 0.47	94.63 ± 0.50	96.27 ± 0.35	97.17 ± 0.38	**97.21** ± 0.26
40	94.96 ± 0.65	96.21 ± 0.63	97.54 ± 0.37	**97.70** ± 0.34	96.38 ± 0.35	96.33 ± 0.45	97.97 ± 0.23	**98.12** ± 0.21

[9] S. Fine, R. Gilad-Bachrach, and E. Shamir. Query by committee, linear separation and random walks. *Theor. Comput. Sci.*, 284(1):25–51, 2002.

[10] Y. Freund, H. S. Seung, E. Shamir, and N. Tishby. Selective sampling using the query by committee algorithm. *Mach. Learn.*, 28(2-3):133–168, 1997.

[11] S. C. Hoi, R. Jin, and M. R. Lyu. Large-scale text categorization by batch mode active learning. In *WWW2006*, Edinburg, 2006.

[12] S. B. C. M. J.A.K. Suykens, G. Horvath and J. Vandewalle. *Advances in Learning Theory: Methods, Models and Applications*. NATO Science Series: Computer & Systems Sciences, 2003.

[13] R. Kondor and J. Lafferty. Diffusion kernels on graphs and other discrete structures. 2002.

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

[15] G. Lanckriet, L. Ghaoui, C. Bhattacharyya, and M. Jordan. Minimax probability machine. In *Advances in Neural Infonation Processing Systems 14*, 2002.

[16] R. Liere and P. Tadepalli. Active learning with committees for text categorization. In *Proceedings 14th Conference of the American Association for Artificial Intelligence (AAAI)*, pages 591–596, MIT Press, 1997.

[17] R. Meir and G. Ratsch. An introduction to boosting and leveraging. In *In Advanced Lectures on Machine Learning (LNAI2600)*, 2003.

[18] A. Ng, M. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *In Advances in Neural Information Processing Systems 14*, 2001.

[19] N. Roy and A. McCallum. Toward optimal active learning through sampling estimation of error reduction. In *18th ICML*, pages 441–448, 2001.

[20] B. Scholkopf, A. Smola, and K.-R. Muller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10:1299–1319, 1998.

[21] A. Smola and R. Kondor. Kernels and regularization on graphs. In *Intl. Conf. on Learning Theory*, 2003.

[22] M. Szummer and T. Jaakkola. Partially labeled classification with markov random walks. In *Advances in Neural Information Processing Systems*, 2001.

[23] S. Tong and E. Chang. Support vector machine active learning for image retrieval. In *Proc ACM Multimedia Conference*, pages 107–118, New York, 2001.

[24] S. Tong and D. Koller. Support vector machine active learning with applications to text classification. In *Proc. 17th ICML*, pages 999–1006, 2000.

[25] V. N. Vapnik. *Statistical Learning Theory*. John Wiley & Sons, 1998.

[26] G. Wu, Z. Zhang, and E. Y. Chang. Kronecker factorization for speeding up kernel machines. In *SIAM Int. Conference on Data Mining (SDM)*, 2005.

[27] T. Zhang and R. K. Ando. Analysis of spectral kernel design based semi-supervised learning. In *NIPS*, 2005.

[28] D. Zhou, O. Bousquet, T. Lal, J. Weston, and B. Schlkopf. Learning with local and global consistency. In *NIPS'16*, 2005.

[29] J. Zhu and T. Hastie. Kernel logistic regression and the import vector machine. In *NIPS 14*, pages 1081–1088, 2001.

[30] X. Zhu. Semi-supervised learning literature survey. Technical report, Computer Sciences TR 1530, University of Wisconsin - Madison, 2005.

[31] X. Zhu, Z. Ghahramani, and J. Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *Proc. ICML'2003*, 2003.

[32] X. Zhu, J. Kandola, Z. Ghahramani, and J. Lafferty. Nonparametric transforms of graph kernels for semi-supervised learning. In *NIPS2005*, 2005.