Probabilistic Kernel Matrix Learning with a Mixture Model of Kernels

Zhihua Zhang, Dit-Yan Yeung, and James T. Kwok

Department of Computer Science

Hong Kong University of Science and Technology

Clear Water Bay, Kowloon, Hong Kong

{zhzhang,dyyeung,jamesk}@cs.ust.hk

Abstract

This paper addresses the kernel matrix learning problem in kernel methods. We model the kernel matrix as a random positive definite matrix following the Wishart distribution, with the parameter matrix of the Wishart distribution represented as a linear combination of mutually independent matrices with their own Wishart distributions. This defines a probabilistic mixture model of kernels that can be represented as a hierarchical model involving three levels, relating the target kernel matrix, the parameter kernel matrix, and the hyperparameter kernel matrices of the mixture components. Since in most cases a linear combination of Wishart matrices is no longer Wishart, we propose an approximation method by employing a new Wishart matrix to approximate the parameter matrix through preserving the first and second moments of the linear combination of Wishart matrices. Based on the Wishart matrix estimated, kernel matrix learning is then solved by an expectation-maximization (EM) learning algorithm to infer the missing data of the kernel matrix and the unknown parameter matrix of the distribution. Moreover, we study the kernel matrix learning problem in the context of classification problems with the use of a kernel nearest neighbor classifier. Classification experiments on several benchmark data sets show promising results. Furthermore, our method has opened up a possible direction for addressing the kernel

model selection and kernel parameter estimation problems.

Keywords: Probabilistic kernel matrix learning; Mixture model of kernels; Wishart distribution; EM algorithm; Kernel classifier

Corresponding author: Zhihua Zhang (Fax: +852-2358-1477, Email: zhzhang@cs.ust.hk)

Contents

1 Introduction

1.1 Kernel Methods

Over the last few years, kernel methods [23, 27] have been increasingly popular in machine learning due to their conceptual simplicity and strong theoretical foundations. Kernel-based learning models and algorithms, such as support vector machines (SVM) [6, 27], kernel principal component analysis (PCA) [22] and kernel Fisher discriminant analysis (FDA) [3], work by nonlinearly mapping data from the input space $\mathcal{X}$ to a higher-dimensional feature space $\mathcal{F}$ as defined by a *Mercer kernel*, where relatively simple algorithms such as many traditional linear algorithms [10] can be applied. Through a nonlinear mapping, simple linear operations in the feature space correspond to powerful, complex nonlinear operations in the input space. In the following, let $\varphi : \mathcal{X} \to \mathcal{F}$ be the mapping that relates $\mathcal{X}$ to $\mathcal{F}$. The kernel function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is then a mapping such that

$$k(\mathbf{x}_i, \mathbf{x}_j) = \varphi(\mathbf{x}_i)'\varphi(\mathbf{x}_j),$$

the inner product of $\varphi(\mathbf{x}_i)$ and $\varphi(\mathbf{x}_j)$ in $\mathcal{F}$. In other words, instead of explicitly carrying out the nonlinear mapping φ in the possibly infinite-dimensional $\mathcal{F}$, this so-called *kernel trick* allows inner products in $\mathcal{F}$ to be computed based on the input vectors in $\mathcal{X}$ only. Let $\mathcal{I} = \{\mathbf{x}_i\}_{i=1}^m \subset \mathcal{X}$ denote a finite set of input vectors, we can define a positive semi-definite kernel matrix (or the *Gram matrix*) $\mathbf{K} = (k_{ij})_{m\times m}$, where $k_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$.

1.2 Kernel Learning

Since the kernel plays a central role in kernel methods, a poor kernel choice can significantly degrade the performance. Most existing kernel methods require that an appropriate kernel be manually selected in advance. A better approach requires only the parametric form of the kernel function be specified, leaving the kernel parameters adjustable in the course of the learning process (e.g., [5, 16]). Examples of commonly used parametric kernels include the polynomial and Gaussian kernel, with the polynomial order and Gaussian width being the corresponding kernel parameters. Typically, adaptation of these kernel parameters is performed by optimizing a certain *quality functional* [21], such as some generalization error bound.

Instead of adapting only the kernel parameters, a recent development allows adapting the form of the kernel itself. In a transductive setting, one can needs to consider the kernel evaluations on the given set of training and testing data only, and the kernel learning problem is thus simplified to the learning of just the kernel matrix. By introducing the notion of *alignment*, Cristianini *et al.* [8] proposed a kernel matrix learning method that optimizes the coefficients in the spectral decomposition of the full kernel matrix on both training and test data. However, there is no direct connection between the alignment and the generalization error. Lanckriet *et al.* [17] derived a generalization bound for choosing the kernel and formulated the kernel matrix learning problem as a convex optimization problem that is not prone to local minima. Nevertheless, even with the recent advances in interior point methods, solving convex programming problems such as a semi-definite programming (SDP) problem are still very computationally expensive on large data sets. Thus, instead of using SDP, Bousquet and Herrmann [4] proposed a simple, efficient gradient-descent algorithm that can be orders of magnitude faster than a typical SDP solver. Besides, Crammer *et al.* [7] also proposed another kernel matrix learning method based on boosting, in which an accurate kernel is constructed from simple base kernels through solving the generalized eigenvector problem. Tsuda *et al.* [26] considered the kernel matrix completion problem as a missing data problem and developed a parametric approach by using the *em* algorithm [1] based on the information geometry of positive definite matrices. Recently, Kandola *et al.* [13, 14] extended the alignment optimization method from the transductive setting to the inductive setting.

In our recent work [29], we proposed a probabilistic approach to the kernel matrix learning problem by devising a probabilistic generative model for the kernel matrix as a random positive definite matrix following the Wishart distribution [11]. The kernel matrix learning problem is then reduced to a missing data problem under the *maximum a posteriori* (MAP) framework. Given the kernel matrix on the training data, we devised an expectation-maximization (EM) algorithm [9] for inferring the kernel matrix on the test data, the kernel matrix relating the training data with the test data, and the parameter matrix of the Wishart distribution. An alternative approach based on the Tanner-Wong algorithm [25] (which is a Markov chain Monte Carlo (MCMC) method) is also proposed in [28]. Both [29] and [28] thus have strong Bayesian flavors.

Besides learning the kernel matrix, another approach is to formulate the problem as learning the kernel function directly. Based on the theory of reproducing kernels [2], Ong *et al.* [20, 21] introduced superkernels (later renamed to hyperkernels) by defining the space of kernels as yet another reproducing kernel Hilbert space. The kernel learning problem is then reduced to an optimization problem. Their method can work in the inductive setting.

1.3 Research Agenda of this Paper

Many existing kernel matrix learning methods [4, 7, 8, 17, 26] constrain the target kernel to a weighted combination of some fixed base kernels. The learning problem can thus be simplified to the estimation of the weighting coefficients only. Motivated by this, we propose in this paper an extension of our probabilistic approach [29] by incorporating a kernel mixture model (Section 4). Unlike other kernel learning methods, this is based on a probabilistic formulation. Specifically, we consider the parameter matrix of the Wishart distribution as a linear combination of mutually independent matrices with their own Wishart distributions. In most cases, however, a linear combination of Wishart matrices is no longer Wishart. This makes it intractable to obtain an analytical solution for the kernel mixture model. Inspired by previous works [15, 24], we use a new Wishart matrix to approximate this linear combination of Wishart matrices. Since the new Wishart matrix preserves the first and second moments of the linear combination, the approximation is optimal in the information-theoretic sense.

On applying our method to the classification problem (Section 5), it is worth noting that kernel learning forms an integral part of the entire classifier training process. On the contrary, traditional kernel-based classification methods, such as SVM [6, 27] and kernel FDA [3], treat the model selection (i.e., kernel selection) problem and the classifier training problem separately as two successive processes. They typically start by selecting an appropriate kernel empirically or learning a target kernel adaptively. A classifier is then built using the selected or learned kernel. From the aspects of both computational cost and classification accuracy, one wants to deal with these two processes jointly in a single paradigm. This paper attempts to address this issue. In the experiments, we will study two different settings of the kernel mixture. In the

first setting, different components of the mixture correspond to positive definite matrices induced by kernel functions of different forms. In the second setting, all components correspond to kernel functions of the same form but with different parameters. We will see that the first setting can be regarded as *kernel model selection* while the second setting can be regarded as *kernel parameter estimation*.

The rest of this paper is organized as follows. Some relevant mathematical background is first introduced in Section 2. The probabilistic kernel matrix learning approach proposed in [29] is then briefly reviewed in Section 3. Section 4 presents an extension of this work based on the development of a kernel mixture model, and Section 5 then applies it in the context of classification problems. Experimental results are presented in Section 6, and the last section contains some concluding remarks.

2 Mathematical Background

To make this paper self-contained, we will first briefly review some relevant terminologies and results from matrix algebra and matrix variate distributions. For more detailed discussions, the readers are referred to [12, chapter 4] and [11, chapter 3].

Throughout this paper, we denote a matrix and a vector by the boldface uppercase letter and the boldface lowercase letter, respectively. Let $\mathbf{A}$ be any matrix. We denote the transpose of $\mathbf{A}$ by $\mathbf{A}'$, the trace of $\mathbf{A}$ by $\mathrm{tr}(\mathbf{A})$, the determinant of $\mathbf{A}$ by $|\mathbf{A}|$, and the inverse of $\mathbf{A}$ (if it exists) by $\mathbf{A}^{-1}$. In addition, we write $\mathbf{A} \succ 0$ if $\mathbf{A}$ is positive definite and $\mathbf{A} \succeq 0$ if $\mathbf{A}$ is positive semi-definite.

2.1 Kronecker Product

Definition 1 *Let* $\mathbf{A} = (a_{ij})$ *be a* $p \times q$ *matrix and* $\mathbf{B} = (b_{ij})$ *be a* $s \times t$ *matrix. The* Kronecker product *of* $\mathbf{A}$ *and* $\mathbf{B}$*, denoted by* $\mathbf{A} \otimes \mathbf{B}$*, is the* $ps \times qt$ *matrix defined by*

$$\mathbf{A} \otimes \mathbf{B} = \begin{bmatrix} a_{11}\mathbf{B} & a_{12}\mathbf{B} & \dots & a_{1q}\mathbf{B} \\ a_{21}\mathbf{B} & a_{22}\mathbf{B} & \dots & a_{2q}\mathbf{B} \\ \vdots & \vdots & \ddots & \vdots \\ a_{p1}\mathbf{B} & a_{p2}\mathbf{B} & \dots & a_{pq}\mathbf{B} \end{bmatrix} = (a_{ij}\mathbf{B}).$$

Definition 2 *If* $\mathbf{Y} = (y_{ij})$ *is a* $p \times q$ *matrix, then* $\mathrm{vec}(\mathbf{Y})$ *is the* $pq \times 1$ *vector defined as*

$$\mathrm{vec}(\mathbf{Y}) = (y_{11}, y_{21}, \dots, y_{p1}, y_{12}, y_{22}, \dots, y_{p2}, \dots, y_{1q}, y_{2q}, \dots, y_{pq})',$$

which is formed by stacking the columns of $\mathbf{Y}$ *to form a column vector. If* $\mathbf{Y} = (y_{ij})$ *is a* $p \times p$ *symmetric matrix, then* $\mathrm{vecp}(\mathbf{Y})$ *is the* $\frac{1}{2}p(p+1)$*-dimensional column vector defined as*

$$\mathrm{vecp}(\mathbf{Y}) = (y_{11}, y_{12}, y_{22}, \dots, y_{1p}, y_{2p}, \dots, y_{pp})',$$

which is formed from the elements above and including the diagonal entries, taken columnwise.

Definition 3 *The* permutation matrix *(also known as the* commutation matrix*)* $\mathbf{H}_{st}$ *of order* $st \times st$ *is defined as*

$$\mathbf{H}_{st} = \sum_{i=1}^{s} \sum_{j=1}^{t} \mathbf{E}_{ij} \otimes \mathbf{E}_{ij}',$$

where $\mathbf{E}_{ij}$ *denotes the* $s \times t$ *matrix with a unit element at the* $(i,j)^{\text{th}}$ *entry and zero elsewhere.*

Some important properties of the Kronecker product are summarized below.

Proposition 1

(a) $(\mathbf{A} \otimes \mathbf{B})' = \mathbf{A}' \otimes \mathbf{B}'$.

(b) *If* $\mathbf{A}$ *is* $k \times l$*,* $\mathbf{B}$ *is* $p \times q$*,* $\mathbf{X}$ *is* $l \times s$*, and* $\mathbf{Y}$ *is* $q \times t$*, then*

$$(\mathbf{A} \otimes \mathbf{B})(\mathbf{X} \otimes \mathbf{Y}) = \mathbf{AX} \otimes \mathbf{BY}.$$

(c) *If* $\mathbf{A}$ *and* $\mathbf{B}$ *are* $k \times l$ *and* $\mathbf{X}$ *and* $\mathbf{Y}$ *are* $p \times q$*, then*

$$(\mathbf{A} + \mathbf{B}) \otimes (\mathbf{X} + \mathbf{Y}) = \mathbf{A} \otimes \mathbf{X} + \mathbf{A} \otimes \mathbf{Y} + \mathbf{B} \otimes \mathbf{X} + \mathbf{B} \otimes \mathbf{Y}.$$

(d) *If* $\mathbf{A}$ *and* $\mathbf{B}$ *are* $m \times m$*, then*

$$\begin{aligned} \operatorname{tr}(\mathbf{A} \otimes \mathbf{B}) &= \operatorname{tr}(\mathbf{A}) \operatorname{tr}(\mathbf{B}), \\ |\mathbf{A} \otimes \mathbf{B}| &= |\mathbf{A}|^m |\mathbf{B}|^m. \end{aligned}$$

Moreover, if $\mathbf{A}$ *and* $\mathbf{B}$ *are nonsingular, then*

$$(\mathbf{A} \otimes \mathbf{B})^{-1} = \mathbf{A}^{-1} \otimes \mathbf{B}^{-1}.$$

If $\mathbf{A}$ *and* $\mathbf{B}$ *are positive (semi-)definite, then so is* $\mathbf{A} \otimes \mathbf{B}$*.*

Proposition 2 *If* $\mathbf{A}$ *is* $s \times s$ *and* $\mathbf{B}$ *is* $t \times t$*, then*

$$\operatorname{tr}\{\mathbf{H}_{st} (\mathbf{A}' \otimes \mathbf{B})\} = \operatorname{tr}(\mathbf{A}'\mathbf{B}).$$

2.2 Wishart Distribution

Definition 4 *A* $m \times m$ *random symmetric positive definite matrix* $\mathbf{W} \succ 0$ *is said to be distributed as a* Wishart distribution*, denoted as* $\mathbf{W} \sim \mathcal{W}_m(r, \mathbf{\Sigma})$*, with parameters* r *and* $\mathbf{\Sigma}$*, if its density function is given by*

$$p(\mathbf{W} \mid \mathbf{\Sigma}, r) = \frac{1}{C(m,r)} |\mathbf{\Sigma}|^{-r/2} |\mathbf{W}|^{(r-m-1)/2} \exp\left(-\frac{1}{2}\operatorname{tr}(\mathbf{\Sigma}^{-1}\mathbf{W})\right), \tag{1}$$

where $r \geq m$ *is the degree of freedom,* $\mathbf{\Sigma} \succ 0$ *is a* $m \times m$ *positive definite parameter matrix, and* $C(m,r) = 2^{rm/2} \pi^{m(m-1)/4} \prod_{j=1}^{m} \Gamma(\frac{r+1-j}{2})$ *is a normalization term with* $\Gamma(\cdot)$ *denoting the Gamma function.*

Definition 5 *A* $m \times m$ *random symmetric positive definite matrix* $\mathbf{X} \succ 0$ *is said to be distributed as an* inverted Wishart distribution*, denoted as* $\mathbf{X} \sim \mathcal{IW}_m(r, \mathbf{\Theta})$*, with parameters* r *and* $\mathbf{\Theta}$*, if its density function is given by*

$$p(\mathbf{X} \mid \mathbf{\Theta}, r) = \frac{1}{C(m,r)} |\mathbf{\Theta}|^{r/2} |\mathbf{X}|^{-(r+m+1)/2} \exp\left(-\frac{1}{2}\operatorname{tr}(\mathbf{\Theta}\,\mathbf{X}^{-1})\right). \tag{2}$$

Proposition 3 *If* $\mathbf{W} \sim \mathcal{W}_m(r, \mathbf{\Sigma})$, *then*

$$\begin{aligned} \mathrm{E}(\mathbf{W}) &= r\mathbf{\Sigma}, \\ \mathrm{Cov}(\mathrm{vec}(\mathbf{W})) &= r(\mathbf{I} + \mathbf{H}_{m^2})(\mathbf{\Sigma} \otimes \mathbf{\Sigma}). \end{aligned}$$

Here $\mathrm{E}(\cdot)$ *denotes the expectation,* $\mathrm{Cov}(\cdot)$ *the covariance matrix,* $\mathbf{I}$ *the identity matrix, and* $\mathbf{H}_{m^2}$ *the* $m^2 \times m^2$ *permutation matrix.*

Theorem 1 *If* $\mathbf{W} \sim \mathcal{W}_m(r, \mathbf{\Sigma})$ *and* $\mathbf{A}$ *is any* $m \times m$ *nonsingular matrix, then*

$$\mathbf{AWA}' \sim \mathcal{W}_m(r, \mathbf{A\Sigma A}').$$

Theorem 2 *If* $\mathbf{W}_1, \mathbf{W}_2, \ldots, \mathbf{W}_K$ *are* $K > 1$ *independent Wishart matrices such that* $\mathbf{W}_k \sim \mathcal{W}_m(r_k, \mathbf{\Sigma})$ *for* $1 \leq k \leq K$, *then*

$$\sum_{k=1}^{K} \mathbf{W}_k \sim \mathcal{W}_m(\sum_{k=1}^{K} r_k, \mathbf{\Sigma}).$$

Theorem 3 *Let* $\mathbf{W} \sim \mathcal{W}_m(r, \mathbf{\Sigma})$. *Then* $\mathbf{W}^{-1} \sim \mathcal{I}W_m(r, \mathbf{\Sigma}^{-1})$.

3 Learning a Generative Model of the Kernel Matrix

In this Section, we briefly review the probabilistic approach on kernel matrix learning proposed in [29]. As in other kernel matrix learning methods [4, 7, 8, 17, 26], we will also focus on the transductive setting.

3.1 Probabilistic Generative Model

Denote the input parts of the training and test sets by $\mathcal{I} = \{\mathbf{x}_i\}_{i=1}^{n_1}$ with n_1 input vectors and $\tilde{\mathcal{I}} = \{\tilde{\mathbf{x}}_i\}_{i=1}^{n_2}$ with n_2 input vectors, respectively. For the classification problem, the outputs corresponding to the $\mathbf{x}_i$'s are also given but the outputs corresponding to the $\tilde{\mathbf{x}}_i$'s are unknown. A kernel matrix $\mathbf{K}$ defined on $\mathcal{I} \cup \tilde{\mathcal{I}}$ can be partitioned as

$$\mathbf{K} = \begin{bmatrix} \mathbf{K}_{11} & \mathbf{K}_{12} \\ \mathbf{K}_{21} & \mathbf{K}_{22} \end{bmatrix}, \tag{3}$$

where $\mathbf{K}_{21} = \mathbf{K}_{12}'$ is a $n_2 \times n_1$ matrix relating the training data with the test data, and $\mathbf{K}_{11}, \mathbf{K}_{22}$ are $n_1 \times n_1$ and $n_2 \times n_2$ kernel matrices defined on the training and test sets, respectively. The problem of interest is thus to infer $\mathbf{K}_{21}$ and $\mathbf{K}_{22}$ from $\mathbf{K}_{11}$.

A probabilistic formulation is adopted in [29]. In particular, $\mathbf{K}$ is assumed to be a random Wishart matrix distributed as

$$\mathbf{K} \sim \mathcal{W}_n(\rho, \mathbf{\Sigma}), \tag{4}$$

where $n = n_1 + n_2$ and $\rho \geq n$ is the user-defined degree of freedom. The prior distribution of the parameter matrix $\mathbf{\Sigma}$ is assumed to be an inverted Wishart distribution $\mathcal{IW}_n(\eta, \mathbf{\Theta})$. Equivalently, Theorem 3 tells us that $\mathbf{C} = \mathbf{\Sigma}^{-1}$ is distributed according to $\mathcal{W}_n(\eta, \mathbf{\Theta}^{-1})$.

3.2 EM Algorithm for Kernel Matrix Learning

With the prior distribution of the parameter matrix $\mathbf{\Sigma}$ available, kernel matrix learning can be formulated as a missing data problem that can be solved by finding the MAP estimate [9, 18]. In [29], we devised such an EM algorithm for inferring the missing parts of the kernel matrix ($\mathbf{K}_{21}$ and $\mathbf{K}_{22}$) as well as the unknown parameter matrix $\mathbf{\Sigma}$.

First, we partition $\mathbf{\Sigma}^{-1}$ and $\mathbf{\Theta}$ in a similar way as $\mathbf{K}$ in (3), and obtain

$$\mathbf{\Sigma}^{-1} = \mathbf{C} = \begin{bmatrix} \mathbf{C}_{11} & \mathbf{C}_{12} \\ \mathbf{C}_{21} & \mathbf{C}_{22} \end{bmatrix}, \quad \mathbf{\Theta} = \begin{bmatrix} \mathbf{\Theta}_{11} & \mathbf{\Theta}_{12} \\ \mathbf{\Theta}_{21} & \mathbf{\Theta}_{22} \end{bmatrix}.$$

Let $\mathbf{K}_{22\cdot 1} = \mathbf{K}_{22} - \mathbf{K}_{21}\mathbf{K}_{11}^{-1}\mathbf{K}_{12}$ and $\mathbf{C}_{11\cdot 2} = \mathbf{C}_{11} - \mathbf{C}_{12}\mathbf{C}_{22}^{-1}\mathbf{C}_{21}$ be the Schur complements of $\mathbf{K}_{22}$ and $\mathbf{C}_{11}$, respectively. Given the t^{th} estimate $\mathbf{C}^{(t)}$ of the parameter matrix $\mathbf{C}$, the E-step of the EM algorithm seeks to obtain the t^{th} estimate of the missing data of $\mathbf{K}$ as

$$\begin{aligned} \mathbf{K}_{22\cdot 1}^{(t)} &= (\rho - n_1)(\mathbf{C}_{22}^{-1})^{(t)}, \\ \mathbf{K}_{21}^{(t)} &= -(\mathbf{C}_{22}^{-1})^{(t)}\mathbf{C}_{21}^{(t)}\mathbf{K}_{11}. \end{aligned} \tag{5}$$

On the other hand, the M-step seeks to obtain the $(t+1)^{\text{th}}$ estimate of the parameter matrix $\mathbf{C}$ from the

t^{th} estimate of the missing data as

$$\begin{aligned}
\mathbf{C}_{11\cdot 2}^{(t+1)} &= (\rho+\eta-n-1)(\mathbf{K}_{11}+\mathbf{\Theta}_{11})^{-1}, \\
(\mathbf{C}_{22}^{-1})^{(t+1)} &= \frac{1}{\rho+\eta-n-1}\Big(\frac{\rho}{\rho-n_1}\mathbf{K}_{22\cdot 1}^{(t)}+\mathbf{\Theta}_{22}+\mathbf{K}_{21}^{(t)}\mathbf{K}_{11}^{-1}\mathbf{K}_{12}^{(t)} \\
&\qquad -(\mathbf{K}_{21}^{(t)}+\mathbf{\Theta}_{21})(\mathbf{K}_{11}+\mathbf{\Theta}_{11})^{-1}(\mathbf{K}_{12}^{(t)}+\mathbf{\Theta}_{12})\Big), \\
(\mathbf{C}_{22}^{-1})^{(t+1)}\mathbf{C}_{21}^{(t+1)} &= -(\mathbf{K}_{21}^{(t)}+\mathbf{\Theta}_{21})(\mathbf{K}_{11}+\mathbf{\Theta}_{11})^{-1}.
\end{aligned} \tag{6}$$

The EM algorithm iterates by alternating between the E-step and M-step until convergence.

4 Learning with a Mixture Model of Kernels

4.1 Mixture Model of Kernels

In the kernel learning literature, the target kernel is often constrained to be a weighted combination of some fixed base kernels. A common choice for these base kernels is $\mathbf{M}_i = \boldsymbol{\mu}_i\boldsymbol{\mu}_i'$ $(i = 1, \ldots, n)$, where $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_n$ are a set of independent vectors. The target kernel matrix $\mathbf{A}$ is thus constrained to be of the form $\sum_{i=1}^{n}\lambda_i\mathbf{M}_i$ and the learning problem is subsequently simplified to the estimation of λ_i's. Usually, these $\boldsymbol{\mu}_i$'s are further assumed to be orthogonal to each other, implying that $\boldsymbol{\mu}_i$'s and λ_i's are respectively the eigenvectors and eigenvalues of $\mathbf{A}$.

More generally, these $\mathbf{M}_i$'s can be different kernel matrices resulting from the use of, say, different kernels functions [14]. As in Section 3.1, we again use $\mathbf{K} \sim \mathcal{W}_n(\rho, \mathbf{\Sigma})$ in (4) but with

$$\mathbf{\Sigma} = \sum_{k=1}^{K}\alpha_k\mathbf{M}_k, \tag{7}$$

where α_k's are nonnegative constants, and $\mathbf{M}_k$'s are K mutually independent random Wishart matrices such that $\mathbf{M}_k \succ 0$ and

$$\mathbf{M}_k \sim \mathcal{W}_n(\eta_k, \mathbf{\Theta}_k). \tag{8}$$

This thus defines a probabilistic mixture model of kernels. Notice that this mixture model differs from the usual ones, such as the Gaussian mixture model [19], in that ours is based on random matrices while the

latters are based on random vectors.

By setting the matrix $\mathbf{A}$ in Theorem 1 to $\sqrt{\alpha_k}\ \mathbf{I}$, it is easy to see that $\alpha_k \mathbf{M}_k \sim \mathcal{W}_n(\eta_k, \alpha_k \mathbf{\Theta}_k)$. If $\alpha_1 \mathbf{\Theta}_1 = \alpha_2 \mathbf{\Theta}_2 = \cdots = \alpha_K \mathbf{\Theta}_K$, the distribution of $\mathbf{\Sigma}$ in (7) will still be Wishart (Theorem 2). Unfortunately, this assumption is often too restrictive. Moreover, since part of the observation in our kernel mixture model contains missing data, it is very difficult to infer the model in the same way as for Gaussian mixture models. Thus, instead, our approach is to first approximate the distribution of $\mathbf{\Sigma}$ by the distribution of a random Wishart matrix (Section 4.2)

$$\mathbf{W} \sim \mathcal{W}_n(\eta, \mathbf{\Theta}), \tag{9}$$

and the EM algorithm in Section 3.2 is then performed with this $\mathbf{W}$ (Section 4.3). Figure 1 depicts a hierarchical model for the relationships between the kernel sub-matrices $(\mathbf{K}_{11}, \mathbf{K}_{21}, \mathbf{K}_{22\cdot1})$, the parameters ($\alpha_k$'s, $\mathbf{M}_k$'s, ρ), and the hyperparameters ($\mathbf{\Theta}_k$'s, η_k's, η).

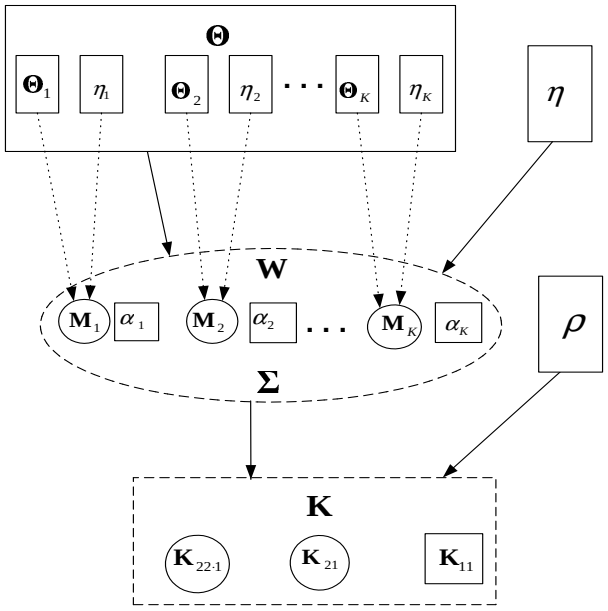


Figure 1: The mixture model of kernels. Here, □ are known while ◯ are not.

4.2 Approximating a Linear Combination of Wishart Matrices

As introduced in Section 4.1, we will use the distribution of a single Wishart matrix $\mathbf{W}$ in (9) to approximate that of $\mathbf{\Sigma}$ in (7). The problem is then on how to obtain the parameters η and $\mathbf{\Theta}$. First, with Proposition 3, the first and second moments of $\mathbf{\Sigma}$ and $\mathbf{W}$ are:

$$\mathrm{E}(\mathbf{\Sigma}) = \sum_{k=1}^{K} \alpha_k \eta_k \mathbf{\Theta}_k, \qquad \mathrm{E}(\mathbf{W}) = \eta \mathbf{\Theta},$$

$$\mathrm{Cov}(\mathrm{vec}(\mathbf{\Sigma})) = \sum_{k=1}^{K} \alpha_k^2 \eta_k (\mathbf{I} + \mathbf{H}_{n^2})(\mathbf{\Theta}_k \otimes \mathbf{\Theta}_k), \qquad \mathrm{Cov}(\mathrm{vec}(\mathbf{W})) = \eta(\mathbf{I} + \mathbf{H}_{n^2})(\mathbf{\Theta} \otimes \mathbf{\Theta}).$$

By equating the moments, i.e., $\mathrm{E}(\mathbf{\Sigma}) = \mathrm{E}(\mathbf{W})$ and $\mathrm{Cov}(\mathrm{vec}(\mathbf{\Sigma})) = \mathrm{Cov}(\mathrm{vec}(\mathbf{W}))$, we obtain

$$\mathbf{\Theta} = \frac{1}{\eta} \sum_{k=1}^{K} \alpha_k \eta_k \mathbf{\Theta}_k, \tag{10}$$

$$\eta(\mathbf{I} + \mathbf{H}_{n^2})(\mathbf{\Theta} \otimes \mathbf{\Theta}) = \sum_{k=1}^{K} \alpha_k^2 \eta_k (\mathbf{I} + \mathbf{H}_{n^2})(\mathbf{\Theta}_k \otimes \mathbf{\Theta}_k). \tag{11}$$

Note, however, that the degree of freedom of the unknown parameters in $\mathbf{\Theta}$ and η is $\frac{1}{2}n(n+1)+1$, but (10) and (11) correspond to $n(n+1)$ different equations. Instead of using equation (11) directly, we equate the respective traces of the two sides. From Propositions 1 and 2, we get

$$\eta\,(\mathrm{tr}(\mathbf{\Theta}^2) + (\mathrm{tr}(\mathbf{\Theta}))^2) = \sum_{k=1}^{K} \alpha_k^2\, \eta_k (\mathrm{tr}(\mathbf{\Theta}_k^2) + (\mathrm{tr}(\mathbf{\Theta}_k))^2). \tag{12}$$

Substituting (10) into (12), we can obtain η as

$$\eta = \frac{(\sum_{k=1}^{K} \eta_k \alpha_k \mathrm{tr}(\mathbf{\Theta}_k))^2 + (\mathrm{tr}(\sum_{k=1}^{K} \eta_k \alpha_k \mathbf{\Theta}_k))^2}{\sum_{k=1}^{K} \eta_k \alpha_k^2 (\mathrm{tr}(\mathbf{\Theta}_k))^2 + \sum_{k=1}^{K} \eta_k \alpha_k^2 \mathrm{tr}(\mathbf{\Theta}_k^2)}. \tag{13}$$

Substituting η back into (10), we can then obtain the estimate of $\mathbf{\Theta}$.

Previously, Tan and Gupta [24] and Khatri [15] have proposed methods for this approximation problem, which are also based on comparing the expectations of $\mathbf{\Sigma}$ and $\mathbf{W}$, and the generalized variances of vecp($\mathbf{\Sigma}$) and vecp($\mathbf{W}$). The method in [15] equates the traces of the generalized variances of vecp($\mathbf{\Sigma}$) and vecp($\mathbf{W}$), and is thus similar to ours. The method in [24], on the other hand, equates the determinants of the generalized variances of vecp($\mathbf{\Sigma}$) and vecp($\mathbf{W}$). If it were applied here, we would get

$$|\eta(\mathbf{I} + \mathbf{H}_{n^2})(\mathbf{\Theta} \otimes \mathbf{\Theta})| = \left| \sum_{k=1}^{K} \alpha_k^2 \eta_k (\mathbf{I} + \mathbf{H}_{n^2})(\mathbf{\Theta}_k \otimes \mathbf{\Theta}_k) \right|$$

instead of (12), and subsequently

$$\begin{aligned}\eta &= \left[\frac{\left|(\sum_{k=1}^{K}\alpha_k\eta_k\boldsymbol{\Theta}_k)\otimes(\sum_{k=1}^{K}\alpha_k\eta_k\boldsymbol{\Theta}_k)\right|}{\left|\sum_{k=1}^{K}\alpha_k^2\eta_k(\boldsymbol{\Theta}_k\otimes\boldsymbol{\Theta}_k)\right|}\right]^{\frac{1}{n^2}} \\ &= \left[\frac{\left|\sum_{k=1}^{K}\alpha_k\eta_k\boldsymbol{\Theta}_k\right|^{2n}}{\left|\sum_{k=1}^{K}\alpha_k^2\eta_k(\boldsymbol{\Theta}_k\otimes\boldsymbol{\Theta}_k)\right|}\right]^{\frac{1}{n^2}}. \end{aligned} \tag{14}$$

However, this requires computing the determinant of an $n^2 \times n^2$ matrix, and is thus intractable as $n = n_1 + n_2$ becomes large. Unfortunately, this is also usually the case in kernel matrix learning. Therefore, we recommend using equation (13) for our kernel matrix learning problem.

Finally, recall from Definition 4 that the condition $\eta \geq n$ should hold for the Wishart distribution $\mathcal{W}_n(\eta, \boldsymbol{\Theta})$ to be meaningful. However, this condition has not been proved in neither [15] nor [24]. The following theorem guarantees that the condition always holds for our methods:

Theorem 4 *Assume that $\eta_k \geq n$ and $\alpha_k \geq 0$ for $k = 1, \ldots, K$, and also that at least one of the α_k's is nonzero. If η is defined by (13) or (14), then $\eta \geq n$.*

The proof is in the appendix.

As in traditional mixture models [19], we shall assume that $\alpha_k \in [0, 1]$ and $\sum_{k=1}^{K}\alpha_k = 1$ in the sequel. Moreover, in the extreme case when $\alpha_j = 1$ (for $j \in \{1, \ldots, K\}$) and $\alpha_k = 0$ for $k \neq j$, it is easy to see that (10) and (13) (or (14)) degenerate to $\boldsymbol{\Theta} = \boldsymbol{\Theta}_j$ and $\eta = \eta_j$, respectively. Thus, our mixture model reduces to the case in [29]. In other words, our mixture model is an extension of the probabilistic kernel matrix learning model in [29].

4.3 Kernel Matrix Learning with the Mixture Model

After obtaining the parameters (η and $\boldsymbol{\Theta}$) for approximating the linear combination of Wishart matrices in $\boldsymbol{\Sigma}$, we can then directly apply the EM algorithm in Section 3.2. Figure 2 summarizes the complete learning algorithm:

Input	Give $\mathbf{K}_{11}$, ρ, α_k's, η_k's, and $\mathbf{\Theta}_k$'s.
Mixture approximation	Estimate η and $\mathbf{\Theta}$ using (13), (10) and obtain $\mathbf{W} \sim \mathcal{W}_n(\eta, \mathbf{\Theta})$.
EM learning	Iterate by alternating between the E-step in (5) and the M-step in (6) with $\mathbf{\Sigma}$ substituted by $\mathbf{W}$.
Output	Obtain $\mathbf{K}_{21}, \mathbf{K}_{22\cdot 1}$ and then calculate $\mathbf{K}_{22} = \mathbf{K}_{22\cdot 1} + \mathbf{K}_{21}\mathbf{K}_{11}^{-1}\mathbf{K}_{12}$.

Figure 2: Proposed algorithm for learning the kernel mixture.

5 Kernel Matrix Learning in Classification Problems

In general, the definitions of both the kernel matrix $\mathbf{K}$ and the hyperparameter matrices $\mathbf{\Theta}_k$'s will depend on both the concerned problem and the prior knowledge available. While the kernel matrix learning framework presented above is not limited to the classification problem, our focus in this paper is to use the classification problem to illustrate how kernel matrix learning can be performed. Typically, kernel-based classification methods such as SVM [6, 27] and kernel FDA [3] first select or learn an appropriate kernel from data empirically, and then train a classifier with the kernel. The two problems, namely, kernel selection and classifier training, are treated as two entirely separate processes. Here, we will integrate these two problems tightly so that kernel learning forms an integral part of the entire classifier training process.

5.1 Defining the Target Kernel Matrix K

As before, we denote the input parts of the training and test sets by $\mathcal{I} = \{\mathbf{x}_i\}_{i=1}^{n_1}$ and $\tilde{\mathcal{I}} = \{\tilde{\mathbf{x}}_i\}_{i=1}^{n_2}$, respectively. Similarly, for the output parts, we have $\mathcal{O} = \{y_i\}_{i=1}^{n_1}$ and $\tilde{\mathcal{O}} = \{\tilde{y}_i\}_{i=1}^{n_2}$. In a classification problem, the outputs $\tilde{y}_i$'s are unknown and have to be estimated. We also denote the entire training and test sets by $\mathcal{T} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{n_1}$ and $\tilde{\mathcal{T}} = \{(\tilde{\mathbf{x}}_i, \tilde{y}_i)\}_{i=1}^{n_2}$, respectively. Furthermore, let $\mathbf{K}_I$ and $\mathbf{K}_O$ denote the input and output kernel matrices defined on the combined input set $\mathcal{I} \cup \tilde{\mathcal{I}}$ and the combined output set $\mathcal{O} \cup \tilde{\mathcal{O}}$, respectively. Since a kernel matrix can also be regarded as an affinity matrix for point pairs, $\mathbf{K}_I$ measures the similarity between input vectors while $\mathbf{K}_O$ measures the similarity between output vectors.

The similarity between kernel matrices $\mathbf{K}_I$ and $\mathbf{K}_O$ can in turn be measured by the *alignment*, as [8]:

$$A = \frac{\langle \mathbf{K}_I, \mathbf{K}_O \rangle_F}{\sqrt{\langle \mathbf{K}_I, \mathbf{K}_I \rangle_F \langle \mathbf{K}_O, \mathbf{K}_O \rangle_F}},$$

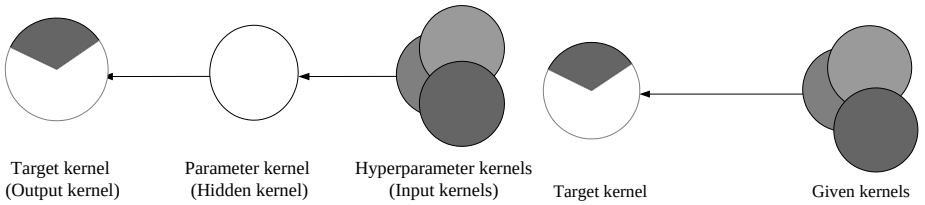
where $\langle \cdot, \cdot \rangle_F$ is the Frobenius norm for matrices.

This motivates us to directly employ the output kernel matrix $\mathbf{K}_O$ to define the $\mathbf{K}$ in Section 4. To be more specific, we define $\mathbf{K} = \begin{bmatrix} \mathbf{K}_{11} & \mathbf{K}_{12} \\ \mathbf{K}_{21} & \mathbf{K}_{22} \end{bmatrix}$, where $\mathbf{K}_{11}$ is defined on $\mathcal{O} \times \mathcal{O}$, $\mathbf{K}_{22}$ is defined on $\tilde{\mathcal{O}} \times \tilde{\mathcal{O}}$, and $\mathbf{K}_{12} = \mathbf{K}'_{21}$ is defined on $\mathcal{O} \times \tilde{\mathcal{O}}$. Though only $\mathbf{K}_{11}$ has been specified in the training data, both $\mathbf{K}_{21}$ and $\mathbf{K}_{22}$ can be obtained by our kernel matrix learning algorithm (Figure 2). While $\mathbf{K}_{11}$ captures the class label information from the training data and $\mathbf{K}_{22}$ captures the class label information from the test data, $\mathbf{K}_{21}$ characterizes the similarity between the class labels of the training and test data. Thus, we expect that it is effective to perform classification by employing the information in $\mathbf{K}_{21}$ or $\mathbf{K}_{22}$.

5.2 Defining the Hyperparameter Matrices $\mathbf{\Theta}_k$'s

Another major decision is on how to select the hyperparameter matrices. Here we set them to different choices of the input kernel matrix $\mathbf{K}_I$. Figure 3(b) illustrates our hierarchical model, which relates the input kernel matrices (hyperparameter kernel matrices) to the output kernel matrix (target kernel matrix) indirectly through a hidden kernel matrix, called the parameter kernel matrix. Our model is clearly different from kernel alignment, as illustrated in Figure 3(a), since our model involves a hidden kernel in between.

Our hierarchical model has at least two advantages. First, as we can see from the kernel matrix learning algorithm summarized in Figure 2, the hyperparameter kernel matrices play the role of a regularization term so that overfitting can be avoided. Second, as argued by Crammer *et al.* [7], the alignment may not be a good enough measure for classification tasks since a classifier may achieve zero error rate even though the alignment is still far below one. This point will further be illustrated by our experiments to be discussed in the next section. To a certain extent, our method can overcome this problem.



(a) Our hierarchical model. (b) Kernel alignment.

Figure 3: Comparison of the proposed hierarchical model with kernel alignment. Here, the shaded parts are known while the unshared parts are not.

5.3 Predicting the Labels of Test Patterns

Based on the discussions above, here we propose a classification method that tightly integrates kernel selection with classifier training. The basic process of the classification method is illustrated in Figure 4 below. It is easy to see that the method is in fact a kernel version of the nearest neighbor classifier, or called *kernel nearest neighbor classifier* (KNNC). Like other kernel methods, KNNC works over the feature space without explicit use of the feature vectors $\varphi(\mathbf{x})$'s themselves. However, unlike other kernel-based classification methods such as SVM [6, 27] and kernel FDA [3], KNNC does not require solving a quadratic programming problem or an eigen decomposition problem. The consequence of this is that our method is simpler and significantly more efficient.

Input	Give $\mathcal{T} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{n_1}$ and $\tilde{\mathcal{I}} = \{\tilde{\mathbf{x}}_i\}_{i=1}^{n_2}$.
Kernel learning	Use the kernel matrix learning algorithm given in Figure 2 to estimate the missing entries of $\mathbf{K}$.
Classification	For every $1 \leq i \leq n_2$, assign $\tilde{y}_i = y_{j^*}$ where $j^* = \arg\max_j [\mathbf{K}_{21}]_{ij}$.

Figure 4: Kernel nearest neighbor classifier

Another possibility is to use a kernel version of the nearest mean classifier, or called *kernel nearest mean classifier* (KNMC), for classification. Apparently, it can be seen as a nearest mean classifier in the kernel-induced feature space. Let us denote the size of class i, denoted $\mathcal{C}_i$, by N_i and the (feature-space) class mean of $\mathcal{C}_i$ by $\mathbf{m}_i = \frac{1}{N_i} \sum_{\mathbf{x}_j \in \mathcal{C}_i} \varphi(\mathbf{x}_j)$. KNMC allocates a data point $\mathbf{x}$ in the test set to $\mathcal{C}_i$ if $\|\varphi(\mathbf{x}) - \mathbf{m}_i\|^2 \leq$

$\|\varphi(\mathbf{x}) - \mathbf{m}_j\|^2$ for all $j \neq i$, where

$$\begin{aligned} \|\varphi(\mathbf{x}) - \mathbf{m}_i\|^2 &= \varphi(\mathbf{x})'\varphi(\mathbf{x}) + \mathbf{m}_i'\mathbf{m}_i - 2\varphi(\mathbf{x})'\mathbf{m}_i \\ &= k(\mathbf{x},\mathbf{x}) + \frac{1}{N_i^2}\sum_{\mathbf{x}_j,\mathbf{x}_l \in \mathcal{C}_i} k(\mathbf{x}_j,\mathbf{x}_l) - \frac{2}{N_i}\sum_{\mathbf{x}_j \in \mathcal{C}_i} k(\mathbf{x},\mathbf{x}_j), \end{aligned} \tag{15}$$

by using the kernel trick. For our problem, we am interested in the implementation of the KNMC on the feature space corresponding to the output space. This is simply done through replacing $\mathbf{x}$ and $\mathbf{x}_i$'s in (15) with y and $y_i's$, respectively.

6 Experiments

In the following experiments, we define the $\mathbf{K}_{11}$ by employing the *ideal kernel* $\mathbf{K}^*(\mathcal{O}) = (k^*_{ij})$ on the output part of the training set $\mathcal{O}$ [8]. Namely,

$$k^*_{ij} = \begin{cases} 1 & y_i = y_j \\ 0 & y_i \neq y_j. \end{cases}$$

Furthermore, we set

$$\mathbf{K}_{11} = \mathbf{K}^*(\mathcal{O}) + \epsilon\mathbf{I},$$

where ϵ is a small positive number and the $\epsilon\mathbf{I}$ term is for avoiding any possible singularity of $\mathbf{K}^*(\mathcal{O})$. Besides, ρ in (4) is always set to n+1, and the EM algorithm is run for 100 iterations. Notice that as both $\mathbf{K}_{11}$ and $\mathbf{\Theta}_{11}$ are known and fixed, it is only required to compute the inverse of $\mathbf{K}_{11} + \mathbf{\Theta}_{11}$ in (6) once. Also, as the remaining steps of the EM algorithm require only basic matrix operations, its computational requirement is considerably low.

Besides the test set classification accuracy, we will also report the alignment between the estimated kernel sub-matrix $\mathbf{K}_{22}$ and the ideal kernel matrix $\mathbf{K}^*(\tilde{\mathcal{O}})$ on the test set $\tilde{\mathcal{O}}$:

$$\frac{\langle \mathbf{K}_{22}, \mathbf{K}^*(\tilde{\mathcal{O}})\rangle_F}{\sqrt{\langle \mathbf{K}_{22}, \mathbf{K}_{22}\rangle_F \langle \mathbf{K}^*(\tilde{\mathcal{O}}), \mathbf{K}^*(\tilde{\mathcal{O}})\rangle_F}}.$$

Moreover, results reported in the sequel are averages over 30 random splits of the data, each with 60% of the data for training and 40% for testing. The standard deviations are also reported (inside brackets) in the

tables.

6.1 Experiment 1: Mixtures of Different Types of Kernels

In the first experiment, we use a mixtures of three components, with each of the hyperparameter matrices $\mathbf{\Theta}_1$, $\mathbf{\Theta}_2$ and $\mathbf{\Theta}_3$ corresponding to the Gaussian kernel

$$\mathbf{\Theta}_1 = \left[\exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right)\right]_{n\times n} \tag{16}$$

with $\sigma^2 = 0.75$, polynomial kernel

$$\mathbf{\Theta}_2 = \left[(1 + \mathbf{x}_i'\mathbf{x}_j)^d\right]_{n\times n}$$

with $d = 2$, and the linear kernel

$$\mathbf{\Theta}_3 = [\mathbf{x}_i'\mathbf{x}_j]_{n\times n}\,,$$

respectively. All the α_k's ($k = 1, 2, 3$) in (7) are set to $\frac{1}{3}$. As for ρ, η_k in (8) is also set to n+1. The experiments are performed on six benchmark data sets (Wisconsin breast cancer, ionosphere, soybean, wine, sonar and iris) from the UCI Machine Learning Repository[1].

First, the experiments are performed in a non-mixture setting by using each of these hyperparameter matrices alone. Namely, we compare the use of each hyperparameter kernel matrix $\mathbf{\Theta}_k$ with the corresponding target kernel $\mathbf{K}$ obtained from kernel matrix learning. Kernel matrix learning is based on the EM learning algorithm according to equations (5) and (6). The comparison is based on both kernel alignment and classification accuracy. Table 1 shows the results for the six data sets.[2] The rows marked with Gaussian, polynomial, and linear show the corresponding results for the three hyperparameter kernel choices. As we can see, the classification accuracy obtained with the target kernel learned always outperforms that when the corresponding hyperparameter kernel is used directly without kernel learning. However, this is not always the case when using the kernel alignment as criterion for comparison. For example, for the Wisconsin breast cancer and ionosphere data sets, the alignment values of the target kernel with the polynomial hyperparameter kernel are significantly lower than the corresponding values using the polynomial kernel directly without

[1]The UCI data sets can be downloaded from `http://www.ics.uci.edu/~mlearn/MLRepository.html`.

[2]The highest alignment values and classification accuracies among all kernel learning methods are shown in boldface.

kernel learning. For the sonar and soybean data sets, the target kernel even gives lower alignment values for all three choices of the hyperparameter kernel. This further supports the argument of Crammer *et al.* [7] that the alignment may not be a satisfactory measure for classification tasks. We conjecture that higher alignment is only sufficient but not necessary for obtaining higher classification accuracy, at least for distance-based classification methods. In other words, higher alignment is expected to give higher classification accuracy, but higher classification accuracy does not necessarily require higher alignment. In terms of the classification accuracy, the Gaussian kernel is the best hyperparameter kernel choice for the Wisconsin breast cancer, ionosphere, sonar and soybean data sets, while the polynomial kernel is the best for the iris data set and the linear kernel is the best for the wine data set. The results show that there does not exist a single kernel choice that is the best for all data sets.

Next, we apply the above kernel mixture model to the six data sets based on the KNNM. Table 2 shows the estimated values of η for the six data sets. Clearly, the inequality $\eta \geq n$ holds for each of the six data sets. The kernel alignment values and classification accuracies with the kernel mixture model are shown in the rows of Table 1 that marked with mixture. The mixture model without kernel learning is simply based on the kernel matrix $\sum_{k=1}^{3} \alpha_k \mathbf{\Theta}_k = \frac{1}{3} \sum_{k=1}^{3} \mathbf{\Theta}_k$. As before, in terms of classification accuracy, the target kernel obtained from kernel learning always outperforms the corresponding hyperparameter kernel without kernel learning. However, when comparing the different cases with kernel learning, the performance of the mixture model is intermediate between the non-mixture cases. This suggests that the kernel mixture model tends to achieve a performance tradeoff among the individual non-mixture models.

6.2 Experiment 2: Mixtures of Same Type of Kernels

In the second experiment, we use a kernel mixtures with five components. All the corresponding hyperparameter matrices $(\mathbf{\Theta}_1, \ldots, \mathbf{\Theta}_5)$ are from the same kernel function (namely, the Gaussian kernel in (16)) but with different kernel parameter (σ_k) settings, as

$$\sigma_k^2 = 0.5 + 0.25(k-1).$$

Table 1: Kernel alignment values and test set accuracies obtained in the first experiment. The highest alignment values and accuracies are shown in boldface.

data set	kernel	target kernel		hyperparameter kernel	
		alignment	accuracy (%)	alignment	accuracy (%)
breast cancer	Gaussian	0.12 (±0.003)	**95.96** (±1.0)	0.12 (±0.004)	74.50 (±3.6)
	polynomial	0.02 (±0.008)	83.61 (±2.4)	0.18 (±0.042)	78.86 (±2.5)
	linear	**0.43** (±0.054)	95.64 (±1.3)	0.36 (±0.022)	92.78 (±1.3)
	mixture	0.03 (±0.016)	90.45 (±1.8)	0.19 (±0.043)	80.66 (±2.6)
ionosphere	Gaussian	0.27 (±0.014)	**92.14** (±1.8)	0.44 (±0.029)	73.24 (±3.4)
	polynomial	0.06 (±0.006)	78.12 (±3.9)	0.53 (±0.023)	70.62 (±4.5)
	linear	**0.71** (±0.034)	84.24 (±1.9)	0.59 (±0.024)	74.83 (±5.4)
	mixture	0.11 (±0.008)	85.48 (±3.1)	0.54 (±0.023)	70.95 (±4.5)
soybean	Gaussian	**0.68** (±0.034)	**99.47** (±1.6)	0.75 (±0.018)	96.84 (±5.1)
	polynomial	0.39 (±0.074)	99.12 (±2.0)	0.65 (±0.009)	94.54 (±7.1)
	linear	0.55 (±0.100)	97.37 (±2.7)	0.60 (±0.006)	95.44 (±7.1)
	mixture	0.40 (±0.074)	99.12 (±2.0)	0.65 (±0.009)	94.56 (±7.1)
wine	Gaussian	0.25 (±0.006)	96.48 (±1.8)	0.26 (±0.006)	77.70 (±4.3)
	polynomial	0.11 (±0.028)	92.54 (±2.6)	0.56 (±0.029)	79.06 (±6.3)
	linear	**0.61** (±0.018)	**97.89** (±1.6)	0.55 (±0.023)	96.90 (±1.5)
	mixture	0.28 (±0.057)	93.00 (±2.4)	0.59 (±0.029)	92.30 (±3.0)
sonar	Gaussian	0.37 (±0.015)	**84.26** (±3.2)	0.56 (±0.013)	71.04 (±6.5)
	polynomial	0.21 (±0.021)	81.08 (±3.0)	0.68 (±0.004)	65.66 (±5.2)
	linear	0.65 (±0.028)	73.33 (±4.8)	0.70 (±0.002)	66.06 (±6.4)
	mixture	**0.71** (±0.033)	82.69 (±3.5)	0.68 (±0.004)	65.74 (±5.3)
iris	Gaussian	0.54 (±0.080)	94.61 (±2.1)	0.75 (±0.027)	90.89 (±4.3)
	polynomial	0.80 (±0.022)	**96.06** (±2.2)	0.64 (±0.018)	87.22 (±3.5)
	linear	0.55 (±0.010)	86.44 (±3.6)	0.54 (±0.010)	81.83 (±3.2)
	mixture	**0.92** (±0.018)	94.78 (±2.7)	0.68 (±0.019)	87.11 (±3.7)

Table 2: Estimated values of η in the kernel mixture model in the first experiment.

	breast cancer	ionosphere	soybean	wine	sonar	iris
n	569	351	47	178	208	150
η	582.6	392.8	49.4	200.1	247.8	192.9

As in Section 6.1, all α_k's in (7) are the same and equal to $\frac{1}{5}$, while η_k in (8) is set to $n+1$. Here, the experiments are performed on the three largest data sets (namely, breast cancer, ionosphere and sonar) only.

As can be seen in Table 3, the test set accuracies obtained by the learned kernel mixture are close to those from the best setting of σ_k. Moreover, the alignment values obtained by the learned kernel mixtures are 0.57, 0.81, and 0.80, respectively, for the Wisconsin breast cancer, ionosphere, and sonar data sets. They all significantly higher than those of the non-mixture cases. Besides, the estimated values of η in (9) on the three data sets are $2849.6, 1756.3, 1034.8$, respectively. Recall that η_k has been set to $n+1$ in this experiment and its values on the three data sets are 570, 352, 209, respectively. Thus, η happens to be related to η_k as $\eta \simeq 5 \times \eta_k = \sum_{k=1}^{5} \eta_k$ as all η_k's are equal. This is a very interesting phenomenon. It may be explained by revisiting the learning procedure. We first approximates the mixture consisting of $\mathbf{\Theta}_k$'s with a single kernel matrix $\mathbf{\Theta}$. If we replace each $\mathbf{M}_k \sim \mathcal{W}_n(\eta_k, \mathbf{\Theta}_k)$ by $\mathbf{M}_k \sim \mathcal{W}_n(\eta_k, \mathbf{\Theta})$. Then, from Theorems 1 and 2, it follows that $\sum_{k=1}^{5} \alpha_k \mathbf{M}_k = \frac{1}{5} \sum_{k=1}^{5} \mathbf{M}_k \sim \mathcal{W}_n(\sum_{k=1}^{5} \eta_k, \frac{1}{5}\mathbf{\Theta})$. We can thus interpret the phenomenon as $\sum_{k=1}^{5} \alpha_k \mathbf{M}_k$ itself approaching a Wishart distribution. Furthermore, the hyperparameter matrix of this Wishart distribution approaches a Gaussian kernel matrix with an appropriate width. Based on these, we think that our kernel mixture model provides an effective way of choosing an appropriate parameter value for the Gaussian kernel. In particular, we can first estimate an interval to which the appropriate value of the width belongs, select several values from this interval, and then apply our model with these values.

On the contrary, the above-mentioned phenomenon does not occur in our first experiment, as the estimated values of η on the same three data sets are 582.6, 392.8 and 247.8 (see Table 2). This implies that a new hyperparameter matrix, differing from either of the Gaussian, polynomial and linear kernels, has been constructed. This opens up a possible direction for addressing the kernel selection problem.

Table 3: Kernel alignment values and test set accuracies obtained in the second experiment.

data set	parameter	target kernel		hyperparameter kernel	
	σ_k^2	alignment	accuracy (%)	alignment	accuracy (%)
breast cancer	0.50	0.10 (±0.001)	95.80 (±1.0)	0.10 (±0.001)	63.42 (±3.6)
	0.75	0.12 (±0.003)	96.20 (±1.0)	0.12 (±0.003)	74.37 (±4.0)
	1.00	0.14 (±0.005)	96.39 (±0.9)	0.15 (±0.007)	79.88 (±3.9)
	1.25	0.17 (±0.007)	96.56 (±0.9)	0.19 (±0.010)	78.14 (±3.0)
	1.50	0.19 (±0.008)	**96.61** (±0.8)	0.23 (±0.013)	78.68 (±3.0)
	mixture	**0.57** (±0.021)	96.58 (±0.9)	0.15 (±0.007)	82.11 (±3.8)
ionosphere	0.50	0.24 (±0.012)	**92.79** (±1.8)	0.36 (±0.025)	68.10 (±5.0)
	0.75	0.27 (±0.012)	91.95 (±1.7)	0.42 (±0.026)	71.86 (±4.5)
	1.00	0.28 (±0.012)	91.95 (±2.0)	0.46 (±0.026)	73.12 (±3.9)
	1.25	0.30 (±0.011)	91.12 (±2.0)	0.50 (±0.026)	73.43 (±3.7)
	1.50	0.31 (±0.011)	90.38 (±2.2)	0.52 (±0.025)	74.10 (±3.4)
	mixture	**0.81** (±0.013)	92.38 (±2.0)	0.46 (±0.026)	73.45 (±4.0)
sonar	0.50	0.31 (±0.011)	83.82 (±3.8)	0.46 (±0.013)	75.30 (±6.9)
	0.75	0.36 (±0.015)	84.26 (±4.0)	0.56 (±0.013)	73.21 (±6.9)
	1.00	0.40 (±0.018)	**84.46** (±3.9)	0.61 (±0.012)	71.53 (±6.3)
	1.25	0.44 (±0.020)	84.30 (±3.7)	0.64 (±0.011)	70.96 (±6.1)
	1.50	0.48 (±0.022)	83.90 (±3.9)	0.66 (±0.009)	70.68 (±5.9)
	mixture	**0.80** (±0.017)	83.98 (±3.4)	0.61 (±0.011)	71.65 (±6.5)

7 Concluding Remarks

In this paper, we extended our previous work on probabilistic kernel matrix learning [29] by representing the parameter matrix of the Wishart distribution as a linear combination of mutually independent Wishart matrices. This probabilistic mixture model of kernels can be seen as a hierarchical model with three levels, relating the target kernel matrix, the parameter kernel matrix, and the hyperparameter matrices of the mixture model. In particular, in the context of classification problems, we define the hyperparameter kernel matrices and target kernel matrix on the input space and output space, respectively, and use a kernel nearest neighbor (or mean) classifier for prediction. The efficacy of our methods is also demonstrated by experiments on several benchmark data sets.

Recall that the mixing coefficients (α_k's) in our model have to be specified in advance. In our experiments, we used $\alpha_k = \frac{1}{K}$ where K is the number of mixture components. In general, they can take any nonnegative values. Better still, the mixing coefficients should preferably be adjusted adaptively. However, unlike traditional finite mixture models such as the Gaussian mixture model [19] in which each component corresponds to the distribution of a random vector and the missing data are the labels of the observations, each component in our mixture model corresponds to the distribution of a random matrix and the missing data is also part of the observation itself. Casting the adaptation of α_k's into our current EM algorithm is non-trivial. Obviously, solving this problem is essential to the design of a general method for kernel model selection and kernel parameter estimation. This will be an interesting topic for further research.

Finally, our classification method represents an attempt to integrate kernel selection with classifier construction. This way of treating kernel selection and classifier construction in a single process, rather than as two separate processes, is also a potentially fruitful research direction in kernel-based procedures.

Appendix — Proof of Theorem 4

In this Appendix, we will prove the condition $\eta \geq n$ with η defined by (13) or (14). First, we quote the following result from [12].

Lemma 1 *Let* $\mathbf{A} \in \mathbb{R}^{p\times q}$ *and* $\mathbf{B} \in \mathbb{R}^{q\times m}$ *be given,* $l = \min\{p, q, m\}$*, and denote the ordered singular values of* $\mathbf{A}$*,* $\mathbf{B}$*, and* $\mathbf{AB}$ *by* $\delta_1(\mathbf{A}) \geq \cdots \geq \delta_{\min\{p,q\}}(\mathbf{A}) \geq 0$*,* $\delta_1(\mathbf{B}) \geq \cdots \geq \delta_{\min\{p,q\}}(\mathbf{B}) \geq 0$*, and* $\delta_1(\mathbf{AB}) \geq \cdots \geq \delta_{\min\{p,q\}}(\mathbf{AB}) \geq 0$*. Then,*

$$\sum_{i=1}^{l} \delta_i(\mathbf{AB}) \leq \sum_{i=1}^{l} \delta_i(\mathbf{A})\delta_i(\mathbf{B}).$$

Furthermore, if $\mathbf{A}$ and $\mathbf{B}$ are $m \times m$ positive definite matrices and denote the ordered eigenvalues of $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{AB}$ by $\lambda_i(\mathbf{A})$, $\lambda_i(\mathbf{B})$, and $\lambda_i(\mathbf{AB})$ for $1 \leq i \leq m$, then $\lambda_i(\mathbf{A}) = \delta_i(\mathbf{A}) > 0$, $\lambda_i(\mathbf{B}) = \delta_i(\mathbf{B}) > 0$, and $\sum_i^m |\lambda_i(\mathbf{AB})| \leq \sum_i^m \delta_i(\mathbf{AB})$.

Proof of Theorem 4. We first prove the case when η is defined by (13). From Lemma 1, we have

$$\begin{aligned} |\mathrm{tr}(\mathbf{AB})| &= \left|\sum_{i=1}^{m} \lambda_i(\mathbf{AB})\right| \leq \sum_{i=1}^{m} |\lambda_i(\mathbf{AB})| \leq \sum_{i=1}^{m} \delta_i(\mathbf{AB}) \\ &\leq \sum_{i=1}^{m} \lambda_i(\mathbf{A})\lambda_i(\mathbf{B}) \leq \sum_{i=1}^{m} \lambda_i(\mathbf{A}) \sum_{i=1}^{m} \lambda_i(\mathbf{B}) = \mathrm{tr}(\mathbf{A})\mathrm{tr}(\mathbf{B}). \end{aligned}$$

Thus, $\mathrm{tr}(\mathbf{AB}) + \mathrm{tr}(\mathbf{A})\mathrm{tr}(\mathbf{B}) \geq 0$. Now, rewrite (13) as

$$\begin{aligned} \eta &= \frac{\sum_{k=1}^K \eta_k^2\alpha_k^2(\mathrm{tr}(\mathbf{\Theta}_k))^2 + \sum_{k=1}^K \eta_k^2\alpha_k^2\mathrm{tr}(\mathbf{\Theta}_k^2)}{\sum_{k=1}^K \eta_k\alpha_k^2(\mathrm{tr}(\mathbf{\Theta}_k))^2 + \sum_{k=1}^K \eta_k\alpha_k^2\mathrm{tr}(\mathbf{\Theta}_k^2)} \\ &\quad + \frac{\sum_{i\neq j}^K \eta_i\eta_j\alpha_i\alpha_j\mathrm{tr}(\mathbf{\Theta}_i)\mathrm{tr}(\mathbf{\Theta}_j) + \sum_{i\neq j}^K \eta_i\eta_j\alpha_i\alpha_j\mathrm{tr}(\mathbf{\Theta}_i\mathbf{\Theta}_j)}{\sum_{k=1}^K \eta_k\alpha_k^2(\mathrm{tr}(\mathbf{\Theta}_k))^2 + \sum_{k=1}^K \eta_k\alpha_k^2\mathrm{tr}(\mathbf{\Theta}_k^2)}. \end{aligned}$$

It is clear that the second term is nonnegative since $\mathrm{tr}(\mathbf{\Theta}_i\mathbf{\Theta}_j) + \mathrm{tr}(\mathbf{\Theta}_i)\mathrm{tr}(\mathbf{\Theta}_j) \geq 0$ for $i \neq j$, $\alpha_i \geq 0$, and $\eta_i \geq n$. As for the numerator of the first term,

$$\sum_{k=1}^{K} \eta_k^2\alpha_k^2(\mathrm{tr}(\mathbf{\Theta}_k))^2 + \sum_{k=1}^{K} \eta_k^2\alpha_k^2\mathrm{tr}(\mathbf{\Theta}_k^2) \geq n\left(\sum_{k=1}^{K} \eta_k\alpha_k^2(\mathrm{tr}(\mathbf{\Theta}_k))^2 + \sum_{k=1}^{K} \eta_k\alpha_k^2\mathrm{tr}(\mathbf{\Theta}_k^2)\right).$$

This shows that $\eta \geq n$.

When η is defined by (14), consider $\mathbf{\Theta}_k \succ 0$, $\alpha_k \geq 0$, and $\eta_k \geq n$ for $1 \leq k \leq K$. From Proposition 1(d), we know that $\mathbf{\Theta}_i \otimes \mathbf{\Theta}_j \succ 0$ for $i \neq j$. Thus,

$$\begin{aligned} \sum_{k=1}^{K} \alpha_k\eta_k\mathbf{\Theta}_k \otimes \sum_{k=1}^{K} \alpha_k\eta_k\mathbf{\Theta}_k &= \sum_{k=1}^{K} \alpha_k^2\eta_k^2(\mathbf{\Theta}_k \otimes \mathbf{\Theta}_k) + \sum_{j\neq i}\sum_{i=1}^{K} \alpha_i\alpha_j\eta_i\eta_j(\mathbf{\Theta}_i \otimes \mathbf{\Theta}_j) \\ &\succeq n\sum_{k=1}^{K} \alpha_k^2\eta_k(\mathbf{\Theta}_k \otimes \mathbf{\Theta}_k). \end{aligned}$$

From [12], we have the result that $|\mathbf{A}| \geq |\mathbf{B}|$ if $\mathbf{A}$ and $\mathbf{B}$ are symmetric positive semi-definite and $\mathbf{A} \succeq \mathbf{B}$. Using this result and Proposition 1(d) again, we have

$$\left|\sum_{k=1}^{K} \alpha_k \eta_k \mathbf{\Theta}_k\right|^{2n} \geq n^{n^2} \left|\sum_{k=1}^{K} \alpha_k^2 \eta_k (\mathbf{\Theta}_k \otimes \mathbf{\Theta}_k)\right|,$$

and hence $\eta \geq n$. □

References

[1] S. Amari. Information geometry of the EM and em algorithms for neural networks. *Neural Networks*, 8(9):1379–1408, 1995.

[2] N. Aronszajn. Theory of reeproducing kernels. *Transactions of the American Mathematical Society*, 68:337–404, 1950.

[3] G. Baudat and F. Anouar. Generalized discriminant analysis using a kernel approach. *Neural Computation*, 12:2385–2404, 2000.

[4] O. Bousquet and D. J. L Herrmann. On the complexity of learning the kernel matrix. In *Advances in Neural Information Processing Systems 15*, Cambridge, MA, 2003. MIT Press.

[5] O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee. Choosing multiple parameters for support vector machines. *Machine Learning*, 46(1-3):131–159, 2002.

[6] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20:273–297, 1995.

[7] K. Crammer, J. Keshet, and Y. Singer. Kernel design using boosting. In *Advances in Neural Information Processing Systems 15*, Cambridge, MA, 2003. MIT Press.

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

[9] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society Series B*, 39(1):1–38, 1977.

[10] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. John Wiley and Sons, New York, second edition, 2001.

[11] A.K. Gupta and D.K. Nagar. *Matrix Variate Distributions*. Chapman & Hall/CRC, 2000.

[12] R. A. Horn and C. R. Johnson. *Topics in Matrix Analysis*. Cambridge University Press, Cambridge, UK, 1991.

[13] J. Kandola and J. Shawe-Taylor. Refining kernels for regression and uneven classification problems. In C.M. Bishop and B.J. Frey, editors, *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics*, Key West, FL, 2003.

[14] J. Kandola, J. Shawe-Taylor, and N. Cristianini. Optimizing kernel alignment over combinations of kernels. NeuroCOLT Technical Report NC-TR-02-121, 2002.

[15] C. G. Khatri. Multivariate generalization of t'-statistics based on the mean square successive difference. *Communications in Statistics: Theory and Methods*, 18(5):1983–1992, 1989.

[16] J.T. Kwok. The evidence framework applied to support vector machines. *IEEE Transactions on Neural Networks*, 11(5):1162–1173, 2000.

[17] G. R. G. Lanckriet, N. Cristianini, L. El Ghaoui, P. Bartlett, and M. I. Jordan. Learning the kernel matrix with semi-definite programming. In *Proceedings of The 19th International Conference on Machine Learning*, pages 323–330, 2002.

[18] G. McLachlan and T. Krishnan. *The EM Algorithm and Extensions*. John Wiley and Sons, New York, 1997.

[19] G. McLachlan and D. Peel. *Finite Mixture Models*. John Wiley and Sons, New York, 2000.

[20] C. S. Ong and A. J. Smola. Machine learning with hyperkernels. In *The 20th International Conference on Machine Learning*, 2003.

[21] C. S. Ong, A. J. Smola, and R. C. Williamson. Superkernels. In *Advances in Neural Information Processing Systems 15*, 2003.

[22] B. Schölkopf, A. Smola, and Klaus-Robert Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10:1299–1319, 1998.

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

[24] W. Y. Tan and R. P. Gupta. On approximating a linear combination of central Wishart matrices with positive coefficients. *Communications in Statistics: Theory and Methods*, 12(22):2589–2600, 1983.

[25] M. A. Tanner and W. H. Wong. The calculation of posterior distributions by data augmentation (with discussion). *Journal of the American Statistical Association*, 82(398):528–550, 1987.

[26] K. Tsuda, S. Akaho, and K. Asai. The *em* algorithm for kernel matrix completion with auxiliary data. *Journal of Machine Learning Research*, 4:67–81, 2003.

[27] V. Vapnik. *Statistical Learning Theory*. John Wiley and Sons, New York, 1998.

[28] Z. Zhang, D. Y. Yeung, and J. T. Kwok. Homotopy kernels, generalized eigenproblem, and Bayesian inference by the Tanner-Wong algorithm. Technical Report HKUST-CS03-11, Department of Computer Science, Hong Kong University of Science and Technology, 2003.

[29] Z. Zhang, D. Y. Yeung, and J. T. Kwok. Probabilistic kernel matrix learning. Technical Report HKUST-CS03-09, Department of Computer Science, Hong Kong University of Science and Technology, 2003. Available from ftp://ftp.cs.ust.hk/pub/techreport/03/tr03-09.ps.gz.