Neurocomputing 74 (2011) 1095–1101

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Non-uniform multiple kernel learning with cluster-based gating functions

Yadong Mu [a,b,*], Bingfeng Zhou [b]

[a] Department of Electrical and Computer Engineering, National University of Singapore, 117576 Singapore, Singapore
[b] Institute of Computer Science and Technology, Peking University, Beijing 100871, PR China

ARTICLE INFO

Article history:
Received 3 September 2009
Received in revised form
19 September 2010
Accepted 22 November 2010
Communicated by J. Kwok
Available online 22 December 2010

Keywords:
Kernel based learning
Multi-kernel learning
Graph embedding

ABSTRACT

Recently, multiple kernel learning (MKL) has gained increasing attention due to its empirical superiority over traditional single kernel based methods. However, most of state-of-the-art MKL methods are "uniform" in the sense that the relative weights of kernels keep fixed among all data.

Here we propose a "non-uniform" MKL method with a data-dependent gating mechanism, i.e., adaptively determine the kernel weights for the samples. We utilize a soft clustering algorithm and then tune the weight for each cluster under the graph embedding (GE) framework. The idea of exploiting cluster structures is based on the observation that data from the same cluster tend to perform consistently, which thus increases the resistance to noises and results in more reliable estimate. Moreover, it is computationally simple to handle out-of-sample data, whose implicit RKHS representations are modulated by the posterior to each cluster.

Quantitative studies between the proposed method and some representative MKL methods are conducted on both synthetic and widely used public data sets. The experimental results well validate its superiorities.

1. Introduction

For real-world classification tasks, the input data are typically not linearly separable due to the highly complex data structure or noises. Kernel-based algorithms [1] such as support vector machine (SVM) retain linear separability by mapping original data into a high-dimensional Hilbert space $\mathcal{H}$ via a mapping function $\phi(\cdot)$. An attractive property of kernelized algorithms is that we need only the so-called *Gram matrix* $G=[\mathbb{K}(x_i,x_j)]$ rather than explicit mapping functions, where $\mathbb{K}(x_i,x_j)$ corresponds to the inner product of $\phi(x_i)$ and $\phi(x_j)$ in $\mathcal{H}$.

Recent studies erose much interest in multiple kernel learning (MKL), *i.e.,* utilizing multiple heterogeneous kernels simultaneously. For classification or regression tasks with multi-channel or multi-kernel inputs, a uniform weighting scheme may obtain good performance, however, it may be far from saying "optimal". In contrast, MKL provides an elegant solution for parameter selection. Taking SVM as an example, traditional cross-validation approach is straightforward yet computationally expensive. While from an MKL standpoint, valid parameter values correspond to distinct kernels, whose optimal combination can be determined in a round. Performances in terms of accuracy or regression residual obtained by cross validation and MKL are comparable, yet the latter is much more efficient.

A commonly used kernel combination strategy in MKL is the linear additive one, i.e.,

$$G(i,j)=\sum_{p=1}^{M}\theta_p G_p(i,j),\ \ \theta_p\geq 0,\ \ \forall p,\quad \sum_{p=1}^{M}\theta_p=1, \tag{1}$$

where M is the number of kernels. For binary classification, given a coefficient vector $\alpha\in\mathbb{R}^N$ (N is the number of training data) and labels $\{y_i\}, i=1\ldots N$, each datum is transformed as below for further investigation, i.e.,

$$\mathcal{F}(x_i)=\sum_{j=1}^{N}G(i,j)y_j\alpha_j=\sum_{j=1}^{N}y_j\alpha_j\sum_{p=1}^{M}\theta_p G_p(i,j). \tag{2}$$

The seminal work in [2] done by Lanckriet et al. pioneered the works of linearly combining heterogenous kernels, where the MKL problem was formulated as a semi-definite program (SDP) and was later reformulated by Bach et al. in [3] as a QCQP (quadratically constrained quadratic programming) problem. In [4], the MKL problem was formulated with max-margin criterion as a semi-infinite linear program (SILP), which was finally efficiently solved based on a standard SVM implementation. Later, an MKL method with LDA-style loss functions was investigated by Ye et al. in [5], where SDP based formulation was proposed. After that, Ye et al. [6] further relaxed the SDP form into a QCQP form for acceleration,

* Corresponding author at: Department of Electrical and Computer Engineering, National University of Singapore, 117576 Singapore, Singapore.
E-mail address: elemy@nus.edu.sg (Y. Mu).

0925-2312/$ - see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.neucom.2010.11.001

however, it is not general and cannot handle regression problems, which was further solved in [7].

All the above-mentioned MKL methods are "uniform", namely the weight assigned to each kernel is spatially constant all over an implicit Hilbert space. Such a treatment ignores the possible disparity of data structure in different spatial areas, which motivates our work about "non-uniform" MKL here.

Several works following this idea exist. For example, in [8,9], the authors demonstrated that assigning spatially varying weights to samples within the same kernel-induced space may enhance classification accuracy. For the consideration of feasibility, these methods usually made strong assumptions to control kernel weights' spatial variation, *e.g.*, the log-linear form in [9]. In the work of group-sensitive MKL (GS-MKL) [10], similar idea as in this paper was experimented on the image classification task. Image data sets such as Caltech-101 or VOC 2007 are clustered into groups in each different category by a pre-processing step. For parameter estimation, they employ an alternating optimization scheme between kernel weights and supporting vectors. This work was strongly motivated by the locality property demonstrated in numerous computer vision data sets, where assigning the images from different *sub-class* or *group* is supposed to result in better discriminating ability. However, this method is known to suffer from low convergence speed and parameter-sensitivity. To overcome these aforementioned limitations, here we propose a novel non-uniform MKL method by directly exploiting data locality, rather than imposing ad hoc assumptions.

The rest of this paper is organized as follows: the motivation is explained in Section 2. The overview of the proposed algorithm can be found in Section 3 and its two key optimizing stages are described in Sections 4 and 5 respectively. Finally, extensive evaluations on several datasets are provided in Section 7.

2. Motivation

In Fig. 1, we plot the evolution line of various MKL algorithms. Most existing MKL methods assume that RKHS-related kernel weights remain constant for all data (*i.e.*, space-level), while algorithms like *LocalizedMKL* [9] seek kernel weights that are data-dependent and locally smooth (*i.e.*, sample-level). Although sample-level non-uniform methods give the largest flexibility, in practice typically relaxations are introduced to enhance tractability. An example is the *LocalizedMKL* algorithm, where the linear gating model spatially varies sample's weights along specific direction in the feature space.

However, existing methods suffer from non-linear heterogeneous structures lying in the data. Fig. 2 shows such a toy data set for binary classification, which is named 4-XmasStar. The data consist of four regularly arranged components, each of which contains two star-like shapes (ChrismasStar) generated from either the positive class or negative class, as shown in Fig. 2(a). Obviously the data points can be naturally grouped into four clusters based on mixture-of-Gaussian model, as seen in Fig. 2(b). Particularly, the averaged distances between two adjacent stars in each of the four cluster significantly varies (*i.e.*, 0.2, 0.8, 1.4, and 2.0, respectively. All the stars have a standard variation of 0.3 along the radial direction). In the context of MKL, assume the adopted kernels are five Gaussian kernels $K_\sigma(x_i,x_j)=\exp(-\|x_i-x_j\|^2/\sigma^2)$ with σ equal to 0.01, 0.1, 1, 10, and 100, respectively. Intuitively, the clusters with high inter-star gap prefer larger σ, and the kernel weights are supposed to be gradually changed from cluster-1 to cluster-4 in Fig. 2(b). Fig. 2(c) shows the failure of *LocalizedMKL* using five linear gating functions, where the yellow-color isolines correspond to the decision boundaries, and each unique polygonal region shaped by the pink lines is dominated by a specific kernel. Intuitively, the estimation from *LocalizedMKL* is not in accord with the expectations (for example, cluster-2 and cluster-3 are supposed to both have larger weights on the kernel with $\sigma=0.1$ yet cannot be achieved by linear gating functions. See Section 7.1 for a better solution), which sketches the difficulties for existing sample-level MKL methods to distinguish these kinds of heterogeneous structures.

To overcome above drawbacks and build MKL algorithms which make better tradeoff between flexibility and tractability, here we propose a novel, practical MKL method on the cluster level. Our main observation lies in that data from the same cluster tend to manifest similar properties, thus the intra-cluster weights can be approximately uniform while kernel weights are allowed to vary over clusters. The example of 4-XmasStar is quantitatively revisited in the experimental section.

3. Cluster-based non-uniform MKL

3.1. Notations

Before proceeding, let us first clarify some frequently used notations in this paper. Denote data space as $\mathcal{X}$ (usually a subspace of $\mathbb{R}^d$) and data matrix as $X=[x_1,\ldots,x_N]$, where each column vector $x_i\in\mathcal{X}$. Suppose we expect to fuse information extracted from M heterogeneous channels. The standard MKL algorithms firstly map the original data into high-dimensional feature spaces via M mapping functions $\phi_p:\mathcal{X}\mapsto\mathcal{H}_p$, where $p=1,\ldots,M$ is the index of sub-space and $\mathcal{H}_p$ is the p-th RKHS (maybe of infinite dimension), and then concatenate them to obtain the global Hilbert space $\mathcal{H}$. Denote the Gram matrix in the p-th Hilbert space as G_p, which is positive semi-definite and the (i,j)-th element can be calculated via a kernel function $\mathbb{K}_p:\mathcal{X}\times\mathcal{X}\mapsto\mathbb{R}$. Other notations can be found in Table 1.

As mentioned in Section 2, the proposed method relies on cluster structures in data space. As the initial step, a soft clustering procedure is undertaken in $\mathbb{R}^d$ to form C clusters (also possible to perform clustering in kernel-induced spaces, however, note that it is another challenging problem). Denote the probability that x_i belongs to the c-th cluster as γ_{ic}. Piling all γ_{ic} together, we can obtain the partition matrix $\Gamma=(\gamma_{ic})\in\mathbb{R}^{N\times C}$ with $0\le\gamma_{ic}\le 1,\forall i,c$, and $\sum_c\gamma_{ic}=1,\forall i$.

Rather than associating a unique θ_p to the whole $\mathcal{H}_p$ as in traditional MKL settings, we instead assign θ_{cp} to the c-th cluster in

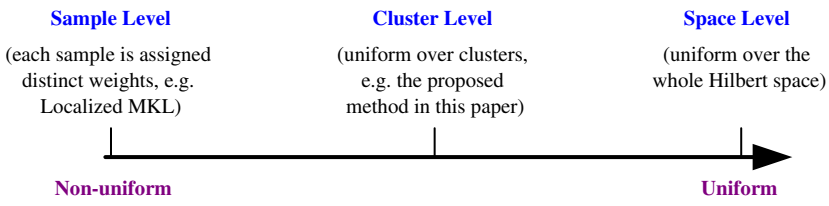

Fig. 1. Illustration for the evolution of various MKL methods.

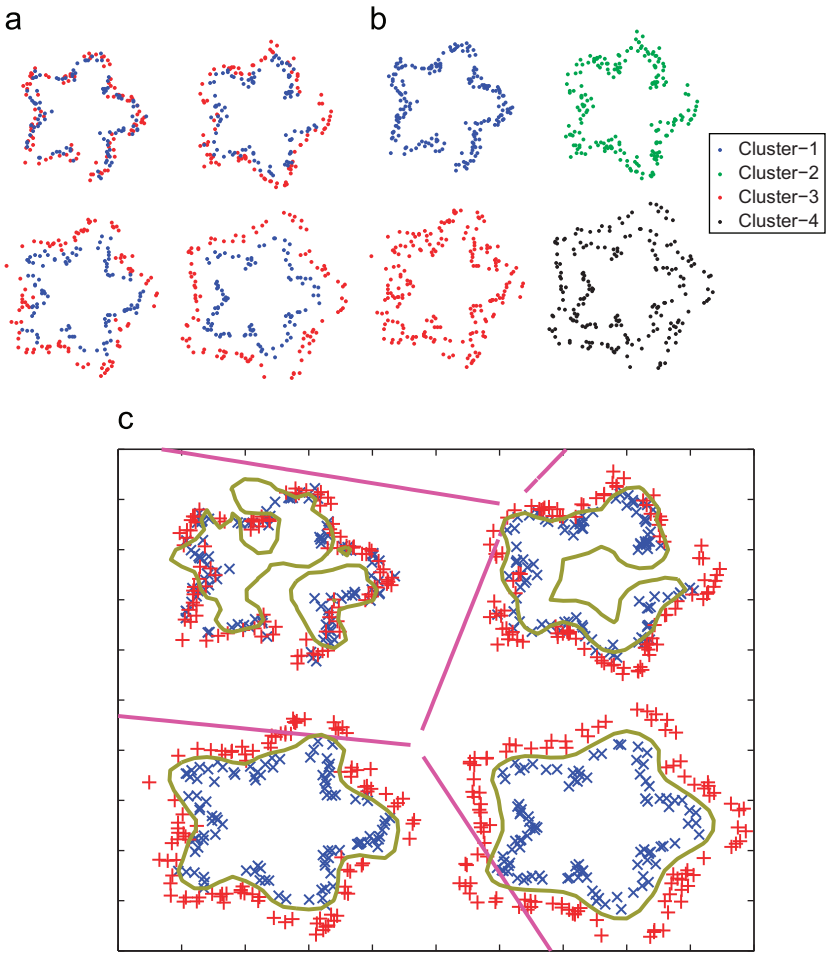

Fig. 2. Experimental results on 4-XmasStar data set. (a) Groundtruth. (b) Cluster structures (cluster number = 4). (c) Results from *LocalizedMKL*. See text for more explanation.

Table 1
Look-up table for notations.

N,M,C	The numbers of data points, heterogeneous kernels and clusters
d	The dimension of original data space
X	Data matrix in $\mathbb{R}^{d\times N}$
Γ	Partition matrix obtained via clustering in original data space
Θ	Kernel weight matrix in $\mathbb{R}^{C\times M}$
$\mathbb{A}$	Projection matrix in graph embedding
W,L,B	Similarity, Laplacian and constraint matrices in graph embedding
S_L^z,S_B^z	Data matrices calculated from L, B, respectively, by treating kernel weight matrix Θ as known
S_L^θ,S_B^θ	Similar to S_L^z,S_B^z

$\mathcal{H}_p$. In this way it is possible to spatially vary kernel weights. Finally we get a *weight matrix* $\Theta\in\mathbb{R}^{C\times M}$ as follows:

$$\Theta=\begin{bmatrix}\theta_{11} & \cdots & \theta_{1M}\\ \vdots & \ddots & \vdots\\ \theta_{C1} & \cdots & \theta_{CM}\end{bmatrix}. \tag{3}$$

To guarantee a valid composite kernel, let $\forall c,p,\ 0\le\theta_{cp}\le 1$ and $\forall c,\ \sum_{p=1}^{M}\theta_{cp}=1$. Throughout this paper we will use the abbreviation Θ_{*p} to denote the p-th column vector of Θ while let Θ_{c*} correspond to the c-th row vector. Our proposed cluster-based gating scheme can be then described as follows: let $\eta_p(x_i)$ denote the data-dependent gating value for x_i in the p-th RKHS. For simplicity, we adopt the linear form $\eta_p(x_i)=\Gamma_{i*}\Theta_{*p}$, and calculate the composite kernel according to

$$G(i,j)=\sum_{p=1}^{M}\eta_p(x_i)\eta_p(x_j)G_p(i,j). \tag{4}$$

It can be easily verified that $\forall i,\ \sum_p\eta_p(x_i)=1$. For clarity, we introduce another notation Λ_p for the p-th RKHS, $\Lambda_p(i,j)=\eta_p(x_i)\eta_p(x_j)$, and then $G=\sum_p\Lambda_p\otimes G_p$, where $\otimes$ denotes the element-to-element matrix multiplication. However, $\Lambda_p(i,j)=\eta_p(x_i)\eta_p(x_j)$ will result in an optimization problem with 4th-order terms related to θ_{ij}. Although it can be relaxed into 2nd-order using the RLT (reformulation linearization technique) trick, however, the large number of parameters ($C^2\times M$ in all) tends to overfit. Hence in practice we adopt the approximation

$\Lambda_p(i,j)=\eta_p(x_i)+\eta_p(x_j)$, which comprises only $C\times M$ parameters. Although not strictly positive semi-definite, it proves to work well in our evaluations.

3.2. Graph embedding (GE)

Targeting a general (rather than specific) learning algorithm, we formulate the non-uniform MKL problem under the graph embedding framework [11]. It is already known that many classical dimensionality algorithms, including principle component analysis (PCA), linear discriminant analysis (LDA), ISOMAP, LLE, locality preserving projections (LPP), can be regarded as special cases of graph embedding. For clear introduction, we will first formulate the 1D case, and then extend it to multi-dimensional.

Treating data as nodes in a graph $\mathcal{G}$. The relationship between any data pairs can be described by a *similarity matrix* $W\in\mathbb{R}^{N\times N}$. For the 1D case, GE seeks $\alpha\in\mathbb{R}^{N\times 1}$ maximally sustaining the relationship defined in W, *i.e.*,

$$\begin{aligned}\alpha^* &= \arg\min_{\alpha^T XBX^T\alpha=1}\sum_{i,j=1}^{N}\|\alpha^T x_i-\alpha^T x_j\|^2 w_{ij}\\ &= \arg\min_{\alpha^T XBX^T\alpha=1}\alpha^T XLX^T\alpha,\end{aligned}\tag{5}$$

where the superscript T denotes matrix transposition, $L=D-W$ is the *Laplacian matrix* and the diagonal matrix D is calculated via $\forall i, D_{ii}=\sum_{j\neq i}w_{ij}$. B is the *constraint matrix* which may be also a Laplacian matrix from a so-called penalty graph. Different dimension reduction (DR) methods correspond to different choices of L and B. Refer to [11] for more instances of L and B.

Extension to multi-dimensional case is straightforward. The aim here is $\mathbb{A}=[\alpha_1,\ldots,\alpha_K]$ rather than a single projection vector α, where K is the targeted low dimensionality. We adopt a criterion in the trace-ratio form, which is in spirit the same to Problem (5):

$$\begin{aligned}&\underset{\mathbb{A}^*,\Theta^*}{\operatorname{argmax}}\ \frac{\operatorname{Trace}(\mathbb{A}^T GBG\mathbb{A})}{\operatorname{Trace}(\mathbb{A}^T(GLG+\gamma G)\mathbb{A})+\gamma_\theta\mathcal{E}(\Theta)}\\ &s.t.\quad \mathbb{A}^T\mathbb{A}=I_K,\end{aligned}\tag{6}$$

where I_K is the identity matrix in $\mathbb{R}^{K\times K}$ and γ is a positive parameter of the Tikhonov regularization term $\mathbb{A}^T G\mathbb{A}$ to tradeoff between empirical accuracy and model complexity. $\mathcal{E}(\Theta)$ is the prior term on Θ whose strength is controlled by γ_θ and it is easy to verify that the denominator is equivalent to $\operatorname{Trace}(\mathbb{A}^T(GLG+\gamma G+(1/K)\gamma_\theta\mathcal{E}(\Theta)I_K)\mathbb{A})$.

3.3. Regularizing Θ

One merit of the proposed criterion is the possibility to incorporate regularization term about kernel matrix Θ, which is difficult, even computationally forbidden for the max-margin based methods such as in [4].

Here is an example: sometimes inter-cluster smoothness in the same RKHS is preferred, *i.e.*, for p-the RKHS, $\forall m,n,m\neq n$, the discrepancy between Θ_{mp} and Θ_{np} will trigger an extra penalty value. Formally speaking, denote averaged weight of p-th RKHS as μ_p, the penalty term for Θ can be expressed as $\mathcal{E}(\Theta)=\sum_{p=1}^{M}\sum_{c=1}^{C}(\Theta_{cp}-\mu_p)^2$.

3.4. Algorithm pipeline

To solve the problem in (6) is a non-trivial task due to the tight coupling of $\mathbb{A}$ and Θ and non-negative constraints of Θ. One possible solution is to decouple $\mathbb{A}$ and Θ via alternating optimization. Following this idea, here we propose a two-stage optimization procedure, where initial guesses gradually converge to their optimal values by alternating between two stages:

1. Updating $\mathbb{A}$ with Θ fixed (see Section 4).
2. Updating Θ with $\mathbb{A}$ fixed (see Section 5).

4. Stage one: update $\mathbb{A}$ with Θ fixed

With Θ known, updating the projection matrix $\mathbb{A}$ becomes rather straightforward. Note that in this case both $GLG+\gamma G+(1/K)\gamma_\theta\mathcal{E}(\Theta)I_K$ and GBG (denote as S_L^α,S_B^α respectively for the purpose of brevity) in Eq. (6) can be estimated, *e.g.*, for S_B^α, we have

$$\mathbb{A}^T GBG\mathbb{A}=\mathbb{A}^T\underbrace{\left(\sum_{p=1}^{M}\sum_{q=1}^{M}(\Lambda_p\otimes G_p)B(\Lambda_q\otimes G_q)\right)}_{\triangleq S_B^\alpha}\mathbb{A}\tag{7}$$

We can get the optimal $\mathbb{A}^*$ under current situation by solving the following trace-ratio optimization problem with orthogonal constraints:

$$\mathbb{A}^*=\arg\max_{\mathbb{A}^T\mathbb{A}=I_k}\frac{\operatorname{Trace}(\mathbb{A}^T S_B^\alpha\mathbb{A})}{\operatorname{Trace}(\mathbb{A}^T S_L^\alpha\mathbb{A})}\tag{8}$$

This type of optimization problem has been well studied in several prior works. Here we simply adopt the approach proposed in [12] due to its simplicity and empirical success.

5. Stage two: update Θ with $\mathbb{A}$ fixed

5.1. Reformulation

Optimizing Θ with $\mathbb{A}$ fixed, however, proves far more complicated than the problem discussed in Section 4. It can be verified that given $\mathbb{A}$, both $\operatorname{Trace}(\mathbb{A}^T GLG\mathbb{A})+\gamma_\theta\mathcal{E}(\Theta)$ and $\operatorname{Trace}(\mathbb{A}^T GBG\mathbb{A})$ can be expressed as the summations of M^2 quadratic terms, e.g.

$$\operatorname{Trace}(\mathbb{A}^T GBG\mathbb{A})=\sum_{p=1}^{M}\sum_{q=1}^{M}\Theta_{*p}^T S_B^\theta(p,q)\Theta_{*q},\tag{9}$$

$$\operatorname{Trace}(\mathbb{A}^T GLG\mathbb{A}))+\gamma_\theta\mathcal{E}(\Theta)=\sum_{p,q=1}^{M}\Theta_{*p}^T S_L^\theta(p,q)\Theta_{*q},\tag{10}$$

where we use $S_B^\theta(p,q),S_L^\theta(p,q)\in\mathbb{R}^{C\times C}$ to denote the data matrices related to column vectors Θ_{*p} and Θ_{*q}, which can be computed if given B (or L), $\mathbb{A}$,Λ_p,Λ_q and Gram matrices G_p, G_q. For the derivation convenience, we further introduce a new variable $\tilde{\Theta}=(\Theta_{*1}^T\ \Theta_{*2}^T\ \cdots\ \Theta_{*M}^T)^T\in\mathbb{R}^{MC\times 1}$, which can be regarded as the results after performing the "matrix-to-vector" concatenation operation to original Θ. The regularization term, *i.e.*, $\gamma\mathbb{A}^T G\mathbb{A}$, can be transformed into a linear term $f^T\tilde{\Theta}$. In this way we obtain an equivalent form of the original problem, i.e.,

$$\begin{aligned}&\underset{\tilde{\Theta}}{\operatorname{argmax}}\ \frac{\tilde{\Theta}^T S_B^\theta\tilde{\Theta}}{\tilde{\Theta}^T S_L^\theta\tilde{\Theta}+f^T\tilde{\Theta}}\\ &s.t.\quad \forall i,\quad 0\le\tilde{\Theta}_i\le 1,\\ &\qquad \forall c,\quad \sum_p\Theta_{cp}=1,\end{aligned}\tag{11}$$

where S_L^θ (or S_B^θ) is the matrix obtained by piling all sub-matrices $\forall p,q$, $S_L^\theta(p,q)$ (or $S_B^\theta(p,q)$) according to (p,q)-indexed 2D grid, *i.e.*,

$$S_L^\theta=\begin{bmatrix}S_L^\theta(1,1) & \cdots & S_L^\theta(1,M)\\ \vdots & \ddots & \vdots\\ S_L^\theta(M,1) & \cdots & S_L^\theta(M,M)\end{bmatrix}\in\mathbb{R}^{CM\times CM}.$$

Problem (11) can be regarded a special case of graph embedding, *i.e.*, with non-negative constraint and embedded to 1D space (since $\tilde{\Theta}$ is a vector). It is challenging to solve owing to the intrinsic non-convex property. However, we argue that a local optimum can be iteratively approached via solving a series of relatively simple trace-difference sub-problems (rather than non-negative trace-ratio itself), as described below.

Algorithm 1. Optimize Θ with $\mathbb{A}$ fixed

Initialize $\tilde{\Theta}^{(0)}$ and $\lambda^{(0)}$.
for t = 1 to $T_{\max}$ **do**
(1) Solve the following trace-difference sub-problem

$$\begin{aligned} \underset{\tilde{\Theta}}{\text{argmax}} \quad & \tilde{\Theta}^T(S_B^\theta - \lambda^{(t-1)} S_L^\theta)\tilde{\Theta} - \lambda^{(t-1)} f^T \tilde{\Theta} \\ s.t. \quad & \forall i, \quad 0 \le \tilde{\Theta}_i \le 1, \\ & \forall c, \quad \sum_{p=1}^{M} \Theta_{cp} = 1, \end{aligned}$$

(2) Update $\lambda^{(t)} = \tilde{\Theta}^T S_B^\theta \tilde{\Theta} / (\tilde{\Theta}^T S_L^\theta \tilde{\Theta} + f^T \tilde{\Theta})$.
(3) Calculate the residual $\Delta\lambda = \lambda^{(t)} - \lambda^{(t-1)}$. Break if $\Delta\lambda$ is small enough.
end for
Output $\tilde{\Theta}^* = \tilde{\Theta}^{(t)}$.

5.2. Pairwise element updating (PEU)

The initial values for $\tilde{\Theta}$ is rather rough and need further refinement. In this subsection we describe a simple yet effective method for the trace-difference sub-problem defined in Algorithm 1, named pairwise element updating (PEU).

The basic idea is as follows: in each iteration, we randomly select two indices k_1 and k_2 from $\tilde{\Theta}$ ($k_1 \neq k_2$) so that there exists a cluster index c with $\tilde{\Theta}_{k_1} \in \Theta_{c*}$ and $\tilde{\Theta}_{k_2} \in \Theta_{c*}$ (in this way, the major constraint on $\tilde{\Theta}$, *i.e.*, $\forall c$, $\sum_{p=1}^{M} \Theta_{cp} = 1$, still holds after updating). Based on k_1 and k_2, a search direction e_k is computed as follows:

$$e_k = \left(\overbrace{0,\ldots,0,1}^{k_1 \text{ elements}}, 0, \ldots, 0, \underbrace{-1}_{k_2 \text{ elements}}, 0, \ldots, 0 \right)^T \tag{12}$$

The role that e_k plays is similar to the candidate gradient direction in Newton method. Specifically, we simply adopt the following linear updating rule:

$$\tilde{\Theta}^{(t)} = \tilde{\Theta}^{(t-1)} + \xi e_k, \tag{13}$$

where ξ is an unknown coefficient to estimate. It can be verified that the original problem can be transformed to a relatively simpler quadratic program (QP), as follows:

$$\begin{aligned} \underset{\xi}{\text{argmax}} \quad & \xi(\xi e_k^T + 2\tilde{\Theta})(S_B^\theta - \lambda S_L^\theta) e_k - \lambda \xi f^T \tilde{\Theta} \\ s.t. \quad & 0 \le \tilde{\Theta}_{k_1} + \xi \le 1, \\ & 0 \le \tilde{\Theta}_{k_2} - \xi \le 1. \end{aligned} \tag{14}$$

Further investigation on Problem (14) will discover that there exists removable redundancy in its objective function. Using the abbreviations $a^\lambda = e_k^T(S_B^\theta - \lambda S_L^\theta)e_k, b^\lambda = \tilde{\Theta}^T(S_B^\theta - \lambda S_L^\theta)e_k - \frac{1}{2}\lambda f^T e_k$, we have an equivalent bounded quadratic optimizing problem:

$$\begin{aligned} \underset{\xi}{\text{argmax}} \quad & \overbrace{a^\lambda \xi^2 + 2b^\lambda \xi}^{\triangleq \mathcal{F}_\lambda^\theta(\xi)} \\ s.t. \quad & \max(-\tilde{\Theta}_{k_1}, \tilde{\Theta}_{k_2} - 1) \le \xi \le \min(1 - \tilde{\Theta}_{k_1}, \tilde{\Theta}_{k_2}). \end{aligned} \tag{15}$$

Note that actually closed-form solution exists for this problem, *i.e.*,

$$\xi^* \in \left\{ -\frac{b^\lambda}{a^\lambda}, \max(-\tilde{\Theta}_{k_1}, \tilde{\Theta}_{k_2} - 1), \min(1 - \tilde{\Theta}_{k_1}, \tilde{\Theta}_{k_2}) \right\}.$$

The whole procedure is summarized as in Algorithm 2.[1]

Algorithm 2. Solve the trace-difference sub-problem

while TRUE **do**
1. Randomly sampling index k_1, and sampling index k_2 from a valid index set determined by k_1.
2. Perform pairwise element updating (PEU) for k_1 and k_2.
3. Update λ, and break if the residual value $\Delta\lambda$ is small enough.

end while

6. Complexity and convergence

Regarding the time complexity, the computational complexity of the first stage in Section 4 consists of two parts, i.e., the calculation of S_B^α, S_L^α and the trace-ratio optimization problem. For the former, each data matrix will require $O(M^2 N^3)$ multiplication operations and $O(M^2 N^3)$ addition operations. While for the latter, solving the trace-ratio problem relies on routines that compute largest n eigenpairs. Efficient Krylov subspace based methods such as implicitly restarted Arnoldi iteration (IRAI) are applicable. In implementation, we adopt the built-in EIGS function in Matlab for it.

While for the PEU algorithm, the computational overhead mainly comes from the calculation of S_B^θ and S_L^θ, both of which are matrices in $\mathbb{R}^{CM \times CM}$ and share similar complexity to S_B^α and S_L^α. Also note that the calculations of a^λ and b^λ in each iteration are on the order of $O(CM)$ rather than $O(C^2 M^2)$ due to the fact that e_k only contains two non-zero elements.

About the convergence of the proposed two-stage algorithm, we first introduce two lemmas about the convergence property during each stage.

Lemma 6.1. *For Problem defined in* (8), *there exists numerical approach to find its global optimum. Refer to* [12] *for detailed proof.*

Lemma 6.2. *The element-wise updating procedure for Problem* (15) *either monotonically increases the trace-ratio λ or reaches its local optimum $(\tilde{\Theta}^*, \lambda^*)$. Denote $b_{min} = \max(-\tilde{\Theta}_{k_1}, \tilde{\Theta}_{k_2} - 1)$ and $b_{max} = \min(1 - \tilde{\Theta}_{k_1}, \tilde{\Theta}_{k_2})$. Note that $\mathcal{F}_\lambda^\theta|_{\xi=0} = 0$ and the objective function is in quadratic form, thus there is $\max(\mathcal{F}_\lambda^\theta|_{\xi=b_{min}}, \mathcal{F}_\lambda^\theta|_{\xi=b_{max}}, \mathcal{F}_\lambda^\theta|_{\xi=-b/a}) \ge 0$ if $-b/a \in [b_{min}, b_{max}]$ or otherwise $\mathcal{F}_\lambda^\theta|_{\xi=b_{min}} \cdot \mathcal{F}_\lambda^\theta|_{\xi=b_{max}} \le 0$. In either case before convergence, $\exists \eta \in [b_{min}, b_{max}]$ such that $\mathcal{F}_\lambda^\theta|_{\xi=\eta} \ge 0$, and thus the conclusion holds.*

Based on the above lemmas, we can reach the conclusion that the iterative two-stage non-uniform MKL method proposed in this paper generally converges to a locally optimal solution after infinitely many steps.

7. Experiment

To validate the effectiveness of the proposed algorithm, we implement the proposed cluster oriented MKL in Matlab and conduct various experiments both on toy data set and UCI machine learning repository. In all experiments we adopt one of the most

[1] The generation scheme for k_1 and k_2 is a totally random one, however, functional gradient based heuristic can be incorporated for acceleration purpose.

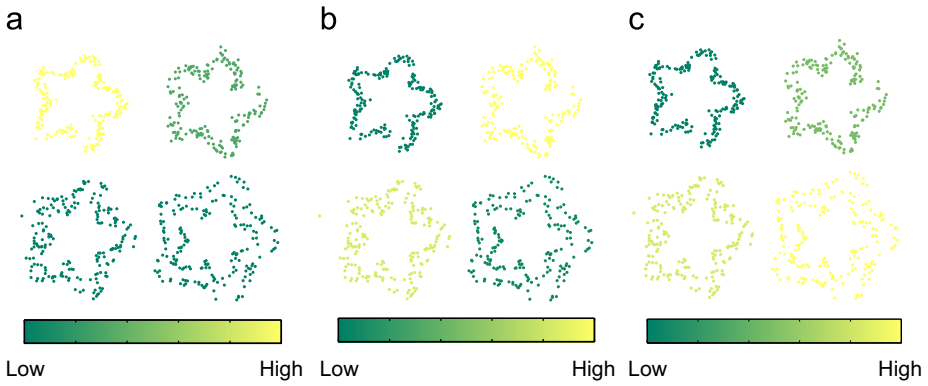

Fig. 3. Visualization of each cluster's relative weights in different RKHS. The three figures from left to right correspond to kernel-induced Hilbert spaces of $K_{0.01}$, $K_{0.1}$ and K_1 respectively. We do not display K_{10} and K_{100}'s since they are never chosen by any of the clusters (thus zero weights).

widely used graph embedding algorithm—linear discriminant analysis (LDA) for classification task. For each data set, 10 random splits into training (typically 30% of all instances) and test data (the rest instances) are generated. Regularization parameter in LDA is fixed as 10^{-3} without optimal selection by cross-validation.

7.1. Toy data set

We illustrate the effect of adaptive kernel selection for the proposed method on the 4-XmasStar toy data set. The original data set and its cluster structure can be seen in Fig. 2. Our aim is to distinguish the different intrinsic sub-structures of clusters (consequently different kernel combinations), which are previously ignored in uniform MKL methods and cannot be well captured in *LocalizedMKL*. As aforementioned, the kernels we adopt are five Gaussian kernels $K_\sigma(x_i,x_j)=\exp(-\|x_i-x_j\|^2/\sigma^2)$ with σ equal to 0.01, 0.1, 1, 10, and 100 respectively. The kernel weight matrix Θ obtained by our proposed algorithm is as follows:

$$\Theta=\begin{pmatrix} 0.84 & 0.16 & 0 & 0 & 0 \\ 0.28 & 0.33 & 0.39 & 0 & 0 \\ 0 & 0.31 & 0.69 & 0 & 0 \\ 0 & 0.17 & 0.83 & 0 & 0 \end{pmatrix}, \tag{16}$$

where Θ_{cp} denotes the c-th cluster' weight in the p-th kernel-induced Hilbert space. It can be seen that the results are consistent with the intuition. For example, $K_{0.01}$ dominates cluster-1, and for other three clusters, kernels with larger widths gradually increase their relative weights. For better understanding, the results are visualized in Fig. 3.

7.2. Benchmark data set

We also select eight two-class classification problems (see Table 2) from UCI machine learning repository [13].[2] For each data set, we split it into training/test data with a ratio of 3:7 and iterate 10 times to reduce the randomness. In all we use four kernels, all belong to the Gaussian kernel family yet differ in the kernel widths, i.e. 0.01σ,0.1σ,σ, and 10σ respectively, where σ is estimated as the squared root of the averaged L_2 distances between all k-nearest neighbor pairs.

The averaged accuracies of each individual kernel and their combination are given in Table 3. We compare three MKL algorithms: QCQP based uniform MKL method [6], *LocalizedMKL*, and our proposed non-uniform algorithm (with cluster number $C=4$ for all data sets). We use the original implementation of *LocalizedMKL* from the authors in [9]. There are two distinct types of gating strategies in *LocalizedMKL*: *softmax* or *sigmoid*. We run with both strategies and report the best.

Table 2
Description for selected UCI data sets, including their names, feature numbers and total instance numbers.

Data set	Feature	Instance
GERMAN	24	1000
HEART	13	303
IONOSPHERE	33	351
LIVER	6	345
SONAR	60	208
SPAMBASE	57	4601
MADELON	500	2600
AUS-CREDIT	14	690

The results can be understood in two aspects: firstly, it can be seen that MKL methods outperform single kernel based ones in almost all cases (6 out of 8), except for data sets on which MKL methods overfit. And secondly, in the eight experiments, ours has slightly better performance compared with uniform MKL and *LocalizedMKL*. In Table 4 we present the performance under different cluster numbers on three UCI data sets. It is shown that in most cases the performance with $C>1$ clusters are better than that of $C=1$ (equivalent to uniform MKL).

8. Conclusion

In this paper we propose a non-uniform method for multiple kernel integration, which is supposed to overcome the limitations of traditional uniform MKL approaches. While regarding computational issues, we design a two-stage alternating optimization algorithm to separate the highly coupled unknown variables and also give its convergence guarantee.

Note that similar to existing non-uniform methods such as *LocalizedMKL*, the proposed method performs well based on some special properties of the data distributions, i.e. the data present spatial locality property and can be accurately described with mixture of clusters. In the future we will relax this assumption and design more general non-uniform MKL algorithms.

[2] http://www.ics.uci.edu/~mlearn/MLRepository.html

Table 3
Performances on eight two-class UCI data sets in terms of classification accuracies for both single kernel and integrated kernel cases.

	LDA-$K_{0.01\sigma}$	LDA-$K_{0.1\sigma}$	LDA-K_{σ}	LDA-$K_{10\sigma}$
GERMAN	70.5 ± 1.2	70.0 ± 1.2	71.4 ± 1.7	74.7 ± 1.5
HEART	53.8 ± 1.2	50.0 ± 4.1	77.8 ± 3.3	82.5 ± 1.7
IONOSPHERE	64.6 ± 1.5	65.3 ± 1.9	93.3 ± 1.2	87.1 ± 1.9
LIVER	58.6 ± 1.7	58.5 ± 2.9	64.3 ± 3.6	62.3 ± 2.8
SONAR	50.8 ± 4.1	50.5 ± 3.6	$\mathbf{78.5 \pm 0.3}$	73.0 ± 4.8
SPAMBASE	65.8 ± 0.5	71.9 ± 0.9	91.1 ± 0.5	88.7 ± 0.8
MADELON	50.6 ± 0.4	50.3 ± 0.2	55.6 ± 0.7	$\mathbf{57.9 \pm 1.1}$
AUS-CREDIT	55.2 ± 1.5	54.8 ± 3.0	85.1 ± 1.0	86.4 ± 0.6

	MKL	*LocalizedMKL*	Our method
GERMAN	72.5 ± 2.6	71.5 ± 1.6	$\mathbf{75.1 \pm 1.8}$
HEART	$\mathbf{83.3 \pm 2.0}$	82.1 ± 1.6	82.8 ± 1.5
IONOSPHERE	$\mathbf{93.8 \pm 0.9}$	86.0 ± 3.1	92.5 ± 1.3
LIVER	57.8 ± 2.1	58.6 ± 3.9	$\mathbf{65.7 \pm 2.1}$
SONAR	73.2 ± 2.8	75.4 ± 2.4	75.6 ± 3.9
SPAMBASE	91.2 ± 0.4	90.5 ± 1.3	$\mathbf{91.8 \pm 0.4}$
MADELON	50.2 ± 0.5	55.2 ± 2.4	56.4 ± 0.8
AUS-CREDIT	86.6 ± 1.2	85.0 ± 1.8	$\mathbf{86.7 \pm 1.0}$

Table 4
Performance under different parameter C on three UCI datasets.

	$C=1$	$C=2$	$C=3$	$C=4$
GERMAN	74.4 ± 1.2	75.0 ± 1.4	75.1 ± 1.8	74.3 ± 1.2
HEART	81.8 ± 1.6	82.5 ± 1.5	82.8 ± 1.5	83.1 ± 2.6
IONOSPHERE	92.5 ± 0.6	92.7 ± 2.2	92.5 ± 1.3	93.9 ± 1.5

Acknowledgement

This work is under the support of a grant from National Science Foundation of China, whose grant no. is NSFC 60973054.

References

[1] B. Scholkopf, A.J. Smola, Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond, MIT Press, Cambridge, MA, USA, 2001.
[2] G.R.G. Lanckriet, N. Cristianini, P.L. Bartlett, L.E. Ghaoui, M.I. Jordan, Learning the kernel matrix with semi-definite programming, in: ICML, 2002, pp. 323–330.
[3] F.R. Bach, G.R.G. Lanckriet, M.I. Jordan, Multiple kernel learning, conic duality, and the smo algorithm, in: ICML, 2004.
[4] S. Sonnenburg, G. Rätsch, C. Schäfer, B. Schölkopf, Large scale multiple kernel learning, Journal of Machine Learning Research 7 (2006) 1531–1565.
[5] J. Ye, J. Chen, S. Ji, Discriminant kernel and regularization parameter learning via semidefinite programming, in: Proceedings of the 24th International Conference on Machine Learning, 2007, pp. 1095–1102.
[6] J. Ye, S. Ji, J. Chen, Learning the kernel matrix in discriminant analysis via quadratically constrained quadratic programming, in: The 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2007, pp. 854–863.
[7] Y.-Y. Lin, T.-L. Liu, C.-S. Fuh, Dimensionality reduction for data in multiple feature representations, in: D. Koller, D. Schuurmans, Y. Bengio, L. Bottou (Eds.), Advances in Neural Information Processing Systems 21, 2009.
[8] D.P. Lewis, T. Jebara, W.S. Noble, Nonstationary kernel combination, in: Proceedings of the 23rd International Conference on Machine Learning, 2006, pp. 553–560.
[9] M. Gönen, E. Alpaydin, Localized multiple kernel learning, in: Proceedings of the 25th International Conference on Machine Learning, 2008, pp. 352–359.
[10] J. Yang, Y. Li, Y. Tian, L. Duan, W. Gao, Group-sensitive multiple kernel learning for object categorization, in: ICCV, 2009.
[11] S. Yan, D. Xu, B. Zhang, H.-J. Zhang, Q. Yang, S. Lin, Graph embedding and extensions: a general framework for dimensionality reduction, IEEE Transactions on Pattern Analysis and Machine Intelligence 29 (1) (2007) 40–51.
[12] H. Wang, S. Yan, D. Xu, X. Tang, T.S. Huang, Trace ratio vs. ratio trace for dimensionality reduction, in: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2007, pp. 1–8.
[13] A. Asuncion, D. Newman, UCI machine learning repository, 2007, URL: 〈http://www.ics.uci.edu/~mlearn/MLRepository.html〉.

Yadong Mu received the B.S. and Ph.D degrees from the Computer Science Department and Institute of Computer Science and Technology (ICST), Peking University, Beijing, China, in 2004 and 2009, respectively. Currently, he is a postdoctoral research fellow in the Department of Electrical and Computer Engineering, National University of Singapore. His research interests include computer graphics, computational photography, computer vision and machine learning.

Bingfeng Zhou was born in 1963, Ph.D., professor. His research interests include graphics simulation of robot kinematics, geometry models and CAD/CAM, color image processing, multimedia system and image special effects, digital image halftone, image based rendering and modeling, virtual reality, etc.