MultiK-MHKS: A Novel Multiple Kernel Learning Algorithm

Zhe Wang, Songcan Chen*, Tingkai Sun

***Abstract*— In this paper, we develop a new effective multiple kernel learning algorithm. First, we map the input data into m different feature spaces by m empirical kernels, where each generated feature space is taken as one view of the input space. Then through borrowing the motivating argument from Canonical Correlation Analysis (CCA) that can maximally correlate the m views in the transformed coordinates, we introduce a special term called Inter-Function Similarity Loss R_{IFSL} into the existing regularization framework so as to guarantee the agreement of multi-view outputs. In implementation, we select the Modification of Ho-Kashyap algorithm with Squared approximation of the misclassification errors (MHKS) as the incorporated paradigm, and the experimental results on benchmark datasets demonstrate the feasibility and effectiveness of the proposed algorithm named MultiK-MHKS.**

***Index Terms*— Multiple kernel learning; Canonical correlation analysis; Regularization Learning; Modified Ho-Kashyap algorithm; Single learning process; Pattern recognition.**

I. Introduction

Kernel-based learning algorithms [22], [26], [30] work through mapping the input data $\mathcal{X}$ into a feature space $\mathcal{F}$, $\Phi : \mathcal{X} \rightarrow \mathcal{F}$, where the mapping Φ is represented by introducing a kernel. In practice, the types and the parameters of the kernels must be selected. In the literature, such a selection is often considered as the open problem of "model selection". For a given application, there may be multiple kernels as the candidates which can possess different types and parameters. The kernel selected from the candidates can yield a model with good performance. Such a selection, equivalently to model selection, can usually be achieved by some methods of optimizing kernels such as Cross Validation (CV) or Leave-One-Out (LOO) [4], [21]. However, these methods are computationally expensive when dealing with a large number of kernel types or parameters. Even the kernel selected by these optimization methods also can not be guaranteed optimal in some cases. Further, since the selected kernel is single and fixed, it can only characterize the geometrical structure of some aspects for the input data and thus not always be fit for the applications which involve multiple, heterogeneous data sources [27].

Recently, a so-called Multiple Kernel Learning (MKL) method [3], [7], [10], [14], [15], [23] have shown the necessity to consider multiple kernels, or the combination of kernels rather than a single fixed kernel. Generally, MKL tries to form an ensemble of kernels so as to be fit for a certain application. It has been proved that MKL can offer some needed flexibility and well manipulate the case that involves multiple, heterogeneous data sources [1], [27], [2]. Since MKL considers multiple kernels, it can be effectively employed for the heterogeneous data sources under the common framework of kernel learning. To a certain extent, MKL also relaxes the model selection about kernels. Lanckriet et al. [15] constructed a convex Quadratically Constrained Quadratic Program (QCQP) by the conic combinations of multiple kernels $K = \sum_i ®_i K_i$ from a library of candidate kernels K_i, and showed that their method can combine multiple possible heterogeneous data sources and moreover emphasize those most useful sources in a given application, such as the genomic data fusion [14]. Then in order to extend Lanckriet et al.'s method to large scale problems, Bach et al. [1] took the dual formulation of QCQP as a Second-Order Cone Programming (SOCP) problem, and Sonnenburg et al. [27], [28] reconstructed QCQP as a semi-infinite linear program that recycles the standard Support Vector Machine (SVM) implementations [30]. On the other hand, Bennett et al. [2] and Bi et al. [3] respectively utilized a boosting approach to achieve the combinations of kernels, and showed that such a combination can incorporate and potentially extract domain knowledge from the heterogeneous sources. Further, de Diego et al. [7], [8], [20] built a kernel matrix from a collection of kernels through quantifying the difference of information among the kernels and their methods have been successfully evaluated for classification.

In this paper, we continue the study on MKL. Different from the existing MKL algorithms which mainly consider the convex combination of multiple kernels, we borrow an argument from Canonical Correlation Analysis (CCA) to develop a new MKL method, whose underlying motivations and contributions are:

- With m kernels, a given input data can be mapped into m feature spaces, where each feature space can be taken as one view of the original input data. Each view is expected to exhibit some geometrical structures of the original data from its own perspective such that all the m views can complement for the subsequent learning task. How to embed such a complementarity into one learning process becomes our aim.

CCA [12] is generally adopted to evaluate the linear correlations between two sets of multi-dimensional variables. It works by finding two basis vectors for two sets of variables such that the correlation between the projections of the variables onto their corresponding basis vectors is mutually maximized. For a

*Corresponding author. Email: s.chen@nuaa.edu.cn

Zhe Wang and Songcan Chen are with Department of Computer Science & Engineering, Nanjing University of Aeronautics & Astronautics, Nanjing, 210016, P.R. China. Tingkai Sun is with College of Computer Science and Technology, Nanjing University of Science and Technology, 210094, P.R. China.

Digital Object Indentifier 10.1109/TPAMI.2007.70786

given application involved two views which both contain the common information, but individually represent in different and specific sets of features, CCA is expected to denoise the individual views and give the common relevant information [9], which is why CCA as a preprocessing step can improve the performance of the subsequent classification algorithm [17]. Thus, considering the characteristic of CCA, we try to exploit CCA so as to embed the complementarity provided by the m views of the input data into one single learning process. However, CCA [12] itself is only fit for two sets of variables and the m here is more than two. Moreover, the generalization of CCA [11] for more than two sets of variables usually leads to a relatively high complexity in seeking its solution since it employs the way by maximizing all the correlations between the projections of the pairwise views, or equivalently, minimizing the objective function $\sum_{k,l=1;k\neq l}^{m} \|S^{(k)}W^{(k)} - S^{(l)}W^{(l)}\|_F^2$ [1] that contains $\frac{m(m-1)}{2}$ terms. It is a complexity that urges us to simplify the formulation so as to form a new CCA called NmCCA. NmCCA makes the projection of each view onto their corresponding basis vectors maximally close to the average projection of all the views, i.e. minimizing the objective function $\sum_{k=1}^{m} \|S^{(k)}W^{(k)} - \frac{1}{m}\sum_{l=1}^{m} S^{(l)}W^{(l)}\|_F^2$ that not only reduces the number of the terms in the objective function from $\frac{m(m-1)}{2}$ [11] to m but also satisfies the inequality $\sum_{k=1}^{m} \|S^{(k)}W^{(k)} - \frac{1}{m}\sum_{l=1}^{m} S^{(l)}W^{(l)}\|_F^2 \leq \sum_{k,l=1;k\neq l}^{m} \|S^{(k)}W^{(k)} - S^{(l)}W^{(l)}\|_F^2$ [2].

• In practice, instead of the general Implicit Kernel Mapping (IKM) [22], [26] in kernel-based methods, we adopt Empirical Kernel Mapping (EKM) [25], [31] that explicitly maps the input data into m different feature spaces (also called m views) by the given m kernels, and then exploit NmCCA to embed the complementarity provided by the m views into a new regularization learning that is expected to effectively improve the generalization of classifiers. The classical regularization framework as an effective and popular method for boosting generalization [5], [16], [30] attempts to obtain the classifier f by minimizing the following function

$$R(f) = R_{emp}(f) + cR_{reg}(f); \quad (1)$$

where $R_{emp}(f)$ is the empirical risk term, $R_{reg}(f)$ represents the regularization term that penalizes the roughness or smoothness of f [5], and $c \geq 0$ is a regularization parameter that controls the trade-off between $R_{emp}(f)$ and $R_{reg}(f)$. Then, with NmCCA that makes the projection of each view onto their corresponding basis vectors maximally close to the average projection of all the views, i.e. minimizing $\sum_{k=1}^{m} \|S^{(k)}W^{(k)} - \frac{1}{m}\sum_{l=1}^{m} S^{(l)}W^{(l)}\|_F^2$, as a result, we construct a new term called Inter-Function Similarity Loss R_{IFSL} and introduce it into $R(f)$ with the purpose to make all the given m classifiers f_l corresponding to the m views of the common labels achieve as much agreement on their outputs as possible. Consequently, we obtain the final decision function F defined as $F = \frac{1}{m}\sum_{l=1}^{m} f_l$ by minimizing the objective function added the new term as follows

$$R(F) = \sum_{l=1}^{m}[R_{emp}(f_l) + c_l R_{reg}(f_l)] + \lambda R_{IFSL}(F); \quad (2)$$

where $R_{IFSL}(F) = \sum_{l=1}^{m}(f_l - \frac{1}{m}\sum_{j=1}^{m} f_j)^2$ plays a role of the agreement penalization on the outputs of multiple f_l, and $\lambda \geq 0$ is a factor controlling the trade-off between $R_{IFSL}(F)$ and $R(f_l)$.

• The base classifier f_l of (2) in fact can be any one coincidental to the learning framework (1). In learning, we focus on the Modification of Ho-Kashyap algorithm with Squared approximation of the misclassification errors (MHKS) proposed by Łeski [16] as the base paradigm f_l mainly due to: 1) MHKS falls into the regularization framework (1); 2) MHKS employs a modification of the gradient descent with a heuristic update-rule and thus it is relatively simple for obtaining the minimizer to the objective function (2); 3) the experiments in Section III have validated that the proposed MKL algorithm MultiK-MHKS in the framework (2) can achieve the convergence within a few training iterations.

• Finally, we should state that from the angle of $R_{IFSL}(F)$, it seems not directly to relate NmCCA. Though $R_{IFSL}(F)$ itself indeed can make the outputs of multiple f_l achieve as much agreement as possible so as to reduce the output variance of the model, in fact, $R_{IFSL}(F)$ is exactly induced through borrowing a motivating argument from NmCCA. NmCCA not only guarantees to correlate among multiple views, but also is an interesting byproduct with a relatively simpler formulation than CCA [11]. Thus NmCCA deserves a separate extended study in future.

The rest of this paper is organized as follows. In Section II, we give the description of the implemental algorithm named MultiK-MHKS in the proposed learning framework (2). In Section III, the experimental results on some benchmark datasets have shown the feasibility and effectiveness of MultiK-MHKS. Finally, both conclusion and future work are given in Section IV.

II. Fusion of NmCCA into Regularization Learning

This section discusses how to introduce the spirit of NmCCA into the regularization learning framework (1) with m kernels. In order to theoretically demonstrate the feasibility of our idea, we select the Modification of Ho-Kashyap algorithm with Squared approximation of the misclassification errors (MHKS) [16] as the learning incorporated paradigm, and then develop a new MKL algorithm named MultiK-MHKS.

[1] The input data $\{x_i\}_{i=1}^{N} \subset \mathbb{R}^d$ can be explicitly mapped into m feature spaces with m kernels (see Section II.B). Then define the mapped data in each view as $\{\Phi_l^e(x_i)\}_{i=1}^{N} \subset \mathbb{R}^{n_l}, l = 1...m$, where n_l is the dimensionality of each feature space. By $S^{(l)} = [\Phi_l^{eT}(x_1); ...; \Phi_l^{eT}(x_N)] \in \mathbb{R}^{N\times n_l}, l = 1...m$, we can get the m views of the input data $\{x_i\}_{i=1}^{N}$ in matrix form $\{S^{(1)}, ..., S^{(m)}\}$. The transformation matrices $W^{(l)} \in R^{n_l\times q}, l = 1...m$ are wanted such that the correlations between all the pairwise views $S^{(k)}$ and $S^{(l)}$ are mutually maximized, i.e. maximizing $\sum_{k,l=1;k\neq l}^{m} Tr(W^{(k)T}S^{(k)T}S^{(l)}W^{(l)})$, where q is the dimensionality of each view after transformation. Further, maximizing $\sum_{k,l=1;k\neq l}^{m} Tr(W^{(k)T}S^{(k)T}S^{(l)}W^{(l)})$ can be transformed into a distance minimization problem $\min \sum_{k,l=1;k\neq l}^{m} \|S^{(k)}W^{(k)} - S^{(l)}W^{(l)}\|_F^2$. If unfolding the latter objective, both the optimization are equivalent and can be solved by singular value decomposition [11].

[2] The inequality can be easily proven by the algebra that the objective function $\sum_{i=1}^{m} s_i \parallel x_i - a \parallel^2$ with the variable a has a unique minimizer $a = \frac{\sum_{i=1}^{m} s_i x_i}{\sum_{i=1}^{m} s_i}$. In this case, $s_i = 1$.

A. MHKS Algorithm

Suppose that there are N labeled training samples $\{(x_i; \ '_i)\}_{i=1}^N$ available, where $x_i \in \mathbb{R}^d$ and its corresponding class label $'_i \in \{+1; -1\}$. In MHKS [16], the decision function

$$g(x_i) = !^T x_i + !_0; \tag{3}$$

is obtained by optimizing the criterion

$$\min_{\omega \in \mathbb{R}^{d+1}, \mathbf{b}_{N\times 1} \geq 0} I = (Y\, ! - \mathbf{1}_{N\times 1} - \mathbf{b}_{N\times 1})^T (Y\, ! - \mathbf{1}_{N\times 1} - \mathbf{b}_{N\times 1}) + c\, !^T !; \tag{4}$$

where $! \in \mathbb{R}^d; !_0 \in \mathbb{R}$ in (3) are the weight vector and the bias respectively, the augmented weight vector $! = [!^T; !_0]^T$, the matrix Y is defined as $Y = [\ '_1 (x_1{}^T; 1); :::; \ '_N (x_N{}^T; 1)]$, $\mathbf{1}_{N\times 1}$ represents the vector of N dimension with all entries equal to 1 and $\mathbf{b}_{N\times 1}$ represents the vector with all entries equal to nonnegative values, the scalar $c \geq 0 \in \mathbb{R}$. In (4), the terms $(Y\, ! - \mathbf{1}_{N\times 1} - \mathbf{b}_{N\times 1})^T (Y\, ! - \mathbf{1}_{N\times 1} - \mathbf{b}_{N\times 1})$ and $!^T !$ correspond to $R_{emp}(f)$ and $R_{reg}(f)$ of (1) respectively, and the c is the regularization parameter. The elaborate description about MHKS can be found in [16].

B. Proposed MultiK-MHKS algorithm

In our method, given N training samples $\{(x_i; \ '_i)\}_{i=1}^N$ with m kernels, we can map each sample x_i, i.e. $\Phi_l : \mathcal{X} \to \mathcal{F}_l^{n_l}; l = 1:::m$ by different kernels from the input space into the m corresponding feature spaces $\{\mathcal{F}_l^{n_l}\}_{l=1}^m$, each of which has n_l dimension. The aim of our method is to work by fully considering all the m feature spaces.

Traditionally, the mapping Φ is *implicitly* represented by specifying a kernel function as the inner product between each pair of samples in the feature space [22], [26]. For the sample set $\{x_i\}_{i=1}^N$, X denotes the $N \times d$ sample matrix where each row is the vector x_i^T, and $K = [ker_{ij}]_{N\times N}$ denotes the $N \times N$ kernel matrix where $ker_{ij} = \Phi(x_i) \cdot \Phi(x_j) = ker(x_i; x_j)$. K is a symmetrical positive-semidefinite matrix. Conversely, the mapping Φ in this paper, is given in an *explicit* form as describe in [31]. If the rank of K is r, the kernel matrix K can be decomposed as

$$K_{N\times N} = Q_{N\times r} \Lambda_{r\times r} Q_{r\times N}^T; \tag{5}$$

where Λ is a diagonal matrix consisting of the r positive eigenvalues of K, and Q consists of the corresponding orthonormal eigenvectors. Thus, the explicit mapping also called the Empirical Kernel Mapping (EKM) in [25], [31], is given as $\Phi^e : \mathcal{X} \to \mathcal{F}^r$

$$x \to \Lambda^{-1/2} Q^T [ker(x; x_1); :::; ker(x; x_N)]^T: \tag{6}$$

Let $B = KQ\Lambda^{-1/2}$, and then the dot product matrix of $\{\Phi^e(x_i)\}_{i=1}^N$ generated by EKM can be calculated as

$$BB^T = KQ\Lambda^{-1/2}\Lambda^{-1/2}Q^T K = K: \tag{7}$$

That is exactly equal to the kernel matrix in the Implicit Kernel Mapping (IKM), and thus the mapped samples respectively generated by EKM and IKM have the same geometrical structure. In [25], [31], it is shown that comparing EKM with IKM, the former is easier to access and easier to study the adaptability of a kernel to the input space than the latter. That is why we select EKM here.

In the MKL problem here, the set of samples $\{x_i\}_{i=1}^N$ is explicitly mapped into the set $\{(\Phi_1^e(x_i); :::; \Phi_l^e(x_i); :::; \Phi_m^e(x_i))\}_{i=1}^N$, where each Φ_l^e, also called one view of the original input space, corresponds to one kernel. Then through borrowing the motivating argument from NmCCA that makes the projection of each view maximally close to the average projection of all the views, we give the fusion of NmCCA with MHKS in the framework (2). Concretely, MHKS is carried out in each view, and meanwhile all the MHKSs are fused into one single process by the spirit of NmCCA. Thus, we can get the following optimization problem:

$$\min_{\substack{\omega_l \in \mathbb{R}^{n_l+1}, \mathbf{b}_l \geq 0; \\ l=1...m}} L = \sum_{l=1}^m [(Y_l\, !_l - \mathbf{1}_{N\times 1} - \mathbf{b}_l)^T (Y_l\, !_l - \mathbf{1}_{N\times 1} - \mathbf{b}_l) + c_l !_l^T !_l] + \ , \sum_{l=1}^m (Y_l\, !_l - \frac{1}{m}\sum_{j=1}^m Y_j\, !_j)^T (Y_l\, !_l - \frac{1}{m}\sum_{j=1}^m Y_j\, !_j); \tag{8}$$

where $Y_l; !_l; !_l; \mathbf{b}_l$ correspond to one MHKS in one view that is determined by the corresponding $\{(\Phi_l^e(x_i); \ '_i)\}_{i=1}^N$. In the right hand side of equation (8), the first term corresponds to the principle of MHKSs in m views. Minimizing the second term characterizes that the outputs of each view $\{(\Phi_l^e(x_i); \ '_i)\}_{i=1}^N$ onto their corresponding weight vector $!_l$ are constrained to be maximally close to the average outputs of all the views, which is induced from NmCCA.

In the framework (2), we have given the final decision function F defined as $F = \frac{1}{m}\sum_{l=1}^m f_l$. Here, the $!_l; l = 1:::m$ need to be solved by minimizing the objective function (8). It can be found that the optimization problem (8) with respect to the single $!_l$ is convex [6] while $!_j; j \neq l$ is fixed. Thus we solve $!_l$ in a sequence $l = 1:::m$, and employ a modification of the gradient descent with a heuristic update-rule for each $!_l$. Now making the gradient of L with respect to $!_l$ be zero, we can obtain

$$!_l = [(1+\ , \frac{m-1}{m}) Y_l{}^T Y_l + c_l \tilde{I}]^{-1} Y_l{}^T (\mathbf{b}_l + \mathbf{1}_{N\times 1} + \ , \frac{1}{m}\sum_{j=1; j\neq l}^m Y_j\, !_j); \tag{9}$$

where $\tilde{I}$ is a diagonal matrix with full 1s except the last element set to zero.

In the lth view, it can be noted that $!_l$ depends on $\mathbf{b}_l$ from equation (9). Then by differentiating L with respect to $\mathbf{b}_l$ and setting the result equal to zero, we can get

$$\mathbf{e}_l = Y_l\, !_l - \mathbf{b}_l - \mathbf{1}_{N\times 1} = \mathbf{0}: \tag{10}$$

From equation (10), the components of $\mathbf{b}_l$ determine the distance from samples to the separating hyperplane, and thus play a similar role to the relaxation variables in SVM [30]. In order to guarantee that the samples are correctly classified in the lth view, the components of $\mathbf{b}_l$ need to be nonnegative. Thus, we employ the iterative algorithm for determining $!_l$ and $\mathbf{b}_l$ analogously to [16]. First, with $\mathbf{b}_l^k$ representing the vector $\mathbf{b}_l$ at the kth iteration, we initialize $\mathbf{b}_l^1 \geq 0$, then keep

Table 1 Algorithm **MultiK-MHKS**

Input: $\{x_i, \varphi_i\}_{i=1}^N$; the m candidate kernels $\{ker_l(x_i, x_j)\}_{l=1}^m$

OutPut: the weight vectors $\omega_l, l = 1...m$

1. Explicitly map $\{x_i\}_{i=1}^N$ into $\{\Phi_1^e(x_i), ..., \Phi_l^e(x_i), ..., \Phi_m^e(x_i)\}_{i=1}^N$ by m kernels as shown in (6).
2. For each view, $Y_l = [\varphi_1(\Phi_l^{eT}(x_1), 1); ...; \varphi_N(\Phi_l^{eT}(x_N), 1)], l = 1...m.$
3. Initialize $\xi > 0, \rho_l > 0, c_l \geq 0, \mathbf{b}_l^1 \geq 0, l = 1...m$ and $\omega_l^1, l = 2...m$; set the iteration index $k = 1$.
4. $\omega_l^k = ((1 + \lambda\frac{m-1}{m})Y_l^T Y_l + c_l\tilde{I})^{-1} Y_l^T (\mathbf{b}_l^k + \mathbf{1}_{N\times 1} + \lambda\frac{1}{m}\sum_{j=1;j\neq l}^m Y_j\omega_j^k), l = 1.$
5. $\mathbf{e}_l^k = Y_l\omega_l^k - \mathbf{b}_l^k - \mathbf{1}_{N\times 1}, l = 1...m.$
6. $\mathbf{b}_l^{k+1} = \mathbf{b}_l^k + \rho_l(\mathbf{e}_l^k + |\mathbf{e}_l^k|), l = 1...m.$
7. $\omega_l^{k+1} = ((1 + \lambda\frac{m-1}{m})Y_l^T Y_l + c_l\tilde{I})^{-1} Y_l^T (\mathbf{b}_l^{k+1} + \mathbf{1}_{N\times 1} + \lambda\frac{1}{m}\sum_{j=1;j\neq l}^m Y_j\omega_j^k), l = 1...m,$

if $\|L^{k+1} - L^k\|_2 > \xi$, then $k = k + 1$, go to Step 5; else stop.

$\mathbf{b}_l^k \geq 0$ at each iteration k, and thus obtain

$$\begin{cases} \mathbf{b}_l^1 & \geq & 0 \\ \mathbf{b}_l^{k+1} & = & \mathbf{b}_l^k + ½_l(\mathbf{e}_l^k + |\mathbf{e}_l^k|) \end{cases} ; \qquad (11)$$

where at the kth iteration, the error vector of the lth view $\mathbf{e}_l^k$ is defined as $\mathbf{e}_l^k = Y_l\,!_l^k - \mathbf{b}_l^k - \mathbf{1}_{N\times 1}$, and the learning rate of the lth view $½_l > 0$. Then $!_l^{k+1}$ can be given by equation (9). In practice, the termination criterion can be designed as $\|L^{k+1} - L^k\|_2 \leq »$. The designed procedure is termed as MultiK-MHKS and summarized in Table 1.

Using the obtained weight vector $!_l; l = 1:::m$, we can give the decision function of MultiK-MHKS for the input patten z with its corresponding mapped patterns $\{\Phi_l^e(z)\}_{l=1}^m$:

$$F(z) = \frac{1}{m}\sum_{l=1}^m !_l^T[\Phi_l^{eT}(z); 1]^T \begin{cases} > 0 & ; \quad then \quad z \in class + 1 \\ < 0 & ; \quad then \quad z \in class - 1 \end{cases} . \qquad (12)$$

Further it can be found that in Algorithm MultiK-MHKS, the update of $!_l^{k+1}$ is determined by $!_j^k; j = 1:::m; j \neq l$ as shown in (9), which reflects that these views *cooperate* each other. Moreover, if $m = 1; ¸ = 0$ of (8), MultiK-MHKS is degenerated to MHKS and so MHKS is the special instance of MultiK-MHKS. Finally, it should be stated that compared with IKM, EKM loses the sparsity [22] and thus our method also inherits the non-sparsity induced from EKM. We plan to address it completely in future work.

III. Experiments

In our experiments, the candidate kernels are linear kernel $ker(x_i; x_j) = x_i^T x_j$, RBF kernel $ker(x_i; x_j) = exp(-\frac{\|x_i - x_j\|_2^2}{2\sigma^2})$ where ¾ is set to the average value of all the l_2-norm distances $\|x_i - x_j\|_2; i; j = 1:::N$ as used in [29], and polynomial kernel $ker(x_i; x_j) = (x_i^T x_j + 1)^d$ where d is set to 2, respectively. Thus, the number of the generated views m is set to 3 in all the experiments. For MultiK-MHKS, the margin vector $\mathbf{b}_l$ is initialized to 10^{-6}; the » in the termination criterions is fixed to 10^{-3}; the parameter ½ is set to 0.99; and the $!_l; l = 2:::m$ are initialized to one unit vector respectively. For MHKS, the $\mathbf{b}; »; ½$ are initialized by the same setting as MultiK-MHKS. Both the regularization parameter c and ¸ in MHKS and MultiK-MHKS, are selected from the set $\{2^{-4}; 2^{-3}; :::; 2^3; 2^4\}$. Benchmark datasets used here are Soybean (35 Attributes/4 Classes/47 Samples), Balance (4/3/625), Water (38/2/116), Sonar (60/2/208), Wdbc (30/2/569), Diabetes (8/2/768), Iris (4/3/150), Wine (12/3/178), Letters (16/26/20000), and Segmentation (19/7/2310, denoted as "Segment." for short on Table 2 and 3) respectively, which are obtained from $http : ==www:ics:uci:edu= mlearn=MLRepository:html$. Meanwhile, ORL face database (28×23/40/400) available at $http : ==www:cam - orl:co:uk$ is also used. The one-against-one classification strategy [13] is adopted for multi-class problems.

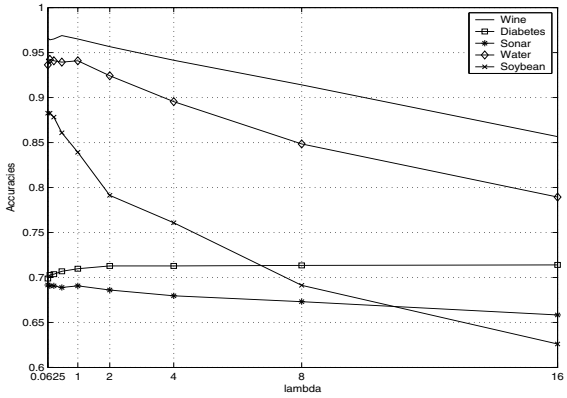

Fig. 1. Cross-validation (10-fold) accuracies on Wine, Diabetes, Sonar, Water, and Soybean as a function of parameter λ for MultiK-MHKS

A. Influence of Parameter ¸ on MultiK-MHKS's Performance

Compared with MHKS on the single kernel, MultiK-MHKS introduces one additional parameter ¸ as shown in (8). In order to show the influence of ¸ on classification, we give the cross-validation (10-fold) accuracies on the validation set as a function of ¸ on Wine, Diabetes, Sonar, Water, and Soybean as shown in Figure 1. From the figure, it can be found that the choice of ¸ plays an important role in terms of the accuracy. The similar phenomenon can also be obtained on the other datasets used. Consequently, in the following experiments, we give the classification performance based on the optimal ¸ by the Cross Validation.

B. Classification Performance Comparison

In order to demonstrate the effectiveness of the proposed algorithm, MHKS algorithm based on the single kernel

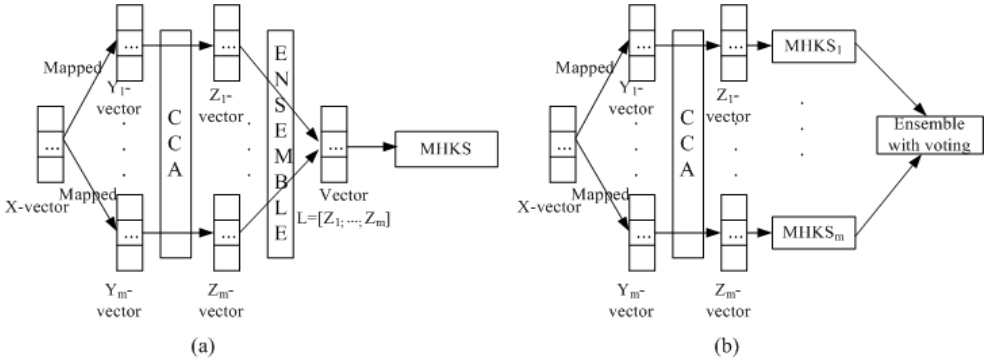

Fig. 2. (a) CCA+MHKS$_1$ combined in the feature level; (b) CCA+MHKS$_2$ combined in the decision level.

and two kinds of combination (denoted as CCA+MHKS$_1$ and CCA+MHKS$_2$ respectively) are carried out, where all the candidate kernels (linear, RBF, polynomial) are used. CCA+MHKS$_1$ shown in Fig.2.(a) means that the m transformed feature vectors will be concatenated into one ensemble vector, and then MHKS will be implemented with such an ensemble vector. CCA+MHKS$_2$ shown in Fig.2.(b) means that m MHKS algorithms are separately carried out in the m feature spaces respectively, and then combined by the majority voting technique. In addition, the other two MKL algorithms denoted as MKL [27] and SVM-2K [9] respectively are both compared with our method. Since SVM-2K only can deal with two kernels in one learning process, we give the best accuracy corresponding to the optimal combination from the 3 candidate kernels for all the datasets. Finally, we also give the best performance of the state-of-the-art SVM on the single kernel (SVM$_1$) and the long feature-vector obtained from concatenating all the single feature-space representations (SVM$_2$) similar to CCA+MHKS$_1$. Each of all the used datasets is divided into the two no-overlapping parts with the one for training and the other one for testing. Then, for each such classification problem, 10 independent runs are performed and their classification accuracies on the test sets are averaged and reported in Table 2, where for the different algorithms, the best, second best, third best results are boldface, italic and underlined respectively. From Table 2, it can be found that there is a clear improvement in the classification performance of MultiK-MHKS over MHKS based on the single kernel and the two kinds of CCA+MHKS. Compared with MKL[27] and SVM-2K[9], MultiK-MHKS also has a comparable or superior performance here. On the whole, on all the used datasets only except Diabetes, MultiK-MHKS gets the first or second place. Even on the Diabetes, MultiK-MHKS also has the comparable performance to the best method SVM$_1$ (only less by 0.06%).

For further finding out whether the proposed MultiK-MHKS is significantly better than the other compared algorithms in Table 2, we implement t-test [19] on the classification results of the 10 runs to calculate the statistical significance of MultiK-MHKS. The null hypothesis H_0 demonstrates that there is no significant difference between the mean number of samples correctly classified by MultiK-MHKS and the other compared algorithms here. The t-test values are also listed in Table 2, from which we can clearly find that the hypothesis H_0 is rejected at the 5% significance level, i.e. the t-test value ≥ 1.7341 on MHKS, CCA+MHKS$_1$ and CCA+MHKS$_2$ in almost all the used datasets. That means that the proposed fusion of CCA with MHKS in one single learning process possesses significantly superior classification performance to the combinations (CCA+MHKS$_1$ and CCA+MHKS$_2$) and MHKS, which exactly validates the effectiveness of our method. Meanwhile, compared with the other MKL algorithms (MKL[27], SVM-2K[9]) and SVM, MultiK-MHKS also shows a comparable or superior performance in statistics.

C. Running Time Comparison

In this section, the training time of MultiK-MHKS and the compared algorithms (CCA+MHKS$_1$, CCA+MHKS$_2$, MHKS, MKL[27], SVM-2K[9], SVM$_1$ and SVM$_2$) with their optimal parameters in 10 runs is reported in Table 3. All the computations are performed on Pentium IV 2.80 GHz processor running Windows 2000 Terminal and MATLAB environment. From Table 3, although MultiK-MHKS has longer running time than MHKS on most of the datasets due to multiple kernels used, our method has shorter running time with respect to the separate processes: CCA+MHKS$_1$ and CCA+MHKS$_2$ on most cases, especially on Letters. Further, compared with the other algorithms except MKL[27], it can also be noted that MultiK-MHKS has the competitive efficiency in computation.

D. Convergence Analysis

In this section, we give a discussion on the convergence of the proposed MultiK-MHKS. The previous section has stated that the optimization problem (8) with respect to a single β_l is convex. But the designed algorithm shown in Table 1 employs a modification of the gradient descent for each $\beta_l, l = 1,\ldots,m$. Thus it supposes that the sequential β_l differs slightly near the optimum solution [16]. Here, due to some theoretical difficulties in proving convergence, we adopt an empirical justification as used in [32] to demonstrate that MultiK-MHKS can converge in the limited iterations. Figure 3 shows the natural logarithm value of the objective function (8) changes with the iteration number of MultiK-MHKS respectively on the binary-class datasets: Diabetes, Sonar, the 2nd vs. 3rd

Table 2 Classification performance (%) and the t-test comparison between MultiK-MHKS, CCA+MHKS, MHKS, MKL[27], SVM-2K[9], SVM_1 on the single kernel and SVM_2 on the long feature-vector obtained from concatenating all single feature-space representations. (Note: In "$A(B)$", "A" denotes the classification accuracy and "B" assesses whether the classification performance of the corresponding algorithm is statistically different from that of MultiK-MHKS. "B^*" represents that the difference between the two algorithms is *not* significant at 5% significance level, i.e. $B^* < 1.7341$. For different algorithms, the best, second best, third best results are boldface, italic and underlined respectively.)

Datasets	MultiK-MHKS	CCA+MHKS$_1$	CCA+MHKS$_2$	MHKS	MKL[27]	SVM-2K[9]	SVM$_1$	SVM$_2$
Soybean	**99.57**	90.00(5.3906)	88.26(4.5810)	96.52(2.2778)	97.83(2.0580)	97.83(1.3950*)	*98.70*(1.0954*)	**99.57**(0*)
Balance	**93.94**	87.88(10.6106)	87.88(10.6106)	90.16(6.8245)	90.19(6.5939)	88.85(9.8502)	*91.09*(3.7080)	88.65(7.2257)
Water	**96.21**	57.27(19.2638)	63.79(24.3671)	94.85(1.4443*)	93.03(2.6877)	60.61(51.8327)	95.45(0.7111*)	*95.61*(0.5009*)
Sonar	**80.65**	72.78(6.8811)	72.78(6.8811)	75.83(4.4371)	75.28(2.3520)	71.67(6.9503)	79.63(0.8278*)	*80.00*(0.5399*)
Wdbc	**63.89**	*63.48*(1.0855*)	62.96(2.5204)	62.48(2.7287)	50.81(9.8425)	62.96(2.5210)	62.96(2.5210)	52.19(12.0261)
Diabetes	71.40	70.83(3.0534)	70.20(2.9210)	69.54(2.5185)	*71.43*(1.0000*)	*71.43*(1.0000*)	**71.46**(0.7878*)	54.17(10.6573)
Iris	*97.33*	92.67(6.2163)	90.53(8.9938)	*97.33*(0*)	*97.33*(0*)	97.20(0.1706*)	97.20(0.1765*)	**97.47**(0.1829*)
Wine	**96.89**	89.81(4.8455)	93.02(3.8432)	*95.19*(2.6056)	79.15(3.8048)	70.38(7.6811)	93.40(3.9220)	93.02(3.8419)
Letters	**84.46**	*83.81*(6.6020)	81.69(27.3007)	82.97(17.5195)	63.35(70.9851)	77.59(14.9665)	68.35(55.8505)	46.80(100.7056)
Segment.	**88.05**	*86.73*(0.5765*)	86.68(0.616*)	84.49(1.7697)	66.54(10.9667)	77.29(5.9461)	73.61(6.9737)	71.02(7.6047)
ORL	*94.80*	37.50(72.5762)	87.65(5.4816)	91.25(3.6481)	94.80(0*)	93.55(1.3992*)	**94.85**(0.0633*)	94.75(0.0635*)

Table 3 Training time (in s) comparison between MultiK-MHKS, CCA+MHKS, MHKS, MKL[27], SVM-2K[9], SVM_1 on the single kernel and SVM_2 on the long feature-vector obtained from concatenating all single feature-space representations.

Datasets	MultiK-MHKS	CCA+MHKS$_1$	CCA+MHKS$_2$	MHKS	MKL[27]	SVM-2K[9]	SVM$_1$	SVM$_2$
Soybean	0.09	0.27	0.19	0.02	483.50	0.09	2.76	1.19
Balance	97.50	26.17	26.69	150.15	1072.69	15.09	7.99	5.11
Water	0.41	4.67	2.91	0.07	98.43	0.08	0.79	0.29
Sonar	2.39	8.69	8.07	0.14	136.46	0.76	0.94	0.34
Wdbc	1.21	60.56	68.45	0.36	439.45	2.01	1.81	3.97
Diabetes	26.16	149.14	156.71	0.01	560.29	1.15	7.28	9.11
Iris	0.73	0.28	0.26	0.13	198.01	2.19	1.84	0.73
Wine	0.75	1.21	0.46	0.09	185.67	1.58	0.81	2.23
Letters	15902	23739	23704	24038	23399	40387	493	12038
Segment.	19.76	26.66	24.13	9.97	970.03	103.77	2.23	15.18
ORL	5.74	4.49	8.10	1.85	3894.20	31.10	11.18	13.70

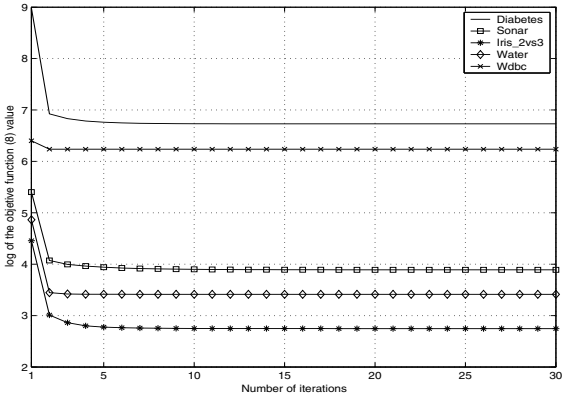

Fig. 3. Convergence of MultiK-MHKS on the natural logarithm value of the objective function (8)

classes of Iris, Water, and Wdbc. From the figure, it can be found that the optimization target (8) on these datasets can obviously converge to stable values, where less than ten iterations are usually enough to achieve convergence.

IV. Conclusion and Future Work

In this paper, we propose a novel MKL algorithm. Different from the existing MKL algorithms which mainly work by the convex combination of multiple kernels, the new algorithm adopts the spirit of CCA to fuse multiple views generated by multiple kernels into one single learning process such that one given algorithm in all the views can agree as much as possible on the outputs. In practice, by borrowing the motivating argument from NmCCA that is a new formulation for CCA in multi-view case, we introduce an additional term called Inter-Function Similarity Loss R_{IFSL} into the regularization learning. Then, with the fusion of R_{IFSL} and the original MHKS algorithm, a new MKL algorithm named MultiK-MHKS is developed. The experiments here illustrate that MultiK-MHKS is feasible and effective.

Motivated by the fact that the kernelization of CCA as a preprocessing step can improve the subsequent classification performance, Farquhar et al. [9] proposed the SVM-2K algorithm that combines the two stage learning (KCCA followed by SVM) into one single optimization. The main differences between our method and SVM-2K lie in: 1) that our method adopts EKM and SVM-2K employs IKM. It has been proved that EKM in the current presentation (6) has the same geometrical structure as IKM. But our method is more

general because EKM defined as (6) can naturally be replaced with a general proximity relation mapping [24], which need neither be Mercer kernels nor be limited to only one feature space; 2) given the data with m views, SVM-2K can only manipulate two views, but our method can deal with more than two views. Thus SVM-2K can be regarded as a special case of our method if SVM is selected as the base paradigm instead of MHKS in (2); 3) MultiK-MHKS has a comparable or superior classification performance to SVM-2K as validated in the experiments. However, due to EKM used here, our method also inherits the non-sparsity induced from EKM, which results in a bad scale with the size of datasets.

In future, we plan to: 1) make a separate study of the byproduct NmCCA derived here; 2) choose a subset of the full training data to define an approximate EKM in (6) for reducing the computation analogous to the work [18]; 3) generalize R_{IFSL} into SVM in the proposed framework (2) for a deeper study.

ACKNOWLEDGMENT

The authors thank the editor and the anonymous referees for their valuable comments, thank J.D.R. Farquhar and S. Szedmak for kindly providing the codes of SVM-2K, and Natural Science Foundations of China under Grant No.60773061 for supports.

REFERENCES

[1] F. Bach, G.R.G. Lanckriet, and M.I. Jordan. Multiple kernel learning, conic duality, and the SMO algorithm. In *Proceedings of the 21st International Conference on Machine Learning*, 2004.

[2] K.P. Bennett, M. Momma, and M.J. Embrechts. MARK: A boosting algorithm for heterogeneous kernel models. In *SIGKDD*, pages 24–31, 2002.

[3] J. Bi, T. Zhang, and K. Bennett. Column-generation boosting methods for mixture of kernels. In *KDD*, pages 521–526, 2004.

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

[5] Z. Chen and S. Haykin. On different facets of regularization theory. *Neural Computation*, 14(12):1481–1497, 2002.

[6] N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines and other kernel-based learning methods*. Cambridge University Press, 2000.

[7] I.M. de Diego, J.M. Moguerza, and A. Muñoz. Combining kernel information for support vector classification. In *MCS, LNCS*, pages 102–111, 2004.

[8] I.M. de Diego, J.M. Moguerza, and A. Muñoz. On the fusion of polynomial kernels for support vector classifiers. In *IDEAL, LNCS*, pages 330–337, 2006.

[9] J.D.R. Farquhar, D.R. Hardoon, H. Meng, J. Shawe-Taylor, and S. Szedmak. Two view learning: SVM-2K, theory and practice. In *Neural Information Processing Systems*, 2005.

[10] Y. Grandvalet and S. Canu. Adaptive scaling for feature selection in SVMs. In *Neural Information Processing Systems*, 2002.

[11] D. Hardoon, S. Szedmak, and J. Shawe-Taylor. Canonical correlation analysis: An overview with application to learning methods. *Neural Computation*, 16:2639–2664, 2004.

[12] H. Hotelling. Relations between two sets of variates. *Biometrika*, 28:321–377, 1936.

[13] U. Kreβel. Pairwise classification and support vector machines. *In: B. Schölkopf, C. Burges, A. Somla (eds.) Advances in kernel methods: support vector machine. MIT Press Cambridge, MA*, pages 255–268, 1998.

[14] G.R.G. Lanckriet, T.D. Bie, N. Cristianini, M.I. Jordan, and W.S. Noble. A statistical framework for genomic data fusion. *Bioinformatics*, 20(16):2626–2635, 2004.

[15] G.R.G. Lanckriet, N. Cristianini, P. Bartlett, L.E. Ghaoui, and M.I. Jordan. Learning the kernel matrix with semidefinite programming. *Journal of Machine Learning Research*, 5:27–72, 2004.

[16] J. Łeski. Ho-kashyap classifier with generalization control. *Pattern Recognition Letters*, 24(14):2281–2290, 2003.

[17] Y. Li and J. Shawe-Taylor. Using KCCA for Japanese-English cross-language information retrieval and classification. *Journal of Intelligent Information Systems*, 2005.

[18] J. Ma. Function replacement vs. kernel trick. *Neurocomputing*, 50:479–483, 2003.

[19] T.M. Mitchell. *Machine Learning*. Boston: McGraw-Hill, 1997.

[20] J.M. Moguerza, A. Muñoz, and I.M. de Diego. Fusion of gaussian kernels within support vector classification. In *CIARP, LNCS*, pages 945–953, 2006.

[21] M. Momma and K. Bennett. A pattern search method for model selection of support vector regression. In *Proceedings of the Second SIAM International Conference on Data Mining. SIAM*, pages 261–274, 2002.

[22] K.-R. Müller, S. Mika, G. Rätsch, K. Tsuda, and B. Schölkopf. An introduction to kernel-based learning algorithms. *IEEE Transactions On Neural Networks*, 12(2):181–202, 2001.

[23] C.S. Ong, A.J. Smola, and R.C. Williamson. Learning the kernel with hyperkernels. *Journal of Machine Learning Research*, 6:1043–1071, 2005.

[24] E. Pekalska, P. Paclik, and R.P.W. Duin. A generalized kernel approach to dissimilarity-based classification. *Journal of Machine Learning Research*, 2:175–211, 2001.

[25] B. Schölkopf, S. Mika, C.J.C. Burges, P. Knirsch, K.-R. Müller, G. Rätsch, and A.J. Smola. Input space versus feature space in kernel-based methods. *IEEE Transactions On Neural Networks*, 10(5):1000–1017, 1999.

[26] J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

[27] S. Sonnenburg, G. Rätsch, and C. Schäfer. A general and efficient multiple kernel learning algorithm. In *Neural Information Processing Systems*, 2005.

[28] S. Sonnenburg, G. Rätsch, C. Schäfer, and B. Schölkopf. Large scale multiple kernel learning. *Journal of Machine Learning Research*, 2006.

[29] I. Tsang, A. Kocsor, and J. Kwok. Efficient kernel feature extraction for massive data sets. In *International Conference on Knowledge Discovery and Data Mining*, 2006.

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

[31] H. Xiong, M.N.S. Swamy, and M.O. Ahmad. Optimizing the kernel in the empirical feature space. *IEEE Transactions On Neural Networks*, 16(2):460–474, 2005.

[32] J. Ye. Generalized low rank approximation of matrices. *Machine Learning*, 61(1-3):167–191, 2005.