

# Feature Selection and Kernel Learning for Local Learning-Based Clustering

Hong Zeng, *Member, IEEE*, and Yiu-ming Cheung, *Senior Member, IEEE*

**Abstract**—The performance of the most clustering algorithms highly relies on the representation of data in the input space or the Hilbert space of kernel methods. This paper is to obtain an appropriate data representation through feature selection or kernel learning within the framework of the Local Learning-Based Clustering (LLC) (Wu and Schölkopf 2006) method, which can outperform the global learning-based ones when dealing with the high-dimensional data lying on manifold. Specifically, we associate a weight to each feature or kernel and incorporate it into the built-in regularization of the LLC algorithm to take into account the relevance of each feature or kernel for the clustering. Accordingly, the weights are estimated iteratively in the clustering process. We show that the resulting weighted regularization with an additional constraint on the weights is equivalent to a known sparse-promoting penalty. Hence, the weights of those irrelevant features or kernels can be shrunk toward zero. Extensive experiments show the efficacy of the proposed methods on the benchmark data sets.

**Index Terms**—High-dimensional data, local learning-based clustering, feature selection, kernel learning, sparse weighting.



## 1 INTRODUCTION

IT is common to perform high-dimensional data clustering in a variety of pattern recognition and data mining problems in which the high-dimensional data are represented by a large number of features. However, the discrimination among patterns is often impeded by the abundance of features. For instance, it is quite common to have thousands of gene expression coefficients as features for a single sample in genomic data analysis, but only a small fraction is capable of discriminating among different tissue classes. Those irrelevant features involved in the prediction may seriously degrade the performance of an inference machine [13]. Therefore, it is desirable to develop an effective feature selection algorithm toward identifying those features relevant to the inference task in hand.

On the other hand, the kernel methods have been widely applied to a variety of learning problems in the past decades, where the data are implicitly mapped into a nonlinear high-dimensional space by kernel function [30]. It is known that the performance of these methods will heavily hinge on the choice of kernel. Unfortunately, the most suitable kernel for a particular task is often unknown in advance. Moreover, exhaustive search on a user-defined pool of kernels will be quite time-consuming when the size of the pool becomes large [29]. Hence, it is crucial to learn an appropriate kernel efficiently to make the performance

of the employed kernel-based inference method robust or even improved.

This paper attempts to obtain an appropriate data representation for clustering in the input space or the Hilbert space (also interchangeably called feature space hereinafter) of kernel methods. Accordingly, two issues, i.e., feature selection and kernel learning, are considered. In fact, either of these two issues have been extensively studied in the context of supervised learning, but are comparatively less explored in the clustering problem. A major reason is that feature selection or kernel learning in unsupervised learning becomes more challenging without the presence of ground-truth class labels that could guide the search for relevant representations. Most recently, some research works regarding these two issues have been done in the unsupervised case, e.g., see [43], [12], [13], [25], [34]. A predominant strategy among these approaches, which have achieved prominent improved clustering performance, is to first relax the binary *hard* decision on the relevance of feature or kernel to a real-valued *soft* one, i.e., a confidence or weight, turning the combinatorial search problem into a continuous learning problem. Then, these approaches apply the following two iterative steps until convergence: 1) estimating the weights for features or kernels using the intermediate clustering result, and 2) refeeding the weighted feature or kernel into the employed clustering algorithm.

Despite the success of such common strategy for both the feature selection and kernel learning in clustering, there are still two problems at least not properly addressed. One problem is on the *exploited clustering algorithm* which generates the intermediate clustering result. The feature or kernel is evaluated by the intermediate clustering result; an improper intermediate partition may lead to a poor weighting. Some employed clustering algorithms in those methods may be prone to such failure, especially when dealing with high-dimensional data lying on manifold. The other problem is the *sparseness* of the weights. Sparse weighting, i.e., a big

• H. Zeng is with the School of Instrument Science and Engineering, Southeast University, China, and the Department of Computer Science, Hong Kong Baptist University, Hong Kong SAR, China.  
E-mail: littlezenghong@gmail.com.

• Y.M. Cheung is with the Department of Computer Science, Hong Kong Baptist University, Hong Kong SAR, China.  
E-mail: ymc@comp.hkbu.edu.hk.

Manuscript received 17 Feb. 2009; revised 12 Dec. 2009; accepted 23 Oct. 2010; published online 29 Nov. 2010.

Recommended for acceptance by M. Meila.

For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number TPAMI-2009-02-0117.

Digital Object Identifier no. 10.1109/TPAMI.2010.215.

gap between the weights for informative and uninformative representation, as well as vanishing weights for uninformative ones, is desirable so that the effect of irrelevant features or kernels can be significantly mitigated. Moreover, it helps to better understand the problem by focusing on only a few dominant features or kernels that most contribute to the task. To the best of our knowledge, few of those methods have provided a principled and effective regularization on the sparsity of weights.

In this paper, we shall propose two methods that perform the feature selection and kernel learning within the framework of the Local Learning-Based Clustering (LLC) [3], respectively. The LLC algorithm tries to ensure that the cluster label of each data point is close to the one predicted by the local regression model, a current supervised learning method, with its neighboring points and their cluster labels [3]. Essentially, it finds the partition which is mostly able to embody such local configuration, it is thereby expected to be good at clustering data sets lying on manifold, e.g., the high-dimensional sparse data sets. Furthermore, by utilizing the ridge regression in the supervised learning to develop an unsupervised clustering method, LLC has a built-in regularization for the model complexity. In this paper, we modify such a built-in ridge regularization in the local regression model to take into account the relevance of each feature or kernel for clustering. It is shown that the modified penalty term with a constraint is equivalent to the existing sparse-promoting penalty. Hence, it is guaranteed that the resulting weights for features are sparse and then local configuration may get refined; a better clustering result can thus be expected. Moreover, the proposed feature selection method is extended from the observation space to the feature space, naturally leading to the problem of learning a convex combination of kernels for the local learning-based clustering. The main contributions of our work are two-fold:

1. A novel feature selection method and a kernel learning method are proposed for local learning-based clustering, respectively, whereas almost all of the existing counterparts are developed for global learning-based clustering.
2. The feature selection and kernel learning for clustering are addressed in a unified approach under the same regularization framework.

The remainder of this paper is organized as follows: Related works are reviewed in Section 2. Section 3 gives an overview of the LLC algorithm. We present the proposed feature selection method in Section 4, and then extend it to learn the combination of kernels in Section 5. Some discussions are given in Section 6. In Section 7, extensive experiments are conducted to show the performance of the proposed methods on several benchmark data sets. Finally, we draw a conclusion in Section 8.

## 2 RELATED WORKS

This section overviews the literature on the unsupervised feature selection and kernel learning only. The reviews of supervised feature selection and kernel learning can be found in [5] and [27], respectively.

The approaches to unsupervised feature selection for clustering can be generally categorized as the *filter* and

*wrapper* ones. The *filter* approaches [9], [40], [7], [8], [6] leave out uninformative features before the clustering. They have demonstrated great computational efficiency because they do not involve clustering when evaluating the quality of features. In general, such a method has to determine the number of selected relevant features. Unfortunately, this crucial issue has rarely been addressed in the literature, thus causing difficulty in practical applications [13]. In contrast, the *wrapper* approaches [10], [11], [12], [13] first construct a candidate of feature subset on which its goodness is then assessed by investigating the performance of a specific clustering. These two steps are repeated until convergence. In general, the *wrapper* approaches outperform the *filter* ones, but is more time-consuming because of the exhaustive search in the space of feature subsets. In the literature, some *wrapper* approaches, e.g., [10], [11], have utilized the greedy search (i.e., a nonexhaustive one), which, however, cannot guarantee to select all relevant features. This shortcoming, as well as the issue of determining the number of selected relevant features in the *filter* approaches, can be alleviated by assigning each feature a nonnegative weight [12], [13] rather than a binary indicator to indicate its relevance to the clustering. Further, the combinatorial explosion of the search space can be avoided as well by casting the feature selection as an estimation problem. Our approach also follows this strategy. Based on recent progress on spectral clustering, the algorithm in [13] tries to optimize the cluster coherence measured by the sum of squared eigenvalues of an affinity matrix, which is constructed by aggregating weak affinity matrices built with weighted feature vectors. The solutions to the clustering and feature weighting are obtained by an efficient iterative algorithm based on eigendecomposition. Nevertheless, the clustering algorithm in [13] is essentially the kernel k-means with a linear kernel, which is a global learning method; thus it is difficult to deal with the data that lie on nonlinear manifold. In [12], feature weights are estimated by modifying the M-step of the EM algorithm through the Bayesian inference mechanism when there are only two clusters. It is noteworthy that, in addition to incorporating feature selection, there are several approaches to learning parameterized similarity functions in the spectral clustering for improving the clustering performance [1], [28]. Despite the success in their application domain, it is often nontrivial to interpret the physical meaning of the parameters specified in these methods, e.g., the parameter associated with a feature having a negative weight in [1], [28]. Also, the parameters specified for the RBF kernel functions may increase the difficulty for the optimization.

For kernel learning in clustering, some heuristic approaches [24], [28] directly learn the kernel parameters of some specific kernels. Although some improvement can often be achieved, an extension of the learning method to other kernel functions is usually nontrivial [42]. In contrast, a more effective framework, termed the multiple kernel learning [26], learns a linear combination of base kernels with the different weights, which will be estimated simultaneously in the inference process, e.g., see [34], [41], [25]. Our proposed method, which will be described later, also belongs to this framework. In [34], the algorithm tries to find a maximum margin hyperplane to cluster data (restricted to the binary-class case), accompanied by

learning a mixture of Laplacian matrices. The method in [41] extends the kernel discriminant analysis technique to clustering and learns a combination of kernel matrices jointly. In [34], [41], no penalty is imposed on the kernel weights; thus the sparsity may not be guaranteed. In [25], clustering is phrased as a nonnegative matrix factorization problem of a fused kernel matrices, and the sparseness of kernel weights is controlled by a heuristic entropy penalty which, however, favors a uniform weighting.

An important application of the multiple kernel learning is to fuse the information from heterogeneous sources as follows [26]: Associate each source with a kernel function, and then combine the set of prototype kernels generated from these sources to perform the inference. In this respect, the multiview clustering is also a related work whose goal is to learn a *consensus* result from multiple representations [39], [46]. However, it implicitly treats all the sources equally, regardless of the clustering performance with each source. In contrast, our proposed method is able to determine the weight for each source automatically according to its capability of discrimination; thus it will be more robust from the practical viewpoint.

### 3 OVERVIEW OF THE LOCAL LEARNING-BASED CLUSTERING ALGORITHM

Given  $n$  data points  $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^n$  ( $\mathbf{x}_i \in \mathbb{R}^d$ ), the data set will be partitioned into  $C$  clusters. The clustering result can be represented by a *cluster assignment indicator matrix*  $\mathbf{P} = [p_{ic}] \in \{0, 1\}^{n \times C}$  such that  $p_{ic} = 1$  if  $\mathbf{x}_i$  belongs to the  $c$ th cluster, and  $p_{ic} = 0$  otherwise. The *scaled cluster assignment indicator matrix* used in this paper is defined as

$$\mathbf{Y} = \mathbf{P}(\mathbf{P}^T \mathbf{P})^{-\frac{1}{2}} = [\mathbf{y}^1, \mathbf{y}^2, \dots, \mathbf{y}^C],$$

where  $\mathbf{y}^c = [y_{1c}, \dots, y_{nc}]^T \in \mathbb{R}^n$  ( $1 \leq c \leq C$ ) is the  $c$ th column of  $\mathbf{Y} \in \mathbb{R}^{n \times C}$ .  $y_{ic} = p_{ic}/\sqrt{n_c}$  can be regarded as the confidence that  $\mathbf{x}_i$  is assigned to the  $c$ th cluster, where  $n_c$  is the size of the  $c$ th cluster. It is easy to verify that

$$\mathbf{Y}^T \mathbf{Y} = \mathbf{I}, \quad (1)$$

where  $\mathbf{I} \in \mathbb{R}^{n \times n}$  is an identity matrix.

The starting point of the LLC [3] is that the cluster assignments in the neighborhood of each point should be as smooth as possible. Specifically, it assumes that the cluster indicator value at each point should be well estimated by a regression model trained locally with its neighbors and their cluster indicator values. Suppose there exists an arbitrary  $\mathbf{Y}$  at first; for each  $\mathbf{x}_i$ , the model is built with the training data  $\{(\mathbf{x}_j, y_{jc})\}_{\mathbf{x}_j \in \mathcal{N}_i}$  ( $1 \leq c \leq C, 1 \leq i, j \leq n$ ), where  $\mathcal{N}_i$  denotes the set of neighboring<sup>1</sup> points of  $\mathbf{x}_i$ , but  $\mathbf{x}_i$  is excluded. The output of the local model is of the following form:

$$f_i^c(\mathbf{x}) = \mathbf{x}^T \boldsymbol{\theta}_i^c, \quad \forall \mathbf{x} \in \mathbb{R}^d, \quad (2)$$

where  $\boldsymbol{\theta}_i^c \in \mathbb{R}^d$  is the local regression coefficient vector,  $f_i^c(\cdot)$  denotes the local model learned with the training data  $\{(\mathbf{x}_j, y_{jc})\}_{\mathbf{x}_j \in \mathcal{N}_i}$ . Here, the bias term is ignored for simplicity

1. The  $k$ -mutual neighbors are adopted in order to well describe the local structure, i.e.,  $\mathbf{x}_j$  is considered as a neighbor of  $\mathbf{x}_i$  only if  $\mathbf{x}_i$  is also one of the  $k$ -nearest neighbors of  $\mathbf{x}_j$ .

provided that one of the features is always 1. In [3], the model is obtained by solving the following  $l_2$  norm regularized least square problem:

$$\min_{\{\boldsymbol{\theta}_i^c\}} \sum_{c=1}^C \sum_{i=1}^n \left[ \sum_{\mathbf{x}_j \in \mathcal{N}_i} \beta (y_{jc} - \mathbf{x}_j^T \boldsymbol{\theta}_i^c)^2 + \|\boldsymbol{\theta}_i^c\|^2 \right], \quad (3)$$

where  $\beta$  is a trade-off parameter. Let  $\{\hat{\boldsymbol{\theta}}_i^c\}$  be the solution to the linear ridge regression problem (3), the predicted cluster assignment for the test data  $\mathbf{x}_i$  can then be calculated by

$$\hat{y}_{ic} = f_i^c(\mathbf{x}_i) = \mathbf{x}_i^T \hat{\boldsymbol{\theta}}_i^c = \boldsymbol{\alpha}_i^T \mathbf{y}_i^c, \quad (4)$$

where

$$\boldsymbol{\alpha}_i^T = \beta \mathbf{x}_i^T (\beta \mathbf{X}_i \mathbf{X}_i^T + \mathbf{I})^{-1} \mathbf{X}_i, \quad (5)$$

$\mathbf{X}_i = [\mathbf{x}_{i_1}, \mathbf{x}_{i_2}, \dots, \mathbf{x}_{i_{n_i}}]$  with  $\mathbf{x}_{i_k}$  being the  $k$ th neighbor of  $\mathbf{x}_i$ ,  $\mathbf{y}_i^c = [y_{i_1c}, y_{i_2c}, \dots, y_{i_{n_i}c}]^T$ , and  $n_i$  is the size of  $\mathcal{N}_i$ .

After all of the local predictors have been constructed, the LLC combines them together so that an optimal cluster indicator matrix  $\mathbf{Y}$  is found via minimizing the following overall prediction error:

$$\begin{aligned} & \sum_{c=1}^C \sum_{i=1}^n (y_{ic} - \hat{y}_{ic})^2 \\ &= \sum_{c=1}^C \|\mathbf{y}^c - \mathbf{A} \mathbf{y}^c\|^2 \\ &= \text{trace}[\mathbf{Y}^T (\mathbf{I} - \mathbf{A})^T (\mathbf{I} - \mathbf{A}) \mathbf{Y}] \\ &= \text{trace}(\mathbf{Y}^T \mathbf{M} \mathbf{Y}), \end{aligned} \quad (6)$$

where  $\mathbf{M} = (\mathbf{I} - \mathbf{A})^T (\mathbf{I} - \mathbf{A})$ ,  $\mathbf{A}$  is an  $n \times n$  sparse matrix, whose  $(i, j)$ th entry  $a_{ij}$  is the corresponding element in  $\boldsymbol{\alpha}_i$  by (5) if  $\mathbf{x}_j \in \mathcal{N}_i$  and 0 otherwise.

As in the spectral clustering [14], [15],  $\mathbf{Y}$  is relaxed into the continuous domain while keeping the property of (1) for the problem (6). The LLC then solves the following tractable continuous optimization problem:

$$\begin{aligned} & \min_{\mathbf{Y} \in \mathbb{R}^{n \times C}} \text{trace}(\mathbf{Y}^T \mathbf{M} \mathbf{Y}) \\ & \text{s.t. } \mathbf{Y}^T \mathbf{Y} = \mathbf{I}. \end{aligned} \quad (7)$$

A solution to  $\mathbf{Y}$  is given by the first  $C$  eigenvectors of the matrix  $\mathbf{M}$  corresponding to the first  $C$  smallest eigenvalues. Similarly to [14], [15], the final partition result is obtained by discretizing  $\mathbf{Y}$  via the method in [15] or by the  $k$ -means as in [14]. Promising results have been reported in [3].

### 4 FEATURE SELECTION FOR LOCAL LEARNING-BASED CLUSTERING

In this section, we will integrate the feature selection into the LLC. It should be noted that the key ingredient of the LLC is to learn the local regression model, which is trained only with the points in each neighborhood. However, there may be too few data points in its neighborhood to learn a good predictor. This can be even more difficult for a high-dimensional data set. Furthermore, it may lead to non-smooth predictions for points from overlapping zones of adjacent neighborhoods as the result of independently

training the local regression model in each neighborhood. Last but not least, the  $l_2$  norm penalty in ridge regression is known to be less robust to the irrelevant features. In order to overcome these limitations, a more effective training method which can reduce the complexity of the local regression model in each neighborhood and enforce smoothness among the local regressors is required. Inspired by recent works on multitask learning [16], [47] which extract a shared representation for a group of related training tasks, demonstrating an improved performance compared to learning each task independently, we propose to select a small subset of features that is good for all the local models.

To this end, we introduce a binary feature selection vector  $\tau = [\tau_1, \tau_2, \dots, \tau_d]$ ,  $\tau_l \in \{0, 1\}$  to the local discriminant function as follows:

$$f_i^c(\mathbf{x}) = \mathbf{x}^T \text{diag}(\sqrt{\tau})\theta_i^c + b_i^c = \sum_{l=1}^d x_l \sqrt{\tau_l} (\theta_i^c)_l + b_i^c, \quad (8)$$

where  $\text{diag}(\sqrt{\tau}) \in \mathbb{R}^{d \times d}$  is a diagonal matrix with  $\sqrt{\tau} \in \mathbb{R}^d$  on the diagonal,  $(\theta_i^c)_l$  is the  $l$ th element of  $\theta_i^c \in \mathbb{R}^d$ , and  $b_i^c \in \mathbb{R}$  is the bias term. In (8), the entries of  $\theta_i^c$  can be turned on and off depending on the corresponding entries of the switch variable  $\tau$ . To avoid a combinatorial search for  $\tau$  later, we relax the constraint  $\tau_l \in \{0, 1\}$  to  $\tau_l \geq 0$  and further restrict its scale by  $\sum_{l=1}^d \tau_l = 1$ .<sup>2</sup> Consequently, the local discriminant function will be solved by

$$\min_{\{\theta_i^c, b_i^c\}, \sum_{l=1}^d \tau_l = 1, \tau_l \geq 0} \sum_{c=1}^C \sum_{i=1}^n \left[ \sum_{\mathbf{x}_j \in \mathcal{N}_i} \beta (y_{jc} - \mathbf{x}_j^T \text{diag}(\sqrt{\tau})\theta_i^c - b_i^c)^2 + \theta_i^{cT} \theta_i^c \right], \quad (9)$$

or equivalently, the following problem:

$$\min_{\{\mathbf{w}_i^c, b_i^c\}, \sum_{l=1}^d \tau_l = 1, \tau_l \geq 0} \sum_{c=1}^C \sum_{i=1}^n \left[ \sum_{\mathbf{x}_j \in \mathcal{N}_i} \beta (y_{jc} - \mathbf{x}_j^T \mathbf{w}_i^c - b_i^c)^2 + \mathbf{w}_i^{cT} \text{diag}(\tau^{-1}) \mathbf{w}_i^c \right], \quad (10)$$

which is obtained by applying a change of variables  $\text{diag}(\sqrt{\tau})\theta_i^c \rightarrow \mathbf{w}_i^c$ . The local model is now tantamount to being of the following form:

$$f_i^c(\mathbf{x}) = \mathbf{x}^T \mathbf{w}_i^c + b_i^c, \quad (11)$$

and the regression coefficient  $\mathbf{w}_i^c$  is now regularized with a weighted  $l_2$  norm:  $\mathbf{w}_i^{cT} \text{diag}(\tau^{-1}) \mathbf{w}_i^c = \sum_l \frac{(\mathbf{w}_i^c)_l^2}{\tau_l}$ , i.e., the second term in the square bracket of (10). Thus, a small value for  $\tau_l$ , which is expected to be associated with an irrelevant feature, will result in a large penalization on  $(\mathbf{w}_i^c)_l$  by this weighted norm. Furthermore, in the extreme case of

2. As will be seen later, such a simplex constraint is crucial for enforcing the sparsity of  $\tau$ . Moreover, we simply set  $\sum_{l=1}^d \tau_l = 1$  rather than  $\sum_{l=1}^d \tau_l = \rho$ , where  $\rho$  is a tunable constant, in order to reduce the number of free parameters.

$\tau_l = 0$ , we will prove later that it leads to  $(\mathbf{w}_i^c)_l = 0 \forall i, c$ .<sup>3</sup> That is, the  $l$ th feature will be completely eliminated from the prediction; thus an improved clustering result can be expected. Subsequently, to perform the feature selection together with the LLC, we develop an alternating update algorithm to estimate the clustering captured in  $\mathbf{Y}$  and the feature weight  $\tau$  as follows:

#### 4.1 Update $\mathbf{Y}$ As Given $\tau$

First, the nearest neighbors  $\mathcal{N}_i$  should be refound according to the  $\tau$ -weighted square euclidean distance, i.e.,

$$d_\tau(\mathbf{x}_1, \mathbf{x}_2) = \|\mathbf{x}_1 - \mathbf{x}_2\|_\tau^2 = \sum_{l=1}^d \tau_l (\mathbf{x}_1^{(l)} - \mathbf{x}_2^{(l)})^2. \quad (12)$$

With the fixed feature weight  $\tau$ , the analytic solution for problem (10) can then be easily obtained by setting the derivatives to zero. That is,

$$\mathbf{w}_i^c = \beta [\beta \mathbf{X}_i \mathbf{\Pi}_i \mathbf{X}_i^T + \text{diag}(\tau^{-1})]^{-1} \mathbf{X}_i \mathbf{\Pi}_i \mathbf{y}_i^c, \quad (13)$$

$$b_i^c = \frac{1}{n_i} \mathbf{e}_i^T (\mathbf{y}_i^c - \mathbf{X}_i^T \mathbf{w}_i^c), \quad (14)$$

where  $\mathbf{e}_i = [11 \dots 1]^T \in \mathbb{R}^{n_i}$ ,  $\mathbf{\Pi}_i = \mathbf{I}_i - \frac{1}{n_i} \mathbf{e}_i \mathbf{e}_i^T$  is a centering projection matrix, satisfying  $\mathbf{\Pi}_i \mathbf{\Pi}_i = \mathbf{\Pi}_i$ .  $\mathbf{I}_i$  is an  $n_i \times n_i$  unit matrix.

For high-dimensional data, the computation of the matrix inversion in (13) will be quite time-consuming because the time complexity is  $O(d^3)$ . Fortunately, by applying the Woodbury's matrix inversion lemma, we can get

$$\begin{aligned} \mathbf{w}_i^c &= \beta \text{diag}(\tau) \mathbf{X}_i \mathbf{\Pi}_i \\ &[\mathbf{I}_i - (\beta^{-1} \mathbf{I}_i + \mathbf{\Pi}_i \mathbf{X}_i^T \text{diag}(\tau) \mathbf{X}_i \mathbf{\Pi}_i)^{-1} \mathbf{\Pi}_i \mathbf{X}_i^T \text{diag}(\tau) \mathbf{X}_i \mathbf{\Pi}_i] \mathbf{y}_i^c, \end{aligned} \quad (15)$$

in which the time complexity of the matrix inversion in (15) is only  $O(n_i^3)$ . In general, we often have  $n_i \ll d$ ; thus the computational cost can be considerably reduced. Besides, from (15), it can be seen that  $(\mathbf{w}_i^c)_l (\forall i, c)$  goes to 0 as the feature weight  $\tau_l$  vanishes.

Subsequently, the predicted cluster assignment confidence for  $\mathbf{x}_i$  will be obtained as follows:

$$\hat{y}_{ic} = \mathbf{x}_i^T \mathbf{w}_i^c + b_i^c = \alpha_i^T \mathbf{y}_i^c, \quad (16)$$

with

$$\begin{aligned} \alpha_i^T &= \beta \left( \mathbf{k}_i^\tau - \frac{1}{n_i} \mathbf{e}_i^T \mathbf{K}_i^\tau \right) \\ \mathbf{\Pi}_i [\mathbf{I}_i - (\beta^{-1} \mathbf{I}_i + \mathbf{\Pi}_i \mathbf{K}_i^\tau \mathbf{\Pi}_i)^{-1} \mathbf{\Pi}_i \mathbf{K}_i^\tau \mathbf{\Pi}_i] &+ \frac{1}{n_i} \mathbf{e}_i^T, \end{aligned} \quad (17)$$

where  $\mathbf{k}_i^\tau = \mathbf{x}_i^T \text{diag}(\tau) \mathbf{X}_i$  and  $\mathbf{K}_i^\tau = \mathbf{X}_i^T \text{diag}(\tau) \mathbf{X}_i$ .

As in the LLC, we construct the key matrix  $\mathbf{M}$  by (17) and (6). To solve the same optimization problem in (7), the columns of  $\mathbf{Y}$  are simply set at the first  $C$  eigenvectors of  $\mathbf{M}$  corresponding to the smallest  $C$  eigenvalues.

3. In this paper, we will use the convention that  $\frac{z}{0} = 0$  if  $z = 0$  and otherwise.

## 4.2 Update $\tau$ As Given $\mathbf{Y}$

With the fixed  $\mathbf{Y}$  and neighborhood determined at each point, a reasonable  $\tau$  is the one that can lead to a better local regression model which is characterized by a lower objective value at the minimum of (10). We will apply this criterion to reestimate  $\tau$ . We remove the bias term by plugging (14) into (10), and we then have

$$\min_{\{\mathbf{w}_i^c\}} F(\{\mathbf{w}_i^c\}, \tau) = \sum_{c=1}^C \sum_{i=1}^n [\beta \|\mathbf{\Pi}_i \mathbf{y}_{ic} - (\mathbf{X}_i \mathbf{\Pi}_i)^T \mathbf{w}_i^c\|^2 + \mathbf{w}_i^{cT} \text{diag}(\boldsymbol{\tau}^{-1}) \mathbf{w}_i^c]. \quad (18)$$

Subsequently, the estimation of  $\tau$  is reformulated as follows:

$$\min_{\boldsymbol{\tau}} \mathcal{P}(\boldsymbol{\tau}), \quad \text{s.t.} \sum_{l=1}^d \tau_l = 1, \tau_l \geq 0, \forall l, \quad (19)$$

where  $\mathcal{P}(\boldsymbol{\tau}) = F(\{\mathbf{w}_i^{c*}\}, \boldsymbol{\tau})$  with  $\{\mathbf{w}_i^{c*}\} = \arg \min_{\{\mathbf{w}_i^c\}} F(\{\mathbf{w}_i^c\}, \boldsymbol{\tau})$  given in (15). Hence, the Lagrange of (19) is

$$\mathcal{L}(\boldsymbol{\tau}, \lambda, \boldsymbol{\varepsilon}) = \mathcal{P}(\boldsymbol{\tau}) + \lambda \left( \sum_{l=1}^d \tau_l - 1 \right) - \sum_{l=1}^d \varepsilon_l \tau_l, \quad (20)$$

where the scalar  $\lambda \geq 0$  and the vector  $\boldsymbol{\varepsilon} \geq \mathbf{0}$  are Lagrangian multipliers. The derivative of  $\mathcal{L}$  with respect to  $\tau_l$ , ( $l = 1, \dots, d$ ) is computed as

$$\frac{\partial \mathcal{L}}{\partial \tau_l} = \frac{\partial \mathcal{P}}{\partial \tau_l} + \lambda - \varepsilon_l, \quad (21)$$

where

$$\begin{aligned} \frac{\partial \mathcal{P}}{\partial \tau_l} &= \frac{\partial F(\{\mathbf{w}_i^c\}, \boldsymbol{\tau})}{\partial \tau_l} \Big|_{\mathbf{w}_i^c = \mathbf{w}_i^{c*}} + \sum_{i,c} \frac{\partial (\mathbf{w}_i^{c*})_l}{\partial \tau_l} \underbrace{\frac{\partial F(\{\mathbf{w}_i^c\}, \boldsymbol{\tau})}{\partial (\mathbf{w}_i^c)_l}}_0 \Big|_{\mathbf{w}_i^c = \mathbf{w}_i^{c*}} \\ &= - \frac{\sum_{c=1}^C \sum_{i=1}^n (\mathbf{w}_i^{c*})_l^2}{\tau_l^2}. \end{aligned} \quad (22)$$

Thus, at the optimality, we have

$$\tau_l^2 = \frac{\sum_{c=1}^C \sum_{i=1}^n (\mathbf{w}_i^{c*})_l^2}{\lambda - \varepsilon_l}, \quad \forall l, \quad (23)$$

$$\lambda \geq 0, \varepsilon_l \geq 0, \tau_l \geq 0, \quad \forall l, \quad (24)$$

$$\sum_{l=1}^d \tau_l = 1, \quad (25)$$

$$\varepsilon_l \tau_l = 0, \quad \forall l. \quad (26)$$

By using the Karush-Kuhn-Tucker (KKT) condition [31], i.e., (26), it is easy to verify the following two cases:

- Case 1:  $\sum_{c=1}^C \sum_{i=1}^n (\mathbf{w}_i^{c*})_l^2 = 0 \Rightarrow \tau_l = 0$ ;
- Case 2:  $\sum_{c=1}^C \sum_{i=1}^n (\mathbf{w}_i^{c*})_l^2 \neq 0 \Rightarrow \varepsilon_l = 0$  and

$$\tau_l = \sqrt{\frac{\sum_{c=1}^C \sum_{i=1}^n (\mathbf{w}_i^{c*})_l^2}{\lambda}} / \sqrt{\lambda}.$$

Together with (25), it follows that the optimal solution of  $\tau$  can be calculated in a closed form:

$$\tau_l = \frac{\sqrt{\sum_{c=1}^C \sum_{i=1}^n (\mathbf{w}_i^{c*})_l^2}}{\sum_{m=1}^d \sqrt{\sum_{c=1}^C \sum_{i=1}^n (\mathbf{w}_i^{c*})_m^2}}. \quad (27)$$

The intuitive interpretation of (27) is as follows: The  $l$ th feature weight  $\tau_l$  is determined by the magnitude of the  $l$ th element in the regression coefficients for all of the clusters which are locally solved at each point. If this element in the regression coefficients has neglectable magnitude for all the clusters at each point, it is likely to indicate that the corresponding feature is unimportant when predicting the confidence of which cluster this point belongs to.

## 4.3 The Complete Algorithm

The complete local learning-based clustering algorithm with feature selection (denoted as LLC-fs) is described in Algorithm 1. The loop stops when the relative variation of the trace value in (7) between two consecutive iterations gets below a threshold (we set it at  $10^{-2}$  in this paper), indicating the partitioning has almost stabilized. After the convergence,  $\mathbf{Y}$  is discretized to obtain the final clustering result with the k-means as in [14].

**Algorithm 1.** Feature selection for local learning-based clustering algorithm.

**input:**  $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^n$ , size of the neighborhood  $k$ , trade-off parameter  $\beta$

**output:**  $\mathbf{Y}, \boldsymbol{\tau}$

- 1 Initialize  $\tau_l = \frac{1}{d}$ , for  $l = 1, \dots, d$ ;
- 2 **while** not converge **do**
- 3 Find  $k$ -mutual neighbors for  $\{\mathbf{x}_i\}_{i=1}^n$ , using the metric defined in (12);
- 4 Construct the matrix  $\mathbf{M}$  in (6) with  $\boldsymbol{\alpha}_i$  given in (17), and then solve the problem (7) to obtain  $\mathbf{Y}$ ;
- 5 Compute  $\mathbf{w}_i^{c*}, \forall i, c$  by (15) and update  $\boldsymbol{\tau}$  using (27);
- 6 **end**

## 5 MULTIPLE KERNEL LEARNING FOR LOCAL LEARNING-BASED CLUSTERING

To deal with some complex data sets, the LLC algorithm can be kernelized as in [3] by replacing the linear ridge regression with the kernel ridge regression. Under such circumstances, selecting a suitable kernel function will become a crucial issue. In this section, we extend the method presented in Section 4 to learn a proper linear combination of several precomputed kernel matrices under the multiple kernel learning framework [26].

In the kernel methods, the symmetric positive semidefinite kernel function  $\mathcal{K} : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$  implicitly maps the original input space into a high-dimensional (possibly infinite) *Reproducing Kernel Hilbert Space* (RKHS)  $\mathcal{H}$ , which is equipped with the inner product  $\langle \cdot, \cdot \rangle_{\mathcal{H}}$  via a nonlinear mapping  $\phi : \mathcal{X} \rightarrow \mathcal{H}$ , i.e.,  $\mathcal{K}(x, z) = \langle \phi(x), \phi(z) \rangle_{\mathcal{H}}$ . Suppose there are altogether  $L$  different kernel functions  $\{\mathcal{K}^{(l)}\}_{l=1}^L$  available for the clustering task in hand. Accordingly, there are  $L$  different associated feature spaces,

denoted as  $\{\mathcal{H}^{(l)}\}_{l=1}^L$ . In general, we do not know which feature space should be used. An intuitive way is to use them all by concatenating all feature spaces into an augmented Hilbert space:  $\tilde{\mathcal{H}} = \bigoplus_{l=1}^L \mathcal{H}^{(l)}$ , and associate each feature space a relevance weight  $\tau_l$  ( $\sum_{l=1}^L \tau_l = 1, \tau_l \geq 0, \forall l$ ), or equivalently the importance factor for kernel function  $\mathcal{K}^{(l)}$ . Later, we will show that performing the LLC in such feature space is equivalent to employing a combined kernel function:  $\mathcal{K}^\tau(x, z) = \sum_{l=1}^L \tau_l \mathcal{K}^{(l)}(x, z)$  for the LLC. A zero weight  $\tau_l$  will correspond to *blend out* the feature space associated with the corresponding kernel analogous to the feature selection in Section 4. Our task is to learn the coefficients  $\{\tau_l\}_{l=1}^L$ , which can lead to a more accurate and robust performance. Again, an alternating algorithm that iteratively performs clustering and estimates the kernel weight is developed.

### 5.1 Update $\mathbf{Y}$ As Given $\tau$

First of all, given a  $\tau$ , the nearest neighbors  $\mathcal{N}_i$  for the LLC algorithm will be refound by the  $\tau$ -weighted squared euclidean distance in  $\tilde{\mathcal{H}}$ , i.e.,

$$\begin{aligned} d_\tau(\mathbf{x}_1, \mathbf{x}_2) &= \|\phi(\mathbf{x}_1) - \phi(\mathbf{x}_2)\|_\tau^2 \\ &= \mathcal{K}^\tau(\mathbf{x}_1, \mathbf{x}_1) + \mathcal{K}^\tau(\mathbf{x}_2, \mathbf{x}_2) - 2\mathcal{K}^\tau(\mathbf{x}_1, \mathbf{x}_2). \end{aligned} \quad (28)$$

Then, the local discriminant function in  $\tilde{\mathcal{H}}$  can be written as follows:

$$f_i^c(\phi(\mathbf{x})) = \phi(\mathbf{x})^T \mathbf{w}_i^c + b_i^c, \quad (29)$$

where  $\phi(\mathbf{x}) = [\phi_1(\mathbf{x})\phi_2(\mathbf{x})\cdots\phi_L(\mathbf{x})]^T \in \mathbb{R}^D$ ,  $\phi_l(\mathbf{x}) \in \mathbb{R}^{D_l}$  is the sample mapped by the  $l$ th kernel function.  $D$  and  $D_l$  are the dimensionalities of  $\tilde{\mathcal{H}}$  and  $\mathcal{H}^{(l)}$ , respectively, with  $\sum_{l=1}^L D_l = D$ . The regression coefficient  $\mathbf{w}_i^c \in \mathbb{R}^D$  and the bias  $b_i^c \in \mathbb{R}$  are estimated via solving the following weighted  $l_2$  norm regularized least square problem:

$$\min_{\{\mathbf{w}_i^c, b_i^c\}} \sum_{c=1}^C \sum_{i=1}^n \left[ \sum_{\mathbf{x}_j \in \mathcal{N}_i} \beta (y_{jc} - \phi(\mathbf{x}_j)^T \mathbf{w}_i^c - b_i^c)^2 + \mathbf{w}_i^{cT} \mathbf{\Lambda}_\tau^{-1} \mathbf{w}_i^c \right], \quad (30)$$

where  $\mathbf{\Lambda}_\tau$  is a diagonal matrix with the vector

$$\underbrace{(\tau_1, \dots, \tau_1)}_{D_1}, \dots, \underbrace{(\tau_L, \dots, \tau_L)}_{D_L}^T$$

in the diagonal, and  $\sum_{l=1}^L \tau_l = 1, \tau_l \geq 0 \forall l$ . Removing the bias term by plugging its optimal solution

$$b_i^c = \frac{1}{n_i} \mathbf{e}_i^T (\mathbf{y}_i^c - \phi(\mathbf{X}_i)^T \mathbf{w}_i^c) \quad (31)$$

into (30), we can reformulate the problem (30) as follows:

$$\min_{\{\mathbf{w}_i^c\}} \sum_{c=1}^C \sum_{i=1}^n [\beta \|\mathbf{\Pi}_i \mathbf{y}_i^c - (\phi(\mathbf{X}_i) \mathbf{\Pi}_i)^T \mathbf{w}_i^c\|^2 + \mathbf{w}_i^{cT} \mathbf{\Lambda}_\tau^{-1} \mathbf{w}_i^c]. \quad (32)$$

Let

$$\zeta_i^c = (\phi(\mathbf{X}_i) \mathbf{\Pi}_i)^T \mathbf{w}_i^c - \mathbf{\Pi}_i \mathbf{y}_i^c, \quad (33)$$

we then have the Lagrangian of problem (32)

$$\begin{aligned} \mathcal{L}(\{\zeta_i^c, \mathbf{w}_i^c, \gamma_i^c\}) &= \sum_{c=1}^C \sum_{i=1}^n (\beta \|\zeta_i^c\|^2 + \mathbf{w}_i^{cT} \mathbf{\Lambda}_\tau^{-1} \mathbf{w}_i^c) \\ &\quad - \sum_{c=1}^C \sum_{i=1}^n \gamma_i^{cT} ((\phi(\mathbf{X}_i) \mathbf{\Pi}_i)^T \mathbf{w}_i^c - \mathbf{\Pi}_i \mathbf{y}_i^c - \zeta_i^c), \end{aligned} \quad (34)$$

where  $\gamma_i^c$ s are the vectors of Lagrangian dual variables, and  $\gamma_i^c \in \mathbb{R}^{n_i}$ . Taking the derivatives of  $\mathcal{L}$  with respect to  $\zeta_i^c$  and  $\mathbf{w}_i^c$  and setting them equal to zero, we obtain

$$\zeta_i^c = -\frac{\gamma_i^c}{2\beta}, \mathbf{w}_i^c = \frac{\mathbf{\Lambda}_\tau \phi(\mathbf{X}_i) \mathbf{\Pi}_i \gamma_i^c}{2}. \quad (35)$$

Finally, we obtain the dual problem

$$\begin{aligned} \max_{\{\gamma_i^c\}} \sum_{c=1}^C \sum_{i=1}^n &-\frac{1}{4\beta} \gamma_i^{cT} \gamma_i^c \\ &- \frac{1}{4} \gamma_i^{cT} \mathbf{\Pi}_i \phi(\mathbf{X}_i)^T \mathbf{\Lambda}_\tau \phi(\mathbf{X}_i) \mathbf{\Pi}_i \gamma_i^c + \gamma_i^{cT} \mathbf{\Pi}_i \mathbf{y}_i^c \\ = \max_{\{\gamma_i^c\}} \sum_{c=1}^C \sum_{i=1}^n &-\frac{1}{4\beta} \gamma_i^{cT} \gamma_i^c - \frac{1}{4} \gamma_i^{cT} \mathbf{\Pi}_i \mathbf{K}_i^\tau \mathbf{\Pi}_i \gamma_i^c + \gamma_i^{cT} \mathbf{\Pi}_i \mathbf{y}_i^c, \end{aligned} \quad (36)$$

which follows from

$$\phi(\mathbf{X}_i)^T \mathbf{\Lambda}_\tau \phi(\mathbf{X}_i) = \sum_{l=1}^L \tau_l \phi_l(\mathbf{X}_i)^T \phi_l(\mathbf{X}_i) = \sum_{l=1}^L \tau_l \mathbf{K}_i^{(l)} = \mathbf{K}_i^\tau, \quad (37)$$

where  $\mathbf{K}_i^{(l)}, \mathbf{K}_i^\tau \in \mathbb{R}^{n_i \times n_i}$  are the base and combined kernel matrices over  $\{\mathbf{x}_j | \mathbf{x}_j \in \mathcal{N}_i\}$ , respectively, i.e.,  $\mathbf{K}_i^{(l)} = [\mathcal{K}^{(l)}(\mathbf{x}_u, \mathbf{x}_v)]$  and  $\mathbf{K}_i^\tau = [\mathcal{K}^\tau(\mathbf{x}_u, \mathbf{x}_v)]$ , for  $\mathbf{x}_u, \mathbf{x}_v \in \mathcal{N}_i$ . For the fixed  $\tau$  constrained on the simplex, the convex combination of the positive semidefinite kernel matrices:  $\mathbf{K}_i^\tau = \sum_{l=1}^L \tau_l \mathbf{K}_i^{(l)}$  is still a positive semidefinite kernel matrix. Therefore, the problem in (36) is a concave one whose unique optimal solution can be obtained analytically, i.e.,

$$\gamma_i^{c*} = 2\beta (\mathbf{I}_i + \beta \mathbf{\Pi}_i \mathbf{K}_i^\tau \mathbf{\Pi}_i)^{-1} \mathbf{\Pi}_i \mathbf{y}_i^c. \quad (38)$$

Together with (31), (35), and (38), the predicted indicator value at point  $\mathbf{x}_i$  for the  $c$ th ( $c = 1, \dots, C$ ) cluster can then be calculated by (29), i.e.,

$$\hat{y}_{ic} = f_i^c(\phi(\mathbf{x}_i)) = \phi(\mathbf{x}_i)^T \mathbf{w}_i^c + b_i^c = \boldsymbol{\alpha}_i^T \mathbf{y}_i^c, \quad (39)$$

with

$$\begin{aligned} \boldsymbol{\alpha}_i^T &= \beta \left( \mathbf{k}_i^\tau - \frac{1}{n_i} \mathbf{e}_i^T \mathbf{K}_i^\tau \right) \\ \mathbf{\Pi}_i \left[ \mathbf{I}_i - (\beta^{-1} \mathbf{I}_i + \mathbf{\Pi}_i \mathbf{K}_i^\tau \mathbf{\Pi}_i)^{-1} \mathbf{\Pi}_i \mathbf{K}_i^\tau \mathbf{\Pi}_i \right] &+ \frac{1}{n_i} \mathbf{e}_i^T, \end{aligned} \quad (40)$$

where  $\mathbf{k}_i^\tau \in \mathbb{R}^{n_i}$  denotes the vector  $[\mathcal{K}^\tau(\mathbf{x}_i, \mathbf{x}_j)]^T$  for  $\mathbf{x}_j \in \mathcal{N}_i$ . The above expression happens to be the same as the one in (17), and the only difference is the way of constructing the  $\mathbf{k}_i^\tau$  and  $\mathbf{K}_i^\tau$ . Note that from (37) and (40), the  $l$ th kernel will be excluded from the inference if the coefficient  $\tau_l$  vanishes.

To sum up, we build the matrix  $\mathbf{M}$  by (6) with  $\alpha_i$  defined in (40), using the combined kernel  $\mathcal{K}^T(\mathbf{x}_i, \mathbf{x}_j) = \sum_{l=1}^L \tau_l \mathcal{K}^{(l)}(\mathbf{x}_i, \mathbf{x}_j)$ .  $\mathbf{Y}$  is then given by the first  $C$  eigenvectors of  $\mathbf{M}$  corresponding to the smallest  $C$  eigenvalues.

## 5.2 Update $\tau$ As Given $\mathbf{Y}$

Subsequently, the  $L$  kernel combination coefficients  $\{\tau_l\}_{l=1}^L$  will be recomputed based on the current estimation of  $\mathbf{Y}$ . Unfortunately, the updating of  $\tau$  in (27) is unable to be applied here because  $\mathbf{w}_i^c \in \mathbb{R}^D$  spans in space  $\tilde{\mathcal{H}}$  of possibly infinite dimension, and the elements of  $\mathbf{w}_i^c$  cannot be expressed in an explicit form. Hence, we estimate  $\tau$  using the reduced gradient descent method, which has been widely used for tackling the optimization problems with equality constraints on the variables (note that  $\tau$  is defined on simplex) [38], [32], [33].

With the fixed  $\mathbf{Y}$  and the neighborhood determined at each point, an optimal  $\tau$  is expected to minimize

$$\mathcal{P}(\tau), \text{ s.t. } \sum_{l=1}^L \tau_l = 1, \tau_l \geq 0, \forall l, \quad (41)$$

where

$$\mathcal{P}(\tau) = \min_{\{\mathbf{w}_i^c\}} \sum_{c=1}^C \sum_{i=1}^n \left[ \beta \|\mathbf{\Pi}_i \mathbf{Y}_{ic} - (\phi(\mathbf{X}_i) \mathbf{\Pi}_i)^T \mathbf{w}_i^c\|^2 + \mathbf{w}_i^{cT} \mathbf{\Lambda}_\tau^{-1} \mathbf{w}_i^c \right]. \quad (42)$$

Then,  $\tau$  will be solved by the update equation:  $\tau^{(new)} = \tau^{(old)} - \eta \nabla \mathcal{P}$ , where  $\eta$  is the step size and  $\nabla \mathcal{P}$  is the reduced gradient. It is expected that the local regression model derived from  $\tau^{(new)}$  is better than the one derived from  $\tau^{(old)}$ . Nevertheless, since both  $\mathbf{Y}$  and  $\mathcal{N}_i$  depend on  $\tau$  as shown in Section 5.1, they need to be recomputed as shown in Section 5.1 once  $\tau$  is updated.

The key issue here is to obtain the derivatives of  $\mathcal{P}(\tau)$  in an analytic form. To this end, we resort to the dual of  $\mathcal{P}(\tau)$  which has been investigated in Section 5.1 and is rewritten as follows:

$$\mathcal{D}(\tau) = \max_{\{\gamma_i^c\}} \sum_{c=1}^C \sum_{i=1}^n -\frac{1}{4\beta} \gamma_i^{cT} \gamma_i^c - \frac{1}{4} \gamma_i^{cT} \mathbf{\Pi}_i \mathbf{K}_i^T \mathbf{\Pi}_i \gamma_i^c + \gamma_i^{cT} \mathbf{\Pi}_i \mathbf{y}_i^c. \quad (43)$$

Note that (32) is convex with respect to  $\mathbf{w}_i^c$ . By the principle of strong duality, we have  $\mathcal{P}(\tau) = \mathcal{D}(\tau)$ . Furthermore, as  $\{\gamma_i^{c*}\}$  (c.f. (38)) maximizes  $\mathcal{D}$ , according to [36],  $\mathcal{D}(\tau)$  is differentiable if  $\{\gamma_i^{c*}\}$ s are unique. Fortunately, this unicity is guaranteed by the unconstrained concave quadratic program in (36). Moreover, as proven in Lemma 2 of [37],  $\mathcal{D}(\tau)$  can be differentiated with respect to  $\tau$  as if  $\{\gamma_i^{c*}\}$  did not depend on  $\tau$  as in (22). Finally, we have

$$\begin{aligned} \frac{\partial \mathcal{P}}{\partial \tau_l} &= \frac{\partial \mathcal{D}}{\partial \tau_l} = -\frac{1}{4} \sum_{c=1}^C \sum_{i=1}^n \gamma_i^{c*T} \mathbf{\Pi}_i \mathbf{K}_i^{(l)} \mathbf{\Pi}_i \gamma_i^{c*} \\ &= -\frac{1}{4} \sum_{i=1}^n \text{trace}(\gamma_i^{*T} \mathbf{\Pi}_i \mathbf{K}_i^{(l)} \mathbf{\Pi}_i \gamma_i^*), \end{aligned} \quad (44)$$

where  $\gamma_i^* = [\gamma_i^{1*}, \dots, \gamma_i^{C*}] \in \mathbb{R}^{n_i \times C}$ .

Note that the equality and nonnegative constraints over  $\tau$  have to be kept inviolate when updating  $\tau$  along the descent gradient direction. According to the principle of reduced gradient descent [38], [32], [33], each element of the reduced gradient  $\nabla \mathcal{P}$  is calculated as follows:

$$\begin{aligned} (\nabla \mathcal{P})_l &= \begin{cases} \frac{\partial \mathcal{P}}{\partial \tau_l} - \frac{\partial \mathcal{P}}{\partial \tau_m}, & \text{if } l \neq m \text{ and } \tau_l > 0; \\ \sum_{\mu \neq m, \tau_\mu > 0} \left( \frac{\partial \mathcal{P}}{\partial \tau_m} - \frac{\partial \mathcal{P}}{\partial \tau_\mu} \right), & \text{if } l = m; \\ 0, & \text{if } \tau_l = 0 \text{ and } \frac{\partial \mathcal{P}}{\partial \tau_l} - \frac{\partial \mathcal{P}}{\partial \tau_m} > 0, \end{cases} \end{aligned} \quad (45)$$

where  $m = \arg \max_l \tau_l$ . The first two cases in (45) enforce the equality by the reduced gradients. If  $\tau_l$  is 0, but the reduced gradient is greater than 0, updating  $\tau_l$  along this decent direction will violate the positivity constraint on  $\tau_l$ ; hence, we set the descent direction at 0 for such an element of  $\nabla \mathcal{P}$ .

When updating  $\tau$  by  $\tau^{(new)} = \tau^{(old)} - \eta \nabla \mathcal{P}$ , we first try  $\eta$  with the maximal admissible step size  $\eta_{max}$  which sets  $\tau_\nu$  to zero, where

$$\nu = \arg \min_{\{l | (\nabla \mathcal{P})_l > 0\}} \frac{\tau_l^{(old)}}{(\nabla \mathcal{P})_l}, \quad \eta_{max} = \frac{\tau_\nu}{(\nabla \mathcal{P})_\nu}. \quad (46)$$

If  $\mathcal{D}(\tau^{(trial)}) \leq \mathcal{D}(\tau^{(old)})$ , where  $\tau^{(trial)} = \tau^{(old)} - \eta_{max} \nabla \mathcal{P}$ ,  $\tau$  will get updated. Otherwise, a one-dimensional line search for  $\eta \in [0, \eta_{max}]$  is applied. Algorithm 2 describes the steps to update  $\tau$ .

**Algorithm 2.** Update kernel weight vector  $\tau$  with the current  $\mathbf{Y}$  and  $\mathcal{N}_i$ .

- 1 Compute the reduced gradient  $\nabla \mathcal{P}$  by (45);
- 2 Compute the maximal admissible step size  $\eta_{max}$  by (46);
- 3  $\tau^{(trial)} = \tau^{(old)} - \eta_{max} \nabla \mathcal{P}$ ;
- 4 Compute  $\mathcal{D}(\tau^{(trial)})$  with  $\{\gamma_i^*\}$  calculated from  $\mathbf{K}^{\tau^{(trial)}} = \sum_{l=1}^L \tau_l^{(trial)} \mathbf{K}^{(l)}$ ;
- 5 **if**  $\mathcal{D}(\tau^{(trial)}) \leq \mathcal{D}(\tau^{(old)})$  **then**
- 6      $\eta = \eta_{max}$ ;
- 7 **else**
- 8     Perform line search for  $\eta \in [0, \eta_{max}]$  along  $\nabla \mathcal{P}$ ;
- 9 **end**
- 10  $\tau^{(new)} = \tau^{(old)} - \eta \nabla \mathcal{P}$ .

## 5.3 The Complete Algorithm

The complete local learning-based clustering algorithm with multiple kernel learning (denoted as LLC-mkl) is presented in Algorithm 3. The loop stops when the relative variation of the trace value in (7) between two consecutive iterations gets below a threshold (we set it at  $10^{-4}$  in this paper), indicating the partitioning has almost stabilized. After the convergence,  $\mathbf{Y}$  is discretized to obtain the final clustering result with the k-means as in [14].

**Algorithm 3.** Multiple kernel learning for local learning-based clustering algorithm.

**input:**  $L$  base kernel matrices  $\mathbf{K}^{(l)}$ 's, size of the neighborhood  $k$ , trade-off parameter  $\beta$

**output:**  $\mathbf{Y}$ ,  $\tau$

- 1 Initialize  $\tau_l = \frac{1}{L}$ , for  $l = 1, \dots, L$ ;

- 2 **while** *not converge* **do**
- 3     Find  $k$ -mutual neighborhoods, using the metric defined in (28);
- 4     Construct the matrix  $\mathbf{M}$  by (6) with  $\alpha_i$  given in (40), and then solve the problem (7) to obtain  $\mathbf{Y}$ ;
- 5     Update  $\tau$  with the steps described in Algorithm 2.
- 6 **end**

## 6 DISCUSSION

### 6.1 Sparse Norm Equivalence

In this section, it will be shown that the weighted  $l_2$  norm regularization associated with the simplex constraint on these weights is equivalent to a well-known sparse-promoting regularization penalty. In the input space, we address this equivalence based on the fact that the infimum of the weighted  $l_2$  norm, with the weights defined on the standard simplex, is equal to a squared special  $l_1$  norm regularization.

**Theorem 1.**

$$\sum_l \inf_{\tau=1, \tau \geq 0} \sum_l \frac{\|\widetilde{\mathbf{W}}_l\|^2}{\tau} = \left( \sum_l \|\widetilde{\mathbf{W}}_l\| \right)^2,$$

where we define  $\|\widetilde{\mathbf{W}}_l\| = \sqrt{\sum_{c=1}^C \sum_{i=1}^n (\mathbf{w}_i^c)_l^2}$ .

**Proof.** This proof is based on the recent works for multitask feature learning [16], [47] and kernel learning [48], [49], and we extend them in the local learning setting. From the Cauchy-Schwarz inequality, we have that

$$\begin{aligned} \sum_l \|\widetilde{\mathbf{W}}_l\| &= \sum_l \tau_l^{\frac{1}{2}} \tau_l^{-\frac{1}{2}} \|\widetilde{\mathbf{W}}_l\| \leq \left( \sum_l \tau_l \right)^{\frac{1}{2}} \left( \sum_l \tau_l^{-1} \|\widetilde{\mathbf{W}}_l\|^2 \right)^{\frac{1}{2}} \\ &\leq \left( \sum_l \tau_l^{-1} \|\widetilde{\mathbf{W}}_l\|^2 \right)^{\frac{1}{2}}, \end{aligned}$$

where equality is obtained when  $\sum_l \tau_l = 1$  and

$$\frac{\|\widetilde{\mathbf{W}}_\mu\|}{\tau_\mu} = \frac{\|\widetilde{\mathbf{W}}_\nu\|}{\tau_\nu}, \quad \forall \mu, \nu,$$

which is equivalent to requiring

$$\tau_l = \|\widetilde{\mathbf{W}}_l\| / \sum_\mu \|\widetilde{\mathbf{W}}_\mu\|.$$

This is satisfied by (27), which always holds in the original input space.  $\square$

In fact,  $\sum_l \|\widetilde{\mathbf{W}}_l\|$  can be viewed as a combination of the  $l_1$  norm regularization on the feature level and the  $l_2$  norm regularization on the cluster level and sample level. In a simplified case where the regression model is not built locally, namely, omitting the subscript  $i$ , then it will just be

$$\sum_l \sqrt{\sum_c (\mathbf{w}^c)_l^2}.$$

It is the so-called Group Lasso regularizer [2], which results in sparse solution at the feature level, i.e.,  $(\mathbf{w}^c)_\mu$  is close to 0,  $\forall c$  on some dimension  $\mu$ . Hence, according to Theorem 1, the weighted  $l_2$  norm regularization with  $\tau$  defined on the simplex should be able to produce at least as sparse as that

of the squared  $\sum_l \|\widetilde{\mathbf{W}}_l\|$  penalty. The case for the feature space can be proven in a similar way; thus we omit it here. Consequently, the sparsity of  $\tau$  follows from (27) in the input space or Algorithm 2 in the feature space.

### 6.2 Complexity Analysis

We now analyze the time complexity of the LLC-fs and the LLC-mkl in each iteration. The complexity for finding  $k$  nearest neighbors is  $O(n^2)$ . Updating of  $\mathbf{Y}$  as given  $\tau$  is just the LLC algorithm whose complexity is bounded by  $O(n^3)$  [3]. In the LLC-fs, the complexity for updating  $\tau$  is  $O(dnC)$  as given  $\mathbf{Y}$ . In the LLC-mkl, the complexity for updating  $\tau$  is of the order  $O(k^3 LnC)$  as given  $\mathbf{Y}$ . Hence, the overall complexity for the LLC-fs and the LLC-mkl in each iteration is  $O(n^3)$ .

### 6.3 Relationship between the LLC-fs and the LLC-mkl

It is not difficult to find that performing the LLC-fs is equivalent to performing the LLC-mkl with a special kernel  $\mathbf{X}^T \text{diag}(\tau) \mathbf{X} = \sum_{l=1}^d \tau_l \mathbf{K}^{(l)} = \sum_{l=1}^d \tau_l \mathbf{f}_l \mathbf{f}_l^T$ , where  $\mathbf{f}_l \in \mathbb{R}^n$  is the  $l$ th column of  $\mathbf{X}^T \in \mathbb{R}^{n \times d}$ , i.e., the  $l$ th feature vector of samples:  $\mathbf{f}_l = (x_{l1}, x_{l2}, \dots, x_{ln})^T$ ,  $l = 1, \dots, d$ . Nevertheless, this way is not desirable because it is quite space consuming to store  $d$  base  $n \times n$  kernels  $\mathbf{K}^{(l)}$ . In particular, when the data set has the moderate sample size  $n$  but a large dimensionality  $d$ , it is highly likely to cause out-of-memory error.

## 7 EXPERIMENTAL RESULTS

In Sections 7.1 and 7.2, we conducted extensive experiments to demonstrate the effectiveness of the proposed feature selection and kernel learning for clustering, respectively.

In all of the experiments, we evaluated the performance with the clustering accuracy (ACC) index [17]. Given a data point  $\mathbf{x}_i$ , let  $c_i$  and  $t_i$  be the obtained cluster label and the true class label from the data, respectively. The ACC is defined as

$$ACC = \frac{\sum_{i=1}^n \delta(t_i, \text{map}(c_i))}{n},$$

where  $n$  is the number of the data set and  $\delta(z_i, z_j)$  is a Kronecker delta function.  $\text{map}(\cdot)$  is a permutation mapping function that maps each cluster index  $c_i$  to a true class label. This optimal mapping can be found with the Kuhn-Munkres algorithm [18]. Furthermore, we simply set the number of clusters at the number of classes in each data set for all the algorithms without considering the selection of the optimal number of clusters, whose studies are, however, beyond the scope of this paper.

### 7.1 Experiments on Feature Selection

Ten benchmark data sets were used in the feature selection experiments, whose characteristics are summarized in Table 1. On each data set, we investigated the performance of the proposed LLC-fs in comparison with the existing LLC algorithm (with linear ridge regression), the baseline  $k$ -means, and spectral clustering. All of these existing algorithms assume all features are equally important.

TABLE 1  
Characteristics of the Data Sets Used  
in the Experiments of Feature Selection

Dataset	#Dimension ( $d$ )	#Sample ( $n$ )	#Class ( $C$ )
wdbc	30	569	2
mfea-fou	76	2000	10
mfea-fac	216	2000	10
mfea-pix	240	2000	10
USPS 4 vs.9	256	1673	2
USPS 0 vs.8	256	2261	2
colon cancer	2000	62	2
SRBCT	2308	63	4
leukemia	3051	38	2
breast cancer	3303	44	2

Furthermore, the LLC-fs was compared with the state-of-the-art unsupervised feature selection method,  $Q-\alpha$  algorithm<sup>4</sup> [13], which is also a *wrapper* approach with iterative eigendecomposition and feature weight estimation, but is global learning-based.

The LLC algorithm has two parameters: the size of the mutual neighbors  $k$  and the trade-off parameter  $\beta$ . For each data sets,  $k$  and  $\beta$  were chosen from the prespecified candidate intervals, respectively. We only report the performance with the best combination of  $k$  and  $\beta$  for the LLC (denoted as LLC-best  $k\beta$ ). As a result, the performance of LLC-best  $k\beta$  reported here might be overoptimistic. For spectral clustering, we used the self-tuning implementation (denoted as SelfTunSpec)<sup>5</sup> [24]. Again, only the best performance among all trials we have tried so far was reported.  $Q-\alpha$  and k-means have no parameters. The mean and the standard deviation of the ACC index over 10 runs were presented. For the proposed LLC-fs algorithm, it also has two parameters:  $k$  and  $\beta$ . We chose them from the same candidate intervals as for the LLC algorithm, and then executed it with each combination 10 times. To demonstrate the robustness of the LLC-fs with respect to  $k$  and  $\beta$  in comparison with the LLC algorithm, we reported the mean and the standard deviation of the ACC index for the LLC-fs with a certain combination, which is usually not the optimal one leading to the best performance, over 10 runs. The parameter sensitivity study for the proposed LLC-fs algorithm will be given at the end of this section.

### 7.1.1 UCI Data Sets

The first four data sets in Table 1 are from the UCI repository [19]. The Wisconsin diagnostic breast cancer data set (wdbc) was used to obtain a binary diagnosis (benign or malignant) based on the features extracted from cell nuclei presented in an image. The mfea-fou, mfea-fac, and

mfea-pix data sets are all from the “multiple feature database” [19]. This database consists of the features of handwritten numerals (“0-9”) extracted from a collection of Dutch utility maps. Digits are represented in several sets of features; we used the data sets with the Fourier coefficients (mfea-fou), the profile correlations (mfea-fac), and the pixel averages (mfea-pix). No preprocessing was performed except on the mfea-fac data set, which has many features of significantly different scales; each feature was then normalized to zero mean and unit variance.

For these data sets,  $k$  and  $\beta$  were chosen from  $10 \sim 40$  and  $[0.1, 10]$ , respectively. For simplicity, we only reported the mean and the standard deviation of the ACC index for the LLC-fs with the combination  $k = 30, \beta = 1$  over 10 runs. The results are summarized in Table 2.

It can be seen that the proposed LLC-fs algorithm almost outperforms the baseline k-means, spectral clustering, and the basic LLC algorithm on all data sets except the mfea-fou, but note that spectral clustering and LLC have used their best parameters. This indicates that feature selection is generally capable of enhancing the clustering performance. Further, although the  $Q-\alpha$  algorithm could improve the performance of the baseline k-means clustering in general, it is less effective than the LLC-fs algorithm. A plausible reason is that many features exhibit similar patterns across all of the handwritten digits; only a few dimensions can discriminate them among different categories. They are believed to be sampled from a low-dimensional nonlinear manifold. Hence, the linear kernel in  $Q-\alpha$  algorithm may not be effective on such data sets. In contrast, LLC-fs directly refines the configuration about the manifold structure from which the clustering will be benefited.

### 7.1.2 Handwritten Digit Data Sets

In this experiment, we focused on the task of clustering on the USPS ZIP code<sup>6</sup> handwritten digits, which are  $16 \times 16$  grayscale images. Each image is thus represented by a 256-dimensional vector. In particular, we considered two binary-class clustering problems, i.e., digits “4 versus 9” and “0 versus 8”, which are known difficult to differentiate.

For the USPS data sets,  $k$  and  $\beta$  were chosen from  $30 \sim 60$  and  $[0.1, 10]$ , respectively. The parameter  $k$  was increased because there are now more than 800 samples per cluster for both data sets, and there are known heavy overlappings within each pair. A larger neighborhood may help obtain a more accurate local prediction. Once again, only the mean and the standard deviation of the ACC index for the LLC-fs, with the combination  $k = 30, \beta = 1$  over 10 runs, are presented. The results are summarized in Table 2.

From Table 2, it can be seen that LLC-fs significantly improves the performance of the other methods on both USPS data sets: USPS49 and USPS08. Actually, the accuracy of the LLC-fs could achieve around 98 percent on both data sets, in a completely unsupervised manner. In contrast, the global approach  $Q-\alpha$  performs even worse than the baseline k-means and spectral clustering on the “4 versus 9” data set that overlaps heavily.

To get a better understanding of what features have been ranked top by our weighting scheme, Fig. 1 shows the top features in the image domain. First, the sorted  $\tau$ s in typical

4. Its MATLAB source code was obtained from the authors of [13].

5. Codes were downloaded from <http://www.vision.caltech.edu/lihi/Demos/SelfTuningClustering.html>.

6. <http://www-stat-class.stanford.edu/~tibs/ElemStatLearn/data.html>.

TABLE 2  
ACC of Various Methods on the Benchmark Data Sets

Dataset	k-means	SelfTunSpec	LLC-best $k/\beta$	Q- $\alpha$	LLC-fs
wdbc	0.8541 $\pm$ 0	0.8858	0.8137	0.8910 $\pm$ 0.0045	<b>0.8910<math>\pm</math>0</b>
mfea-fou	0.6298 $\pm$ 0.0019	0.6060	<b>0.7960</b>	0.5109 $\pm$ 0.0197	0.7463 $\pm$ 0.0005
mfea-fac	0.7098 $\pm$ 0.0701	0.7415	0.8355	0.7561 $\pm$ 0.0410	<b>0.8642<math>\pm</math>0.0219</b>
mfea-pix	0.6783 $\pm$ 0.0814	0.6910	0.8185	0.6788 $\pm$ 0.0140	<b>0.9442<math>\pm</math>0.0002</b>
USPS49	0.7764 $\pm$ 0	0.7669	0.7992	0.5686 $\pm$ 0.0796	<b>0.9871<math>\pm</math>0.0003</b>
USPS08	0.8262 $\pm$ 0	0.8355	0.8859	0.8755 $\pm$ 0.0008	<b>0.9858<math>\pm</math>0.0034</b>
colon	0.6000 $\pm$ 0.0847	0.5484	0.6129	0.5306 $\pm$ 0.0092	<b>0.7419<math>\pm</math>0</b>
SRBCT	0.5047 $\pm$ 0.0380	0.6190	0.5079	0.4174 $\pm$ 0.0130	<b>0.6302<math>\pm</math>0.0107</b>
leukemia	0.6526 $\pm$ 0.0732	0.6579	0.7368	0.7263 $\pm$ 0.0222	<b>0.7895<math>\pm</math>0</b>
breast	0.5454 $\pm$ 0	0.5227	0.5682	0.5273 $\pm$ 0.0096	<b>0.6134<math>\pm</math>0</b>

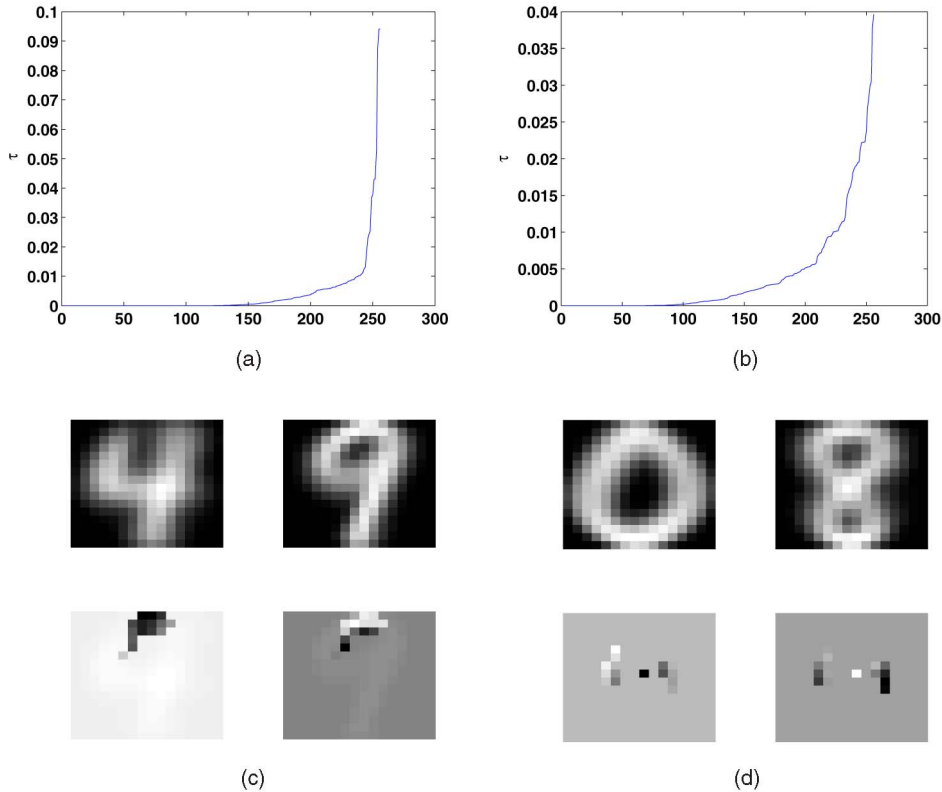


Fig. 1. Unsupervised feature selection by the LLC-fs with  $k = 30, \beta = 1$  on the USPS digits. (a) and (b) show the (sorted)  $\tau$  values on the “4 versus 9” and “0 versus 8” data sets, respectively. In (c) and (d), the first row plots the class mean images, while the second row shows the top 15 features in each mean image ranked by the  $\tau$  weight vector. (a) USPS 4 versus 9, (b) USPS 0 versus 8, (c) USPS 4 versus 9, and (d) USPS 0 versus 8.

runs on the two data sets are presented in Figs. 1a and 1b, respectively. It can be seen that both of the  $\tau$  vectors are sparse, and only a few of the feature weights are above a clear threshold. Next, the 15 top-ranked features are plotted in Figs. 1c and 1d. It is found that these features have generally covered the most distinct regions for each digit pairs, thus resulting in more accurate partitions.

### 7.1.3 Genomic Data Sets

In this experiment, we studied the clustering on four public gene expression data sets: colon cancer [22], SRBCT [21], leukemia [23], and breast cancer [20]. The details of these data sets and the preprocessing steps on the raw public data sets are given below:

- **colon cancer.** This data set consists of 2,000 genes over 62 samples from two classes of colon-cancer

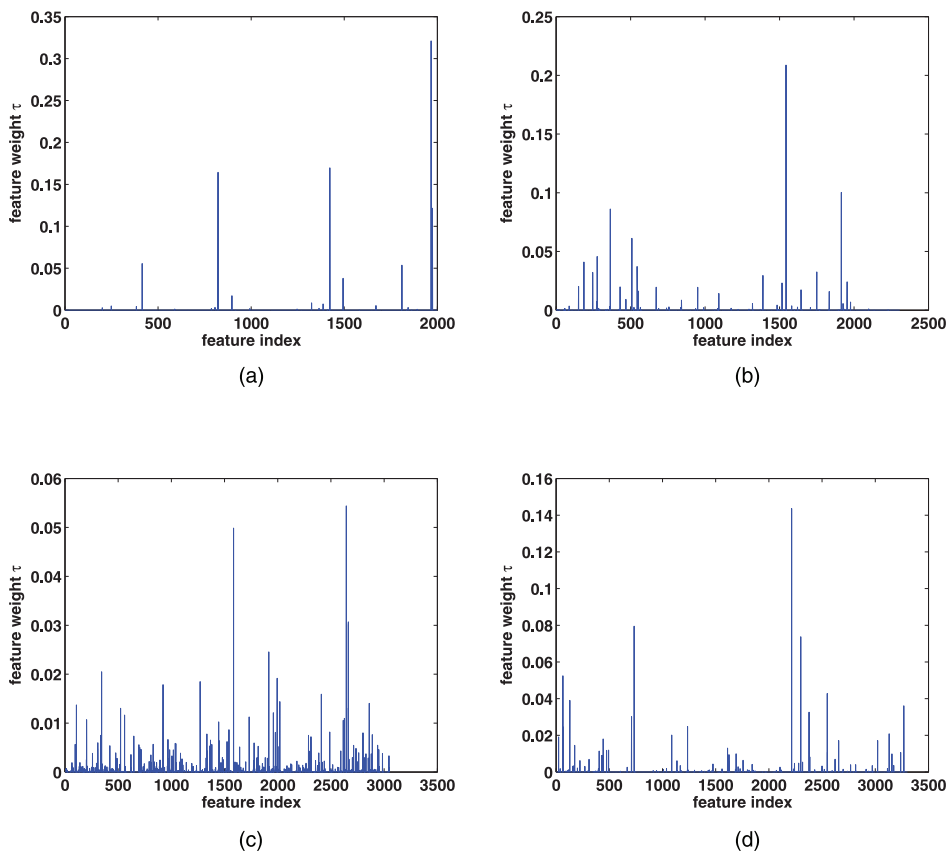


Fig. 2. The feature weight vectors  $\tau$ s learned by LLS-fs with  $k = 30, \beta = 1$  on the genomic data sets. (a) Colon cancer, (b) SRBCT, (c) leukemia, and (d) breast cancer.

patients: 40 normal healthy samples and 22 tumor samples. We preprocessed the data by carrying out a base 10 logarithmic transformation.

- **SRBCT.** This data set contains 63 samples from four classes of small round blue-cell tumors of childhood (SRBCT): 23 Ewing family of tumors, 20 rhabdomyosarcoma, 12 neuroblastoma, and 8 non-Hodgkin lymphoma. The expression profiles already preprocessed in [21] were used.
- **leukemia.** This data set contains 38 samples from two classes of leukemia: 27 acute lymphoblastic leukemia (ALL) and 11 acute myeloid leukemia (AML). The expression values were first thresholded with a floor of 100 and a ceiling of 16,000. Then, we filtered out the genes with  $max/min \leq 5$  or  $(max - min) \leq 500$ , where  $max$  and  $min$  are the maximum and minimum expression values of a gene. Subsequently, 3,051 genes were kept. After a base 10 logarithmic transform, each gene was standardized to zero mean and unit variance across samples.
- **breast cancer.** This data set contains the gene expression levels of 49 tumor samples for 7,129 human genes. The response variable describes the status of the estrogen receptor (ER): 25 samples are ER+, and 24 samples are ER-. The five conflicting samples were excluded from the analysis. We first thresholded the raw data with a floor of 100 and a ceiling of 16,000, and filtered out the genes with  $max/min \leq 10$  or  $(max - min) \leq 1,000$ . Subsequently, 3,303 genes were kept. After a base 10 logarithmic transform, each gene

was standardized to zero mean and unit variance across samples.

For these genomic data, the size  $k$  of the mutual nearest neighbors should be neither too small (otherwise it would be less accurate with the deficient local training sets of high dimensionality) nor too large because of a limited number of samples. We here chose  $k$  from 20 ~ 40 for all the data, except the leukemia data set that has only 38 samples, for which we set it at an element of the set:  $\{20, 25, 30\}$ .  $\beta$  was still selected in  $[0.1, 10]$ . To save space, we reported the results of LLC-fs in Table 2 with the combination  $k = 30, \beta = 1$  only over 10 runs.

It can be seen from Table 2 that the superiority of the LLC-fs over the compared algorithms is remarkable on these high-dimensional data sets with scarce samples. The typical feature weighting results in the 10 runs are also plotted in Fig. 2. For each data set,  $\tau$  is sparse and only a few of them have significant magnitudes, while most feature weights are close to zero. This can explain the reason why the LLC-fs significantly improves the performance of the LLC on these data.

#### 7.1.4 Sensitivity Studies of Parameter $k$ and $\beta$

The effects of parameter  $k$  and  $\beta$  in the LLC-fs algorithm are presented in Fig. 3, where the value of a parameter varies while the other one is fixed. From Figs. 3a and 3b, it can be seen that for all of these four UCI data sets, which have approximately 200 samples per cluster on average, the LLC-fs can typically achieve the promising results when  $k$  and  $\beta$

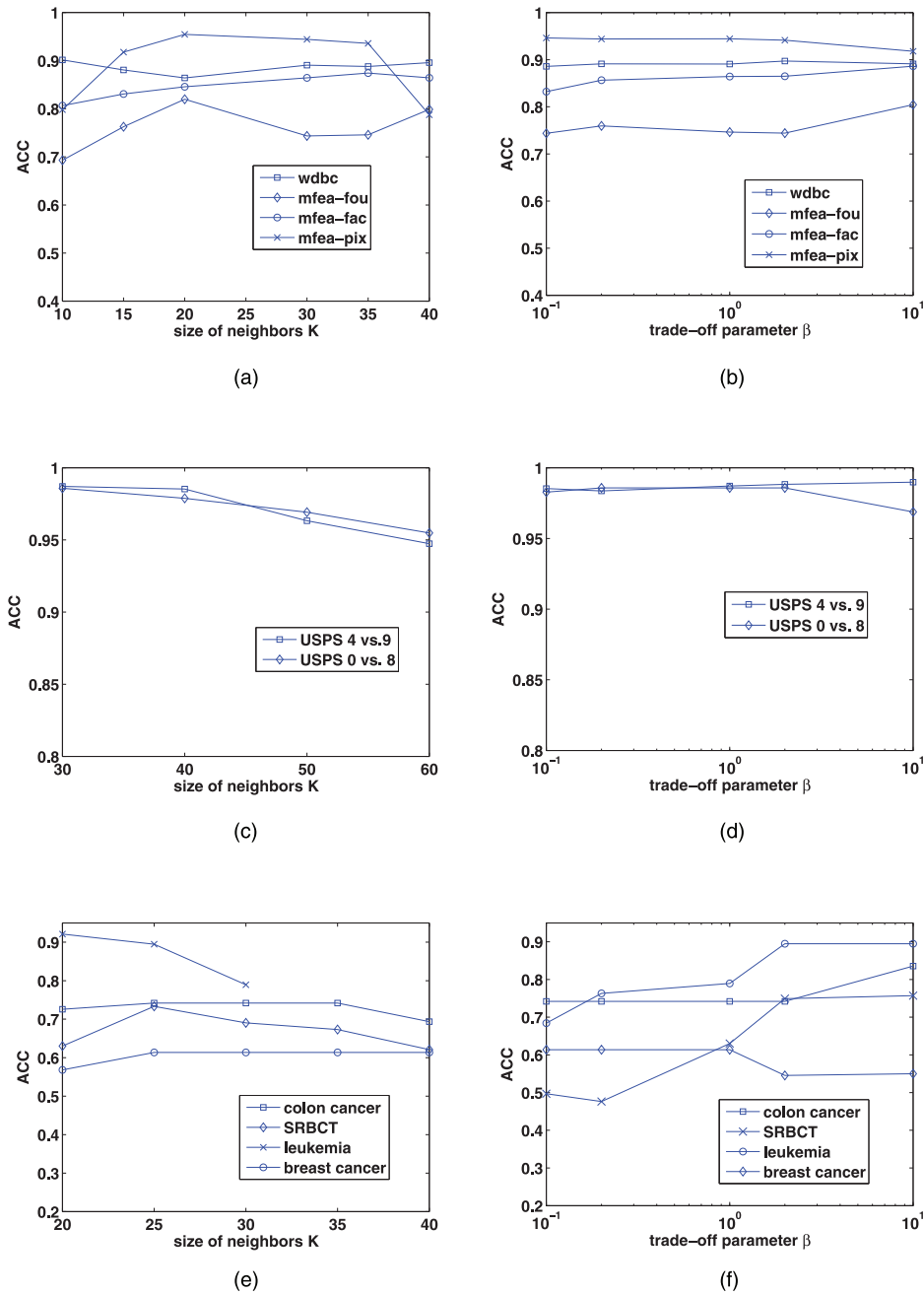


Fig. 3. The parameter sensitivity studies for LLC-fs on the UCI data sets, USPS data sets, and genomic data sets, respectively, where the values on each line represent the average ACC over 10 independent runs. In (a), (c), and (e), the neighborhood size  $k$  varies with  $\beta = 1$ , while (b), (d), and (f) show that the trade-off parameter  $\beta$  varies with  $k = 30$ . (a) and (b) UCI data sets, (c) and (d) USPS data sets, and (e) and (f) genomic data sets.

are chosen from  $15 \sim 30$  and  $[0.1, 10]$ , respectively. In Figs. 3c and 3d, on these two USPS data sets which have more than 800 samples per cluster and are of high dimensionality, the performance of LLC-fs does not vary much when  $k$  and  $\beta$  are chosen from  $30 \sim 60$  and  $[0.1, 10]$ , respectively. Fig. 3e justifies the setting of  $k$  on these genomic data sets. That is, the performance is stable and satisfactory when  $k$  is neither too large nor too small. From Fig. 3f, it is observed that a large value of  $\beta$  would generally lead to better performance on genomic data sets. A plausible reason is that the weighted  $l_2$  norm penalty is large in (10) because there are many irrelevant features. In general, a large trade-off parameter  $\beta$  can balance the model fitting error and the penalty.

### 7.2 Experiments on Kernel Learning

Two sets of experiments were conducted in this section. In the first experiment, we show that the proposed LLC-mkl algorithm outperforms the basic LLC algorithm, in which the local prediction is performed with kernel ridge regression. In the second experiment, we apply the LLC-mkl algorithm, which could provide an easy and principled way to combine the spatial features of different types, e.g., intensity and edge maps, to perform unsupervised face detection.

For comparison, the counterpart unsupervised multiple kernel learning algorithm based on NMF [25] (denoted as NMF-mkl) was conducted. We also compared with the self-tuning spectral clustering [24] (denoted as SelfTunSpec),

TABLE 3  
Characteristics of the Document Data Sets

Dataset	#Sample ( $n$ )	#Class ( $C$ )
CSTR	476	4
WebACE	2340	20
tr11	414	9
tr31	927	7

which tries to build a single best kernel for clustering. Besides, the spectral clustering with multiple views [39] (denoted as Spec-mv) was implemented as well, which generalizes the normalized cut from a single view to multiple views, and each view is represented by normalized adjacency matrix computed with some kernel function. The sensitivity of the proposed LLC-mkl algorithm with respect to  $k$  and  $\beta$  will be presented at the end of this section.

### 7.2.1 Document Data Sets

The characteristics of the benchmark document data sets used in this experiment are summarized in Table 3.

- **CSTR**. This is the data set of the abstracts of technical reports published in the Department of Computer Science at a university between 1991 and 2002. The data set contains 476 abstracts, which are divided into four research areas: Natural Language Processing, Robotics/Vision, Systems, and Theory.
- **WebACE**. This data set is from the WebACE project, and it contains 2,340 documents consisting of news articles from Reuters news service with 20 different topics in October 1997.
- **tr11** and **tr31**. Both of the data sets are from the CLUTO toolkit [45]; they contain 414 and 927 articles categorized into 9 and 7 topics, respectively.

To preprocess the CSTR and WebACE data sets, we removed the stop words using a standard stop list, all HTML tags were skipped, and all header fields except subject and organization in the posted articles were ignored. Then, the original data sets were represented by the Bag-of-Words model. The data sets associated with the CLUTO toolkit had already been preprocessed. For all data sets, we only used the top 1,000 words by mutual information with the class labels.

We applied the LLC-mkl with 10 precomputed base kernels altogether, i.e., seven RBF kernels  $\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2/2\delta^2)$ , with  $\delta = \text{const} * D$ , where  $D$  is the maximum distance between samples and  $\text{const}$  varies in the prespecified range of  $\{0.01, 0.05, 0.1, 1, 10, 50, 100\}$ , two polynomial kernels  $\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) = (1 + \mathbf{x}_i^T \mathbf{x}_j)^d$  with degree  $d = \{2, 4\}$ , and a cosine kernel  $\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j / (\|\mathbf{x}_i\| \cdot \|\mathbf{x}_j\|)$ . All of the kernels have been normalized through the formula:  $\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) / \sqrt{\mathcal{K}(\mathbf{x}_i, \mathbf{x}_i)\mathcal{K}(\mathbf{x}_j, \mathbf{x}_j)}$ . Besides, we also implemented the case where, each time a single candidate kernel  $\mathcal{K}^{(l)}$  ( $l = 1, \dots, 10$ ) was adopted in the LLC algorithm, the best (denoted as LLC-bkernel) and the worst (denoted as LLC-wkernel) performance out of the 10 kernels were reported. The NMF-mkl was applied on the same 10 base kernels. The adjacency matrix in SelfTunSpec [24] was built by its local scaling method [24] on the data set. As for Spec-mv [39], the combination weights are unknown a priori and it does not involve the reestimation for them. Without loss of generality, we therefore applied the uniform weighting for the 10 kernels. For the NMF-mkl, SelfTunSpec, and Spec-mv, we reported the best accuracy only among the extensive trials of their free parameters. For the LLC-mkl, the mean and standard deviation of ACC with  $k = 30, \beta = 10$  over 10 runs were reported. The results are summarized in Table 4.

From Table 4, it can be seen that there is a big gap between the best and the worst performance of the LLC with the different choices of kernel. On the tr11 and tr31 data sets, the performance of the LLC-mkl is close to that of the LLC with the best kernel, but the LLC-mkl is apparently more desirable from the practical viewpoint, where we often do not know which kernel is the best a priori. On the CSTR and WebACE data sets, the LLC-mkl even outperforms the LLC with the best kernel. In fact, by combining multiple kernels and exploiting the complementary information revealed from the different kernels, the LLC-mkl indeed improves the robustness and accuracy of the LLC. Moreover, the text data are believed to lie on a low-dimensional manifold because it is impossible for them to fill in the entire high-dimensional space [4]. We can also observe that the performance of LLC-mkl, which utilizes the manifold information, is consistently superior to that of NMF-mkl, which does not utilize such information. From Table 4, we can find that the LLC-mkl and NMF-mkl both outperform the selfTunSpec, which tries to construct a single “best” kernel in this experiment. Moreover, the performance of the Spec-mv is generally inferior to that of the LLC-mkl because it cannot update the combination weights. The algorithm with equal weights for all the adjacency matrices may tend to be affected by the improper kernel functions adopted.

TABLE 4  
Accuracies of Various Methods on the Document Data Sets

Data Set	LLC-wkernel	LLC-bkernel	LLC-mkl	NMF-mkl	SelfTunSpec	Spec-mv
CSTR	0.3487	0.7374	<b>0.8508±0.0012</b>	0.6387	0.5210	0.3741
WebACE	0.2436	0.4885	<b>0.6316±0.0215</b>	0.4960	0.4880	0.3286
tr11	0.4251	<b>0.5966</b>	0.5609±0.0166	0.5145	0.4106	0.5242
tr31	0.5297	<b>0.6721</b>	0.6512±0.0007	0.5372	0.4412	0.5599

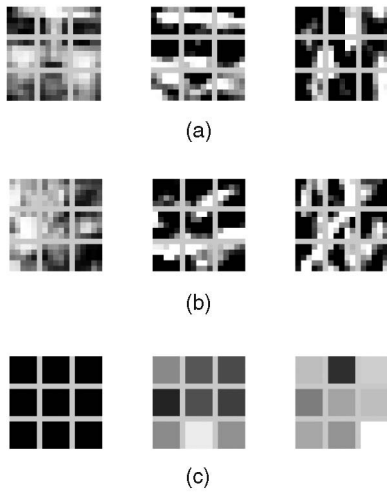


Fig. 4. (a) A sample face image and its edge map in horizontal and vertical orientations, (b) a sample nonface image and its edge map in horizontal and vertical orientations, and (c) weight maps obtained by the LLC-mkl with  $k = 60, \beta = 10$ , where the patches with the lighter intensities have larger weight values.

### 7.2.2 Unsupervised Face Detection

This experiment was conducted on the MIT CBCL Face data set, which consists of 31,022 cropped face and nonface images in total. We randomly selected a subset of 1,000 face images and 1,000 nonface images, rescaled each image to  $15 \times 15$  pixels, and then processed with the histogram equalization.

Based on the fragment idea of [35], [44], it is rational to assume that the different local regions in an image have different relevance in determining whether the image contains a face or not. Therefore, we divided each image into nine nonoverlapping patches of size  $5 \times 5$ . Each patch is considered as a different source which contains the different spatial information. Note that we only used nonoverlapping patches for simplicity, but it is quite straightforward to apply the proposed method to use arbitrary, possibly overlapping patches. In addition to these intensity patches, we also computed the edge maps, i.e., the Sobel filter responses on each raw image for both the horizontal and vertical orientations. Similar to the original images, each edge map image was divided into nine patches. Therefore, there are 27 patches in total for each image: nine patches from the original image, nine patches from the horizontal edge maps, and nine patches from the vertical edge maps. In order to combine these fragments in a principled way,  $\mathcal{K}^{(l)}$  ( $1 \leq l \leq 27$ ) is defined as the cosine kernel restricted to the  $l$ th patch between each pair of images, and then the combination of  $\mathcal{K}^{(l)}$ s which can lead to

a more accurate unsupervised partition on these images is learned by LLC-mkl.

The obtained weight maps indicating the weights for different patches are shown in Fig. 4c. It can be seen that the weights for edge map patches generally dominate the weighting solution in this task, while the intensity patches seem to be less discriminative than the former. To confirm the rationality of this weighting result, we ran the LLC algorithm with the uniformly combined cosine kernels computed from the nine raw intensity patches (denoted as LLC-pix) and the 18 edge map ones (denoted as LLC-tex), respectively. The results based on these two types of features are listed in Table 5. It can be found that such a weighting result is reasonable because the edge map features lead to higher accuracy in comparison with the intensity features (0.9740 versus 0.9660). Besides, we also conducted the experiment where these 27 cosine kernels were uniformly combined and then applied to the LLC algorithm (denoted as LLC-uniWei). Although it outperforms the case where the intensity or edge map features are used alone (i.e., LLC-pix and LLC-tex), its performance is still worse than that of the nontrivial weighting solution obtained by the LLC-mkl, which has automatically assigned weight on each patch (see Fig. 4c). A reasonable explanation is that the uniform weighting cannot make full use of the complementary information among these kernels. This is further confirmed by the experiment with the Spec-mv, where 27 adjacency matrices were formed on each patch by the Gaussian kernel function with local scaling [24] and equal weights were associated with these matrices for general purpose. It is observed from Table 5 that the performance of such multiview spectral clustering falls behind that of the LLC-mkl. Moreover, the LLC-mkl again yields a more accurate partition than the NMF-mkl with the same 27 cosine kernels, as well as the SpecTunSpec performed on the data of 675 dimensions by simply stacking up the 27 patches into a “big” vector.

### 7.2.3 Sensitivity Studies of Parameter $k$ and $\beta$

The effects of these two parameters, i.e.,  $k$  and  $\beta$ , on the performance of LLC-mkl are presented in Fig. 5. From Figs. 5a and 5b, it can be observed that, for the document data sets where there are no more than 300 samples per class on average, the proposed LLC-mkl algorithm with  $k \in 30 \sim 50$  and  $\beta \in [0.01, 10]$  could produce considerably accurate results and the corresponding performance does not vary much. From Figs. 5c and 5d, for the face data set in which there are 1,000 samples per class, the neighborhood size  $k$  and the trade-off parameter  $\beta$  chosen from  $60 \sim 100$  and the range

TABLE 5  
Confusion Matrices and Accuracies of Various Methods for the Unsupervised Face Detection

	LLC-pix		LLC-tex		LLC-uniWei		LLC-mkl		NMF-mkl		SelfTunSpec		Spec-mv	
	face	non-face	face	non-face	face	non-face	face	non-face	face	non-face	face	non-face	face	non-face
face	963	37	999	1	1000	0	1000	0	962	38	935	65	952	48
non-face	31	969	51	949	50	950	1	999	558	442	16	984	4	996
Accuracy	0.9660		0.9740		0.9750		<b>0.9995</b>		0.7020		0.9595		0.9740	

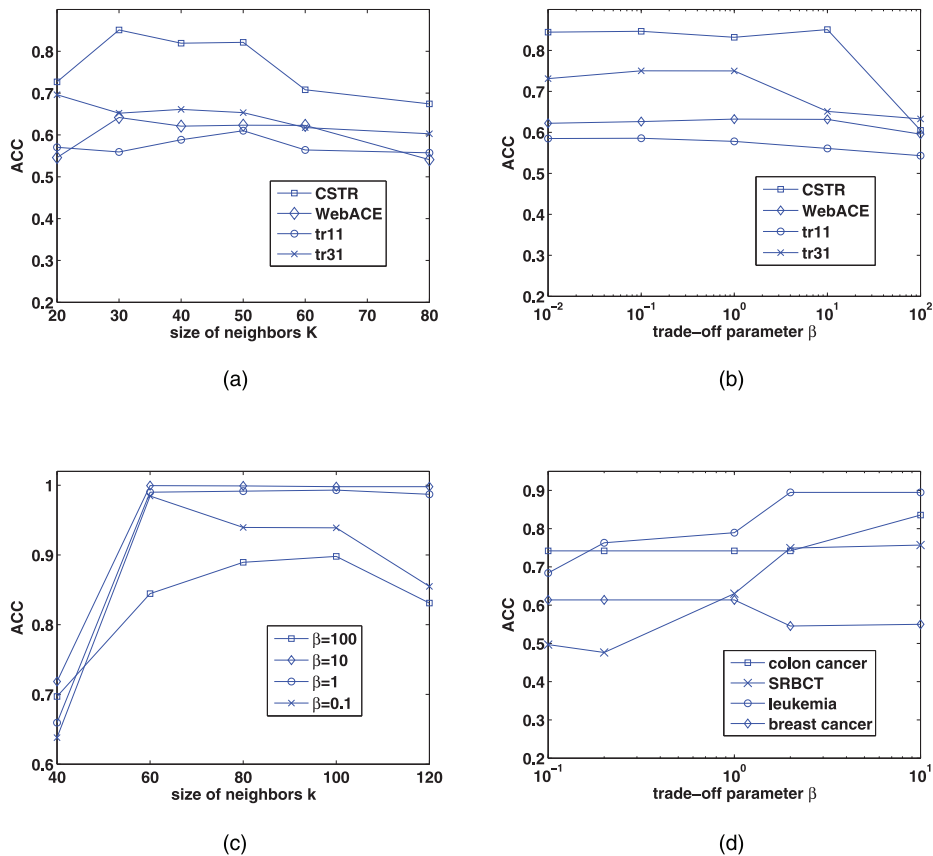


Fig. 5. The parameter sensitivity studies of the LLC-mkl algorithm, where the values on each line represent the average ACC over 10 independent runs. In (a), the size  $k$  of neighborhood varies with  $\beta = 10$ , in (b),  $\beta$  varies with  $k = 30$ , (c) shows the change of ACC over  $k$ , while (d) shows the change of ACC over the trade-off parameter  $\beta$ . (a) and (b) Document data sets, and (c) and (d) MIT CBCL data set.

of  $[1, 10]$ , respectively, could result in considerably accurate and stable performance.

## 8 CONCLUSION

In this paper, a novel feature selection method and a kernel learning method have been proposed for the local learning-based clustering. These two algorithms have been developed in a unified approach under the same regularization framework. The resulting feature weights or kernel combination coefficients are sparse. Experimental results have demonstrated that the proposed feature selection and kernel learning methods are able to improve the robustness and accuracy of the basic local learning clustering. Furthermore, on the data sets with manifold structure, e.g., the high-dimensional sparse data sets, the performance of the proposed methods generally outperforms that of their global learning-based counterparts which ignore the manifold information.

A common limitation of both proposed algorithms is that each algorithm is not optimizing the same objective function in the iterative estimation of  $\mathbf{Y}$  and  $\tau$ . Currently, we have just found the convergence empirically without a theoretical guarantee. The future work will focus on solving the feature selection/kernel learning problem and the clustering problem with a unified objective function.

## ACKNOWLEDGMENTS

The work described in this paper was jointly supported by grants from the Research Grant Council of Hong Kong SAR (Project Nos. HKBU 210306 and HKBU 210309) and a Faculty Research Grant of Hong Kong Baptist University under Project Nos. FRG/07-08/II-54 and FRG2/08-09/122. Yiu-Ming Cheung is the corresponding author for this paper.

## REFERENCES

- [1] S. Shortreed and M. Meila, "Unsupervised Spectral Learning," *Proc. Conf. Uncertainty in Artificial Intelligence*, pp. 534-541, 2005.
- [2] M. Yuan and Y. Lin, "Model Selection and Estimation in Regression with Grouped Variables," *J. Royal Statistical Soc. Series B*, vol. 68, no. 1, pp. 49-67, 2006.
- [3] M. Wu and B. Schölkopf, "A Local Learning Approach for Clustering," *Advances in Neural Information Processing Systems*, vol. 19, pp. 1529-1536, MIT Press, 2007.
- [4] F. Wang, C.S. Zhang, and T. Li, "Regularized Clustering for Documents," *Proc. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval*, pp. 95-102, 2007.
- [5] I. Guyon and A. Elisseeff, "An Introduction to Variable and Feature Selection," *J. Machine Learning Research*, vol. 3, nos. 7/8, pp. 1157-1182, 2003.
- [6] Y.M. Cheung and H. Zeng, "Local Kernel Regression Score for Selecting Features of High-Dimensional Data," *IEEE Trans. Knowledge and Data Eng.*, vol. 21, no. 12, pp. 1798-1802, Dec. 2009.
- [7] X. He, D. Cai, and P. Niyogi, "Laplacian Score for Feature Selection," *Advances in Neural Information Processing Systems*, vol. 18, pp. 507-514, MIT Press, 2005.
- [8] Z. Zhao and H. Liu, "Spectral Feature Selection for Supervised and Unsupervised Learning," *Proc. Int'l Conf. Machine Learning*, pp. 1151-1158, 2007.

- [9] M. Dash, K. Choi, P. Scheuermann, and H. Liu, "Feature Selection for Clustering—A Filter Solution," *Proc. IEEE Int'l Conf. Data Mining*, pp. 115-122, 2002.
- [10] J.G. Dy and C.E. Brodley, "Feature Selection for Unsupervised Learning," *J. Machine Learning Research*, vol. 5, pp. 845-889, 2004.
- [11] M.H.C. Law, A.K. Jain, and M.A.T. Figueiredo, "Feature Selection in Mixture-Based Clustering," *Advances in Neural Information Processing Systems*, vol. 15, pp. 609-616, MIT Press, 2003.
- [12] V. Roth and T. Lange, "Feature Selection in Clustering Problems," *Advances in Neural Information Processing Systems*, vol. 16, pp. 473-480, MIT Press, 2004.
- [13] L. Wolf and A. Shashua, "Feature Selection for Unsupervised and Supervised Inference: The Emergence of Sparsity in a Weight-Based Approach," *J. Machine Learning Research*, vol. 6, pp. 1855-1887, 2005.
- [14] A. Ng, M. Jordan, and Y. Weiss, "On Spectral Clustering: Analysis and an Algorithm," *Advances in Neural Information Processing Systems*, vol. 14, pp. 849-856, MIT Press, 2002.
- [15] S.X. Yu and J. Shi, "Multiclass Spectral Clustering," *Proc. IEEE Int'l Conf. Computer Vision*, pp. 313-319, 2003.
- [16] A. Argyriou, T. Evgeniou, and M. Pontil, "Multi-Task Feature Learning," *Advances in Neural Information Processing Systems*, pp. 41-48, MIT Press, 2007.
- [17] H. Zha, C. Ding, M. Gu, X. He, and H. Simon, "Spectral Relaxation for K-Means Clustering," *Advances in Neural Information Processing Systems*, vol. 14, pp. 1057-1064, MIT Press, 2001.
- [18] C.H. Papadimitriou and K. Steiglitz, *Combinatorial Optimization: Algorithm and Complexity*. Dover, 1998.
- [19] D.J. Newman, S. Hettich, C.L. Blake, and C.J. Merz, UCI Repository of Machine Learning Databases, <http://www.ics.uci.edu/~mllearn/MLRepository.html>, 1998.
- [20] M. West, C. Blanchette, H. Dressman, E. Huang, S. Ishida, R. Spang, H. Zuzan, J.A. Olson, Jr., J.R. Marks, and J.R. Nevins, "Predicting the Clinical Status of Human Breast Cancer by Using Gene Expression Profiles," *Proc. Nat'l Academy of Sciences USA*, vol. 98, no. 20, pp. 11462-11467, 2001.
- [21] J. Khan et al., "Classification and Diagnostic Prediction of Cancers Using Gene Expression Profiling and Artificial Neural Networks," *Nature Medicine*, vol. 7, pp. 673-679, 2001.
- [22] U. Alon, N. Barkai, D.A. Notterman, K. Gish, S. Ybarra, D. Mack, and A.J. Levine, "Broad Patterns of Gene Expression Revealed by Clustering Analysis of Tumor and Normal Colon Tissues Probed by Oligonucleotide Arrays," *Proc. Nat'l Academy of Sciences USA*, vol. 96, no. 12, pp. 6745-6750, 1999.
- [23] T.R. Golub et al., "Molecular Classification of Cancer: Class Discovery and Class Prediction by Gene Expression Monitoring," *Science*, vol. 286, no. 5439, pp. 531-537, 1999.
- [24] L. Zelnik-Manor and P. Perona, "Self-Tuning Spectral Clustering," *Advances in Neural Information Processing Systems*, vol. 17, pp. 1601-1608, MIT Press, 2005.
- [25] T. Lange and J. Buhmann, "Fusion of Similarity Data in Clustering," *Advances in Neural Information Processing Systems*, vol. 18, pp. 723-730, MIT Press, 2006.
- [26] G.R.G. Lanckriet, N. Cristianini, P. Bartlett, M.I.E. Ghaoui, and M.I. Jordan, "Learning the Kernel Matrix with Semidefinite Programming," *J. Machine Learning Research*, vol. 5, pp. 27-72, 2004.
- [27] J. Ye, S. Ji, and J. Chen, "Multi-Class Discriminant Kernel Learning via Convex Programming," *J. Machine Learning Research*, vol. 9, pp. 719-758, 2008.
- [28] F.R. Bach and M.I. Jordan, "Learning Spectral Clustering, with Application to Speech Separation," *J. Machine Learning Research*, vol. 7, pp. 1963-2001, 2006.
- [29] O. Chapelle and V. Vapnik, "Model Selection for Support Vector Machines," *Advances in Neural Information Processing Systems*, vol. 12, pp. 230-236, MIT Press, 2000.
- [30] B. Scholköpfung and A.J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2002.
- [31] E.K.P. Chong and S.H. Zak, *An Introduction to Optimization*. John Wiley and Sons Inc., 2001.
- [32] A. Rakotomamonjy, F. Bach, S. Canu, and Y. Grandvalet, "More Efficiency in Multiple Kernel Learning," *Proc. Int'l Conf. Machine Learning*, pp. 775-782, 2007.
- [33] A. Rakotomamonjy, F. Bach, Y. Grandvalet, and S. Canu, "SimpleMKL," *J. Machine Learning Research*, vol. 9, pp. 2491-2521, 2008.
- [34] H. Valizadegan and R. Jin, "Generalized Maximum Margin Clustering and Unsupervised Kernel Learning," *Advances in Neural Information Processing Systems*, pp. 1417-1424, MIT Press, 2007.
- [35] S. Ullman, M. Vidal-Naquet, and E. Sali, "Visual Features of Intermediate Complexity and Their Use in Classification," *Nature Neuroscience*, vol. 5, no. 7, pp. 683-687, 2002.
- [36] J.F. Bonnans and A. Shapiro, *Perturbation Analysis of Optimization Problems*. Springer, 2000.
- [37] O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee, "Choosing Multiple Parameters for Support Vector Machines," *Machine Learning*, vol. 26, no. 1, pp. 131-159, 2002.
- [38] P.H. Calamai and J.J. Moré, "Projected Gradients Methods for Linearly Constrained Problems," *Math. Programming*, vol. 39, no. 1, pp. 93-116, 1987.
- [39] D.Y. Zhou and C.J.C. Burges, "Spectral Clustering and Transductive Learning with Multiple Views," *Proc. Int'l Conf. Machine Learning*, pp. 1159-1166, 2007.
- [40] P. Mitra, C.A. Murthy, and S.K. Pal, "Unsupervised Feature Selection Using Feature Similarity," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 24, no. 3, pp. 301-312, Mar. 2002.
- [41] J. Chen, Z. Zhao, J. Ye, and H. Liu, "Nonlinear Adaptive Distance Metric Learning for Clustering," *Proc. ACM SIGKDD*, pp. 123-132, 2007.
- [42] D.Y. Yeung, H. Chang, and G. Dai, "Learning the Kernel Matrix by Maximizing a KFD-Based Class Separability Criterion," *Pattern Recognition*, vol. 40, no. 7, pp. 2021-2028, 2007.
- [43] M.H.C. Law, M.A.T. Figueiredo, and A.K. Jain, "Simultaneous Feature Selection and Clustering Using Mixture Models," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 26, no. 9, pp. 1154-1166, Sept. 2004.
- [44] G. Bakır, M. Wu, and J. Eichhorn, "Maximum-Margin Feature Combination for Detection and Categorization," technical report, Max Planck Inst. for Biological Cybernetics, 2005.
- [45] G. Karypis, CLUTO-A Clustering Toolkit, <http://www-users.cs.umn.edu/~karypis/cluto/>, 2002.
- [46] B. Long, P.S. Yu, and M.Z.F. Zhang, "General Model for Multiple View Unsupervised Learning," *Proc. SIAM Int'l Conf. Data Mining*, pp. 822-833, 2008.
- [47] A. Argyriou, T. Evgeniou, and M. Pontil, "Convex Multi-Task Feature Learning," *Machine Learning*, vol. 73, no. 3, pp. 243-272, 2008.
- [48] C.A. Micchelli and M. Pontil, "Learning the Kernel Function via Regularization," *J. Machine Learning Research*, vol. 6, pp. 1099-1125, 2005.
- [49] C.A. Micchelli and M. Pontil, "Feature Space Perspectives for Learning the Kernel," *Machine Learning*, vol. 66, no. 2, pp. 297-319, 2007.



**Hong Zeng** received the PhD degree in computer science from Hong Kong Baptist University in 2010. He is currently with the Robotic Sensor and Control Laboratory (RSCL) in the School of Instrument Science and Engineering, Southeast University, China. His research interests are in the areas of pattern recognition, machine learning, data mining, and haptics. He is a member of the IEEE.



**Yiu-ming Cheung** received the PhD degree from the Department of Computer Science and Engineering, Chinese University of Hong Kong, in 2000. Currently, he is an associate professor in the Department of Computer Science at Hong Kong Baptist University. His research interests include machine learning, information security, signal processing, pattern recognition, and data mining. He is the founding chair of the Computational Intelligence Chapter of the IEEE Hong Kong Section. Also, he is a senior member of the IEEE and the ACM. More details can be found at <http://www.comp.hkbu.edu.hk/~ymc>.